# Step-by-Step Guide to Building Routes for Microsoft Train Simulator

By Michael Vone

# Table of Contents

## FOREWORD

Microsoft Train Simulator (MSTS) comes tantalizingly close to allowing everyone to build his/her own routes. However, there's a lot to learn in order to become a proficient route builder. While trying to make my own routes I have learned a lot about the powerful MSTS Editors and Tools. In order to help others get a fast start, I decided to put together this guide to assist beginners and non-experts get over the learning curve.

This guide will show you how to work with the **Route Geometry Extractor** (RGE), the **Route Editor** (RE) and parts of the **Activity Editor** (AE). You'll find that it has in depth information about **laying tracks**, **shaping terrain**, and **adding textures, transfers and objects**. It also includes information about **using the Activity Editor** (AE), and about **preparing a route for distribution** to other users.

I have assembled this guide partly from my own experience in building a few routes for MSTS. But much essential help came from the Train-Sim.com Message Area: I gratefully thank all those who contributed to the Message Forums there for sharing their expertise. I also thank numerous readers for sending me directly excellent suggestions for improvements. I have tested and adapted many of their suggestions for this guide.

For easy identification, major changes and additions since the last published versions (1.106 and 2) are marked as [NEW SINCE V1.106] or [NEW SINCE V2].

[NEW SINCE V1.106] Version 2 adds **many updates and new details, as well as several new topics**. These include:
- **adding non-default tracks and roads**, including XTracks (section 3.3.13);
- **changing the track's appearance** (section 3.5);
- **linking distant places with straight tracks** (section 3.4.5);
- **controlling forests** (section 3.12.6);
- an extensive discussion of **how to control the MSTS Environment**, which includes fog, sky, water, rain or snow, and wind (section 3.15);
- a discussion of **how activities depend on the installed types of switches and signals** (sections 5.4 to 5.7);
- **making Introductory Train Rides** (section 5.8);
- a much amplified discussion of **how to prepare documentation and illustrations** (such as the logo, map and "detail" images that the user sees at start-up of a route);
- a discussion of **the file size of a route** (section 7.1), and of **the add-on utility Route-Riter** to compress routes and prepare them for distribution (section 7.2.1);
- **several new appendices** dealing with degrees of curvature (App. E), the structure of the route folder (App. F), a suggested checklist for route building (App. G), changing and making texture graphics (App. H), color definitions in the hexadecimal code used by MSTS (App. I), and controlling the camera in MSTS, useful for screenshots (App. J).

[NEW SINCE V1.106] **A new add-on tool accompanies this Guide: Object Rotator** is an Excel spreadsheet for advanced users of MSTS that rotates objects accurately. It is used to modify an object's QDirection and/or Position in world tile files (*.w). It requires Microsoft

Excel (it will not function with an Excel viewer). Major applications of this tool are the precise alignment of bridges and platforms along tracks or roads, the accurate orientation of objects, the precise lining up of rows of objects, along a straight or curved line, and the accurate aiming of straight tracks or roads between distant points.

[NEW SINCE V2] Version 3 adds **further items** to Version 2. These items were prepared in 2002-2003, but not verified since then. In particular, links to web sites and references to external software have not been updated after 2003. The current version of this Guide mainly adds the following items:
- **how tiles are numbered** (see section 2.7);
- a discussion of **how to avoid bad track joints in sloped curves** (section 3.3.12);
- **making terrain cover the trackbed** (section 3.5.5);
- **recovering backed-up terrain shapes** (section 3.5.6);
- **covering the trackbed with roads** (section 3.8.4);
- **showing deep snow** (section 3.9.1d);
- **drawing on terrain textures** (section 3.9.3);
- **making new terrain textures and transfers** (section 3.9.6);
- **snapping objects together** (section 3.10.2);
- **resizing objects** (section 3.10.11);
- **showing snow on objects when it does not snow** (section 3.10.12);
- **de-electrifying a route and removing gantries** (section 3.14.5);
- **fixing the "flashing water" problem** (sections 3.15.3a and 3.15.3c);
- **turning a train around in an Activity** (section 5.2).

For further information, I suggest that you visit online train simulation forums for newer discoveries and tips on these and other matters.

Michael Vone
August 2013

## HOW TO USE THIS GUIDE

**Building "First Route": a project for the first-time user of the MSTS Editors and Tools**

**This guide includes a project that you'll complete in order to master the techniques of route building. You'll build a project called "First Route". For easy recognition, this project appears in blue boxes like the box which you are now reading.**

**For the first-time user, I strongly suggest that you complete this small project first, before reading anything else** - just move from one blue box to the next, in sequence.

These blue boxes will give you detailed step-by-step instructions and the experience you'll need to build more complex routes later. Of course you can also read the information outside of these boxes if you have questions about what I've asked you to do or about other options.

After finishing your "First Route", you can return to read the rest of this guide to learn more details and more options, as well as more advanced topics.

**This guide is accompanied by my version of "First Route"**: it is named "`First Route MV3`". I created it at the same time as I was writing the "First Route" project of this guide, so it is virtually identical to what this project asks you to produce. The "decoration" with textures, objects, hills, roads, sounds and environments (weather) is a bit more complete and specific than described in this project, so you can see a more "final" version here (it did not import more textures, objects, etc., than is described in the project, but added more copies into the route; however, I created a number of textures for the environment: these are included within "`First Route MV3`").

My version of "First Route" is contained in the file `FirstRouteMV3.zip`. Version 3 adds 12 different environments (weather) to the route, to illustrate the new section 3.15 on the environment in MSTS. You'll find the installation instructions in the text file `install.txt` (and repeated in the documentation). You can use these as a starting point for writing instructions for your own projects. The installation installs "`First Route MV3`" in the folder "First Route" (with a space). The project will ask you to produce a different folder (Directory) name "FirstRoute" (without space) and a different route name "First Route", so your and my versions of the route will not be confused: you will be able to open either one in the Route Editor, and you will be able to use either version in MSTS.

**If you are ready to start your "First Route", jump to the next blue box below!**

Text outside blue boxes gives much more information about methods, options and alternatives. It also covers more advanced topics.

**Text in blue draws your attention to the most important actions to perform.**

**Text in red gives warnings about potential problems and actions to avoid.**

Advanced users can refer to the Other Resources listed below for other topics not covered in this guide.

## DISCLAIMER

A lot of the information given here is based on the documentation that is supplied with the Microsoft Editors and Tools. I take no responsibility for any adverse consequences. Hopefully, as users discover more about the MSTS Editors and Tools, things will become clearer and more precise.

## OTHER RESOURCES

There are other sources of information available for more details and for other topics not covered in detail here:

- Richard Garber's **Step by Step Guide to Train Sim Activities**: it is the perfect complement to this route building guide;

- the on-line **Editors and Tools Help**: this is a useful source of help, but is not written with the first-time user in mind;

- the **Train Simulator Manual** (normally available from Windows through Start - Programs - Microsoft Games - Train Simulator): this focuses on running trains, not on designing routes, but its section on Activities in particular should give you ideas on what is possible in the area of activities;

- the **MSTS Tech Docs** (available on the first MSTS CD in the TECHDOCS folder as TECHDOCS.EXE, to be installed on your hard disk): this contains many Word files describing various relatively advanced and technical topics for designing routes and associated objects; the most useful one for the beginner is "Laying Track and Terrain.doc";

- the **Train-Sim.com Message Area**: this Forum provides of lot of help in the form of questions and answers by people like you, who have struggled (or are still struggling) to learn to use the MSTS Editors and Tools; in particular look for the MSTS FAQ (Frequently Asked Questions) area, which is kept up to date with useful information;

- the file **routhlp2.zip** (available for download from the Train-Sim.com library): this is a very useful collection of Q&As selected by Rich Garber from threads in the Train-Sim.com Route Design Forum.

- the volume **Microsoft Train Simulator: Sybex Official Strategies & Secrets**: this guide devotes 40 pages (with many figures) to building routes; most of that deals with modifying existing routes, which turns out to be far simpler than creating a brand-new route;  it does not cover important basic information for adding textures, transfers and objects to brand-new routes; for example, creating tunnel entrances is left out, despite its importance;  on the other hand, 50 pages are devoted to Activities (not counting the detailed description of most of the default MSTS Activities).

## 1. PLANNING A ROUTE

**A railway route consists of many types of components.** The following list is not meant to frighten the route builder, but to give an idea of what goes into a "complete" route, like the 6 default MSTS routes.

**A "complete" MSTS route includes:** terrain shapes and textures, tracks, yards, bridges, tunnels and other underground structures, water, stations, platforms, overhead electric wires and gantries, signals, speed limits, mileposts, roads with traffic, controlled level crossings, refueling equipment, static and dynamic trains, houses and other buildings, all sorts of visual objects from rocks and fences to junk piles and railroad equipment, from city objects to industrial complexes, and of course people and animals, as well as train and environmental sounds. In addition, a route needs at least one "activity" to make it possible to use in MSTS; other activities are desirable to add challenges and adventures for the user.

Of course, **a route can be made much simpler** than those default MSTS routes: there is no need to include everything that is available. However, a route does need to include a wide variety of features to look realistic.

Building a "complete" route from the ground up is therefore a rather large enterprise, even for a short route. It must therefore be carefully planned. In particular, **it is important to build a route in an orderly fashion**. This Guide is structured to reflect the need for such an order.

Some very **general recommendations** are useful at this point:

- **Start by building a short route** (perhaps 10 kilometers or miles) to gain experience: the "First Route" project in this Guide serves that purpose, but it is still a good idea not to design a big 100 kilometer route right after that. Try something small and entertaining, so you create a satisfactory route to run on in a relatively short time. In fact, you can start a large route by building first a small portion for later expansion: you then have a working portion of a route to enjoy (and even distribute). Also, you are likely to encounter errors that may force you back several steps, and that is much less traumatic with a small route.

- It is important to realize that **track and terrain** (mountains, valleys, etc.) **are very closely coupled in the building process**: you will normally shape the terrain at the same time as you lay track. Unless you import terrain and track layout from another source (a separate job that requires additional knowledge, software and techniques, not supplied with MSTS), you will have to coordinate the terrain shaping closely with the track laying: except on perfectly flat ground, these are inseparable.

- Remember that the process of laying tracks normally operates by attaching one track section to a previous track section. Therefore, **you will lay a first track section and build outward from it**, in one or two directions (or more directions if you split the tracks with switches). So track laying will be very systematic, progressing one small track section at a time. Adding other items will be less systematic, but should nevertheless follow a certain sequence to minimize problems.

- It is equally important to understand that **once you have laid tracks**, especially over hills and mountains, **it can be quite hard to change them**: although you can tear out existing track, it is very laborious to fill in new track in the resulting gap: the reason is that it is hard to make two tracks meet within a couple of centimeters, especially on sloping terrain.

- Because of the large memory requirements of the MSTS Route Editor, **it is absolutely essential to frequently save and back up your route, and to frequently reboot your computer**. If you don't, errors will occur that can ruin your work. It can be very hard to survive errors in the route building process, so that it is necessary to be able to go back to a recent error-free version of the route. Otherwise, many hours of labor can be wasted. More than one route builder has given up as a result of such errors. Investing some time in backing up will repay itself manifold in recovering from fatal errors.

- A rich source of ideas and techniques can be found in existing routes, whether the default MSTS routes or add-on routes: by studying them within the Route Editor you will certainly find extremely useful methods for your own route.

- As an alternative to building e new route, you may instead modify an existing route, such as one of the 6 default MSTS routes. This is perfectly possible, but there is one major complication in doing this: to change tracks in an existing route, you must first delete all "track objects" (signals, speedposts, mileposts, hazards, sound regions, level crossings and car spawners) within the "tile" where you change the tracks, and then reposition them after making those track changes. (Even so, this is a risky job, as the Route Editor has bugs that can corrupt the result.) Furthermore, if you change tracks over which an "activity" runs, that activity will have to be updated as well.

- An easier way to modify an existing route is to extend it by adding new tracks beyond its free ends, without changing existing track. In this case, there is no need to remove and replace any objects of the existing route. However, to use the added parts of the route, it is desirable to create new activities that run over those new parts.

## 1.1 Steps in designing and building a route

**Designing and building a route consists of several steps:**

- **planning the layout of the route and the terrain**: You'll have to decide on the characteristics of your route. These characteristics answer questions such as:

- What kind of route do you want?
- Will it use real terrain elevations, or will it be fictional?
- Will it consist of one or more mainline(s), dual-tracked or otherwise, yards, sidings, spurs, loops, wyes, bridges, tunnels, overpasses, grades, switchbacks?
- What kind of terrain will you make?
- Where will be the stations, towns, hills, mountains, valleys, rivers, lakes, sea?

- ♦ How much detail do you want to include, of the type that you see in the MSTS default routes?

Keep in mind that you can use the default MSTS routes to get lots of ideas and solutions - you can open them in the Route Editor and examine how they are constructed (you can even modify them to suit your desires!).

- your planning will have to take some **limitations of MSTS** into account (see section 1.3), such as the fact that MSTS does not accept routes that form a closed circuit (the form that most model railroad tracks have);

- your planning should result in a **rough drawing of your route**, so that you can start to lay track and know how to extend the route from there; of course you can also improvise as you build, but you will have to keep in mind that you may then have to expand the geographic space that you will have defined before you start laying track;

- before laying track you will need to **create the geographic space** where it will reside, namely where in the world it will be placed: you do this **with the Route Geometry Extractor (RGE)**; you will need to know roughly how large that geographic space will be and roughly what shape it will have; the RGE utility also creates a separate new folder in the Train Simulator Routes folder: this new folder is reserved for your route, so that your route will be completely independent of any existing or future add-on routes;

- MSTS and its Editors and Tools unfortunately do not offer terrain altitudes from the real world, but freeware utilities are becoming available to prepare those for MSTS;

- in the absence of real terrain data, you will be able to shape the terrain in several stages within RE, first roughly and then more finely, to some extent while laying tracks, but mainly after that;

- now you can **lay tracks**, one track piece at a time, attaching each piece to one that you have already put on the terrain; that way, the route will grow until it is complete; this is done with the Route Editor (RE); you can already at this stage make your tracks slope up or down and raise or lower the terrain locally to match its slopes;

- it is strongly advised to **test your tracks** after each significant extension, such as after adding a set of switches, and especially after adding a loop or a wye (because these cause much trouble in MSTS!);

- testing requires first **making one or more simple activities** with the Activity Editor (AE): these allow you to **drive a train over your track** with MSTS; if MSTS fails to start your route, or if it fails while you drive, you must change some tracks in your route (for that reason, don't refine your terrain until later); if you delay testing until after the completion of the entire route, you may spend endless hours trying to find errors in your track layout;

- after you have laid a stretch of tracks and tested it, it is convenient to **fill in the terrain shapes** farther away from your tracks, to form hills, mountains, valleys, etc.; this is also done in the Route Editor (RE);

- once you have laid tracks and shaped the terrain, you can start to **"decorate" the route**, using RE, with all sorts of objects, from bridges and forests to railway signals and houses; however, this usually requires "importing" objects from other MSTS routes (or even creating your own); "decorating" can take much more time than laying tracks and shaping terrain: in fact, you could easily go on forever;

- finally, you will need to **make one or more activities**, so as to be able to drive around on your route;

- if you want to make your route available to other MSTS users, you could finally **package the route** for distribution.

These different steps are the subject matter of this guide: we will go through them all in proper sequence, so you can gradually build up your route, starting from nothing.

[NEW SINCE V1.106] **Appendix G offers a convenient checklist that you can use when building a route**: it leads you through the different stages of composing a route, in an order that minimizes the risk of making time-consuming mistakes.
[NEW SINCE V2] Joining two routes into one route is possible. The two initial routes need to be "healthy" enough to accept a track database rebuild. If one route needs to be moved geographically to properly join the other, it is necessary to rename its world and tile files (see section 2.7). After joining the routes, a track database rebuild is needed.
[NEW SINCE V2] **Collaborating on building a route** is possible to a limited extent. It is however necessary for a single person to define the geographical space with tiles (using the Route Geometry Extractor). Also only one person at a time should lay tracks and roads, including track and road objects (such as signals, level crossings, etc.). On the other hand, the terrain shaping and object positioning can be done in parallel by different people working on different "tiles" of the route: this will require exchanging the appropriate tile files from the TILES and WORLD folders of the route. In any case, this requires a good understanding of the function of the different folders and files (see Appendix F for an outline).

## 1.2 Designing a route

It is very useful to have at least a **rough plan for your route**, whether it will model a real-life route or become a fictional route. One reason is that you will need to first select a set of square tiles on which your route will reside. Another reason is that you should have a good idea where you want big things to be: curves, loops, wyes, big valleys and mountains take much space, while each tile is only about 2km by 2km square (1.24mi by 1.24mi).

If you plan to model a **major real-life route**, there is a good chance that the Route Geometry Extractor (RGE) has it in its database. This only means that the RGE can guide you in selecting the correct geographical area (and "tiles") for it. It does not mean that it provides the tracks (or any help to place those tracks) or real terrain elevations. You must find that information yourself: see section 2.6.

Especially for a fictional route, **draw a route map on paper with a square grid**: show each tile (which is 2.048km x 2.048km in size) with its 16 x 16 "patches" (these are 128m x 128m in size). Such a map will form a very convenient guide when laying track, because you can refer track positions to each of those tiles and patches.

The 128m x 128m patch size is a convenient reference for judging track curvature. A circle filling one patch has a 128m diameter or 64m radius: such a radius is quite tight for curved tracks, even in a yard or in mountains, but it is usable at low speeds; an 80m radius is more reasonable for tight curves (but long passenger cars will still have trouble with such curvatures). With dynamic tracks you can go down to a 25m radius, but that is almost useless (some trains grind to a halt in too tight turns, or even bounce backward!). The tighter the turn in MSTS, the more oscillations will occur in a train, and the harder it will be to pull the train through the curve (for some reason, there is increased resistance in tight curves, especially with longer cars).

You can make a "marker" file for guidance in laying track (this will place markers on the terrain, so you can lay your track from marker to marker). But this requires that you know the coordinates of each marker: these are not so easy to generate, especially for free-style routes. I prefer the following method.

Use the drawn map as a guide to place coasts, rivers, then tracks, then roads, etc., by reference to the tile and patch boundaries. One efficient way to do this is to place single trees along the route that you plan, and then to lay tracks to join the trees (the trees can stay as long as you like: they don't prevent trains from running).

You can lay independent (unconnected) routes on the same set of tiles (under the same route name in the same route folder). In other words, you can build one route (with the RGE and RE), and then add a second route (with RE) that does not connect to the first route.

One advantage of <u>not</u> connecting routes is that this reduces the risks of closed circuits, loops and wyes freezing up MSTS. Another advantage is that you can run "computer-controlled" trains on one route in an Activity, while the player runs another train on another route, without risk of collision or blockage. A disadvantage is that, to drive on the other route, you have to start another Activity or choose a different start position in Explore Route mode.

You can also build unconnected routes on the same set of tiles, and connect them later. You may certainly build completely independent routes in the same geographical area (say both from London to Brighton), if you store them in separate route folders and give them different names: you can run only one of them in MSTS at any one time (the other will be unknown to MSTS during that time).

## 1.3 MSTS limitations

MSTS has severe limitations on the way routes can be put together. For instance, **you cannot make a closed circuit** (like a model railroad that allows running round and round forever): if you do, MSTS will freeze up (there are exceptions, but they are not properly understood). **Loops and wyes can create similar problems**. What matters is the "organization" (or "connectivity") of your route, not the length or curvature or orientation of tracks: the order in which switches occur is the only important consideration here.

The exact rules for correctly setting up loops and wyes are not yet well established, so **a trial-and-error approach is needed** to deal with them.

A few guidelines are the following (see section 3.4.4 for more details):

- **loops cause more trouble than wyes do**: I have replaced fatal loops by wyes without problem;

- **you should not make activities go through loops or wyes**; they simply will not work (but you may place a reversing point before a wye and let the player use the wye to reverse a train);

- **in Explore Route mode, wye switches leading into loops are frozen** (although a train passing the other way appears to switch them automatically!); you can't choose which way to turn.

If you plan to include loops and wyes or any other tracks that are not simply point-to-point, **first try out your layout in simplified form on bare flat terrain**. For that purpose, make a simple flat route that includes all the right and left switches, loops, and wyes, in the proper order, but with a minimum of straight and curved tracks. You can make tracks cross each other on the same level, instead of building overpasses: MSTS accepts these. Next test this route with simple activities, and fix it as needed until it works, before making the real route. To do this, follow the same procedures as you would for the final route, but ignore all track grades and terrain shaping.

[NEW SINCE V1.106] There is also a limitation on the number of objects that you can place on any one tile: **you can place at most about 1500 objects on one tile** (this number may depend on your computer). The RE shows the number of objects on a tile in the bottom right corner of the Placement tool. Here are other hints of approaching trouble: as you get near this limit, you will find that it takes longer to enter a nearly full tile when you fly around from tile to tile in the RE (the pause is due to the large number of objects needed on that tile); also saving may take noticeably more time. If you exceed this limit, very serious corruption of your route's files may occur.

TIP: To reduce the number of objects on a tile, use forests rather than many single trees, since a forest is a single object.

## 2. THE ROUTE GEOMETRY EXTRACTOR (RGE)

### 2.1 What does the RGE do and not do?

The RGE does less than what most people expect at first!

**The RGE is used to create a bare flat green surface on which you can then lay track and shape terrain. It provides a very small number of textures and objects to decorate your route.**

The train simulator world is composed of a set of connected square tiles. These tiles correspond to specific geographic coordinates on the Earth's surface. Each tile is 2.048km x 2.048km in size, a bit over 2km x 2km, regardless of your choice of parameter values, such as the Terrain Detail Scaling Factor. The tile edges are oriented North/South and East/West.

But since this surface is bare and flat, those geographic coordinates don't really matter for fictional routes: your route will look the same wherever on Earth you position it. Also, it seems that the MSTS climate is the same everywhere on Earth. Perhaps future versions of MSTS or add-on software will make use of the actual coordinates, so you might as well keep this in mind in choosing where in the world to position your route.

The RGE colors your tiles with a green texture; the altitude is near sea level (at +1m). If you don't see the green texture when opening your route with the RE, you made a mistake in the RGE: perhaps you forgot to specify a start tile.

The most important thing that **RGE** does for you is to **create a new folder in the Train Simulator's ROUTES folder**. The folder has a name that you choose, for instance "NewRoute". Within the new folder, the RGE copies the items that you'll need to make a route with tracks and terrain shaping, and several objects that you can place in your new route. To add more objects requires other steps.

**The RGE does not install any terrain shapes or tracks or objects on your terrain.**

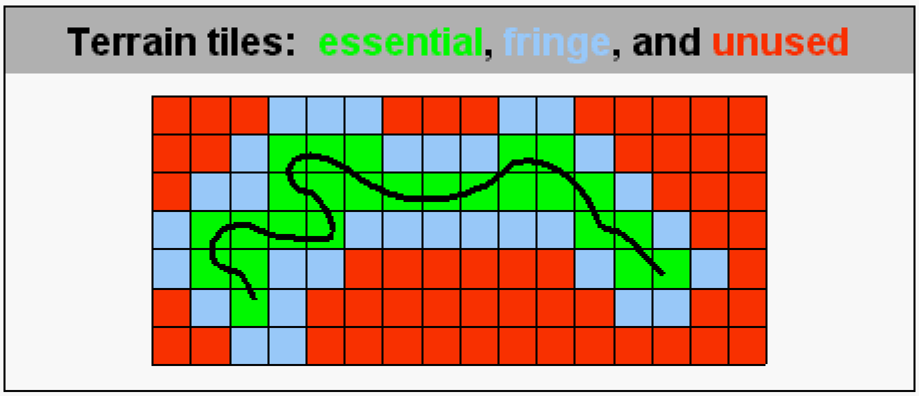
**The RGE, the Route Editor and the Activity Editor require that your monitor have a resolution of 1024 x 768, or higher.**

### 2.2 Three types of tiles

It's a good idea to **minimize the number of tiles that you will use**. This reduces the amount of space used on your hard drive and also the file size. This is especially an important factor if you want to distribute your route to others by download. Later, we'll see how to do this.

For discussion purposes, there are three types of tiles: the ones you really need (I call them "essential tiles"), nearby tiles that you may need ("fringe tiles"), and tiles that you certainly don't need ("unused tiles").

In the next illustration, the curved black line represents the intended route. The **essential tiles** (shown in green) are those on which you will lay tracks. The **fringe tiles** (shown in blue) are the tiles nearest to the essential tiles. You may need these if you miscalculate and your track spills over from an essential tile onto a fringe tile, or for scenery if your track gets close to the edge of a fringe tile; they may also be used temporarily to shape scenery at the edge of essential tiles. The **unused tiles** (shown in red) are all the other tiles further away from your route.




**In RGE, select only the essential tiles and the fringe tiles, avoiding unused tiles if you can.** After finishing your route, you can remove the fringe tiles that were only used to help shape your terrain on essential tiles. If you accidentally omit a needed tile, you can later still add it (with RGE).

## A. Building "First Route": preparing the geographic tiles on which the route will lay

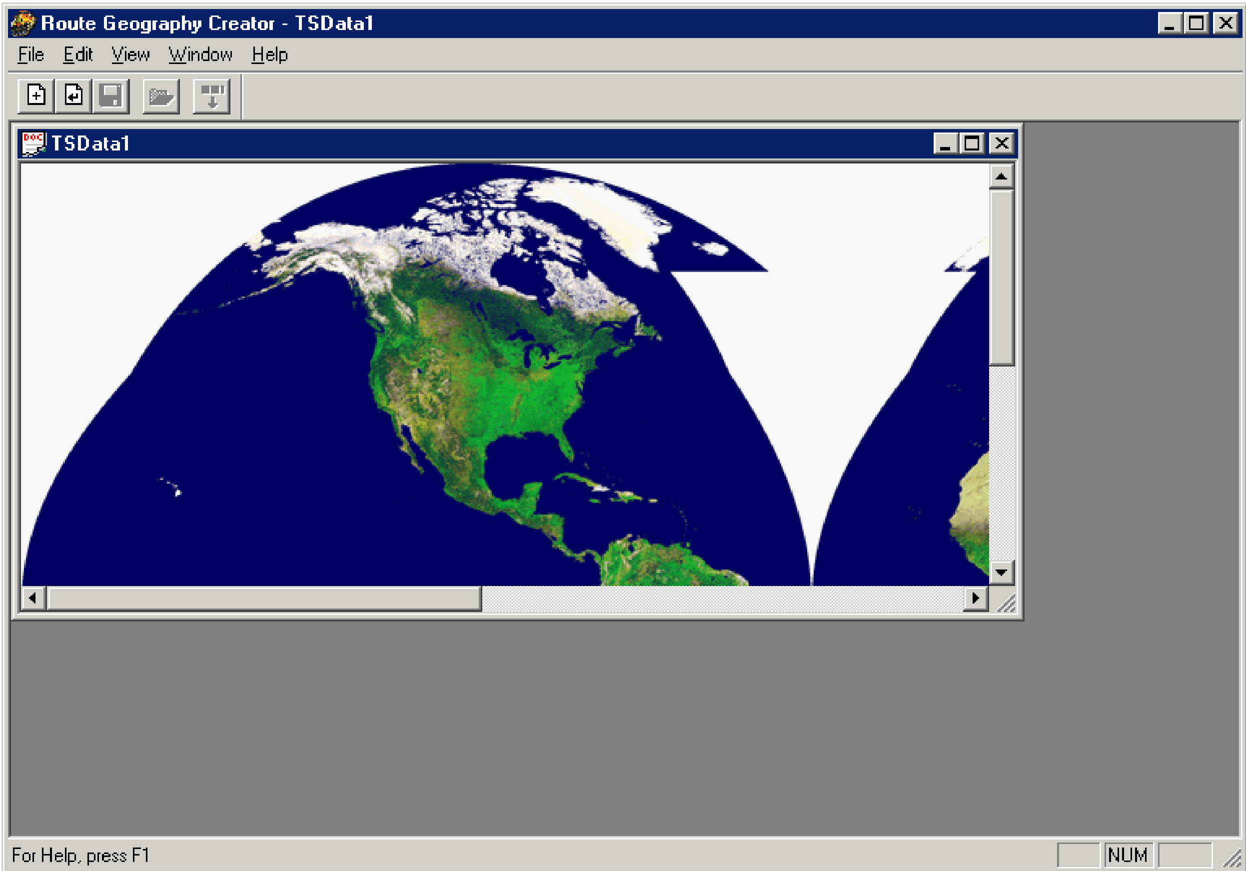
We will make a simple, short first route heading North from London. For this practice route, we'll prepare 3 x 5 "tiles", representing about 6km from East to West and 10km from South to North.

This is done with the MSTS utility called **Route Geometry Extractor**. For more details, see section 2.3.

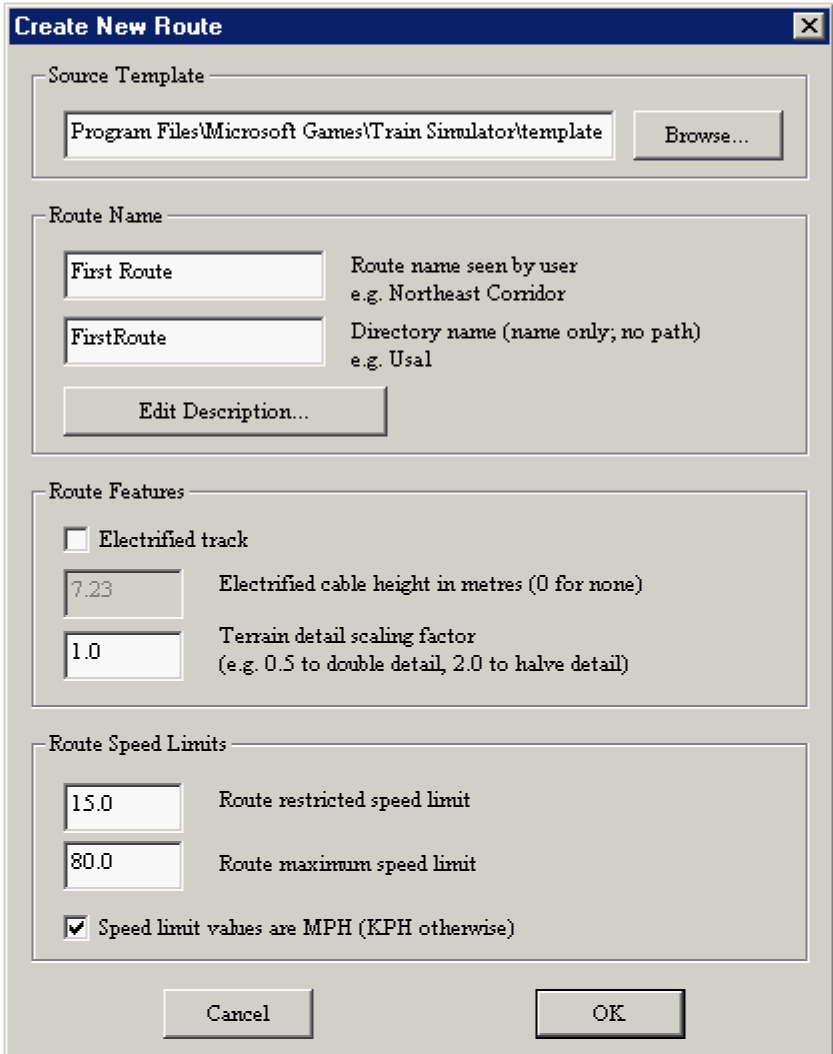
**Do the following:**

- start the Route Geometry Extractor through the Windows Start button: select **Start - Programs - Microsoft Games - Train Simulator - Train Simulator Editors and Tools**. You'll see the Editors and Tools sign-on screen. (For convenience you may want to make a shortcut for the Editors and Tools on your desktop.)

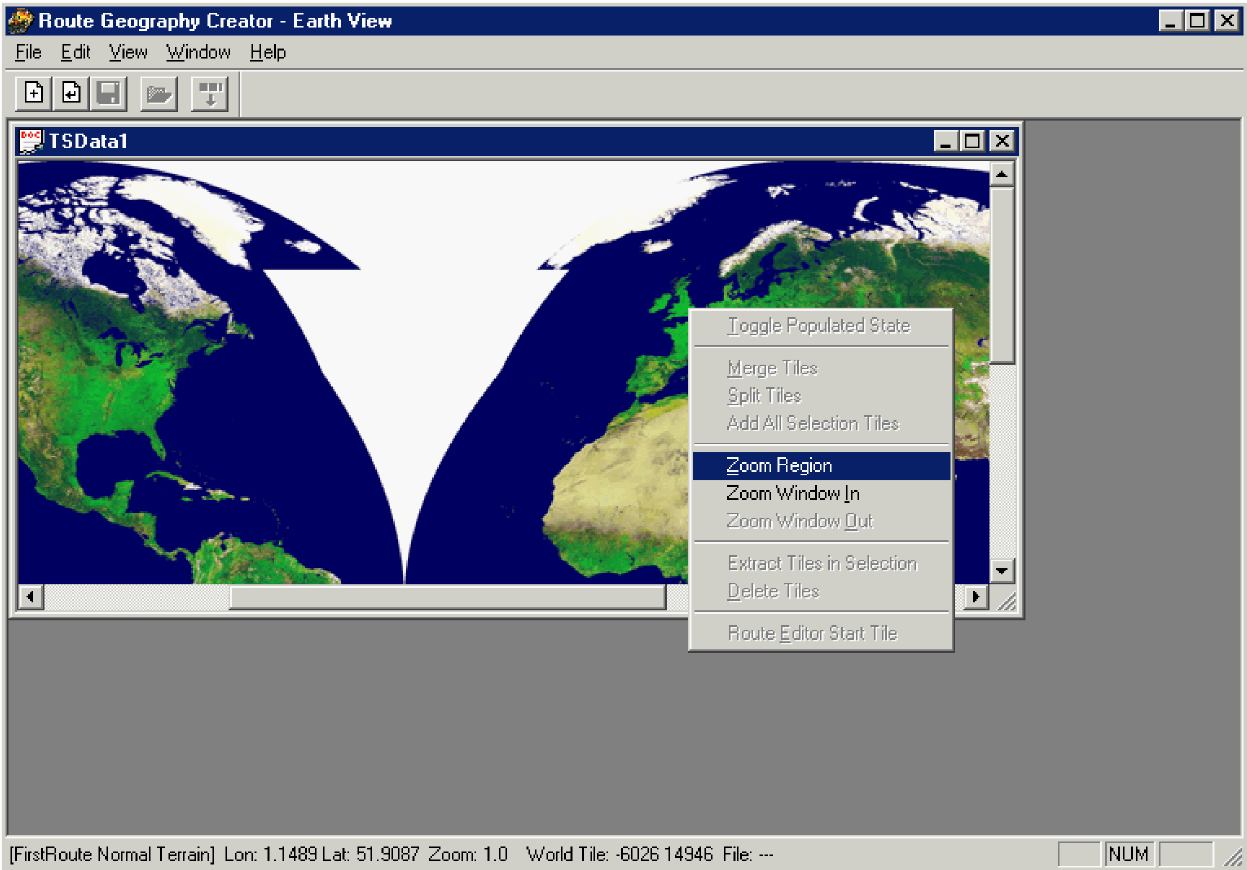

- select **Route Geometry Extractor**:  this opens a new window with a world map:



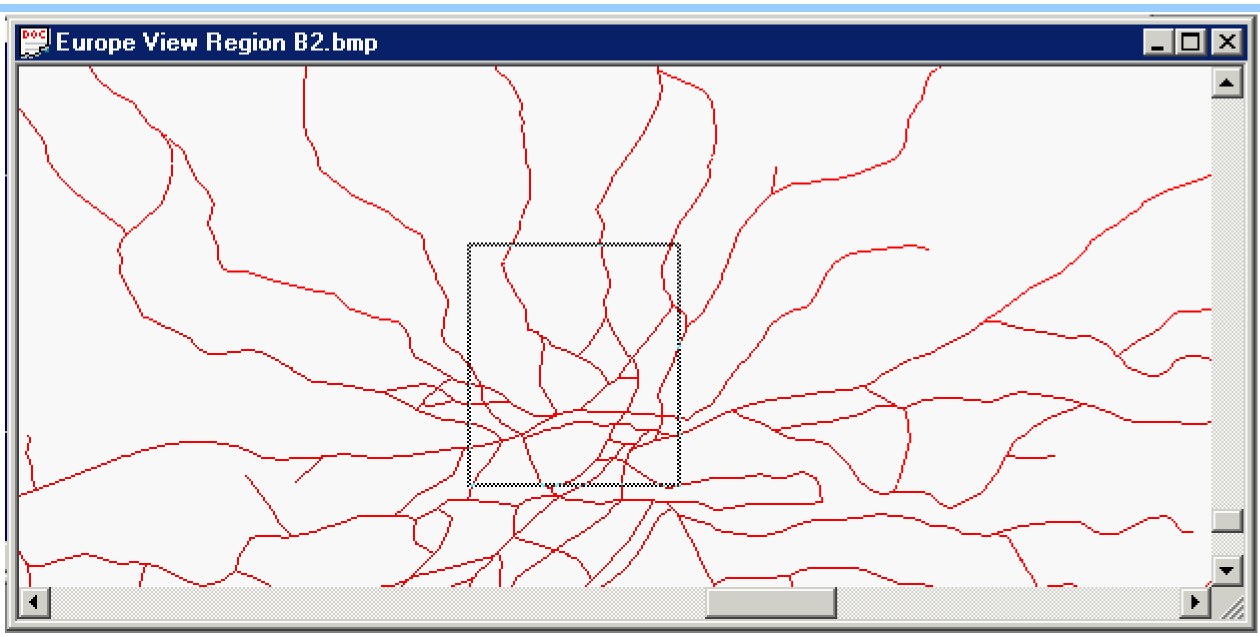

- click **File**, then click **New Route**: this opens a window entitled Create New Route;
- under Route Name, enter "First Route" for the Route name seen by the user, and "FirstRoute" for the Directory (folder) name (don't use the quotation marks ""; for the Directory name it is recommended not to use spaces; also don't use a path);
- leave everything else as given by default, resulting in the following window (the path to the source template will be different if you installed Train Simulator in a different folder than the default choice; in any case don't change what you see there):



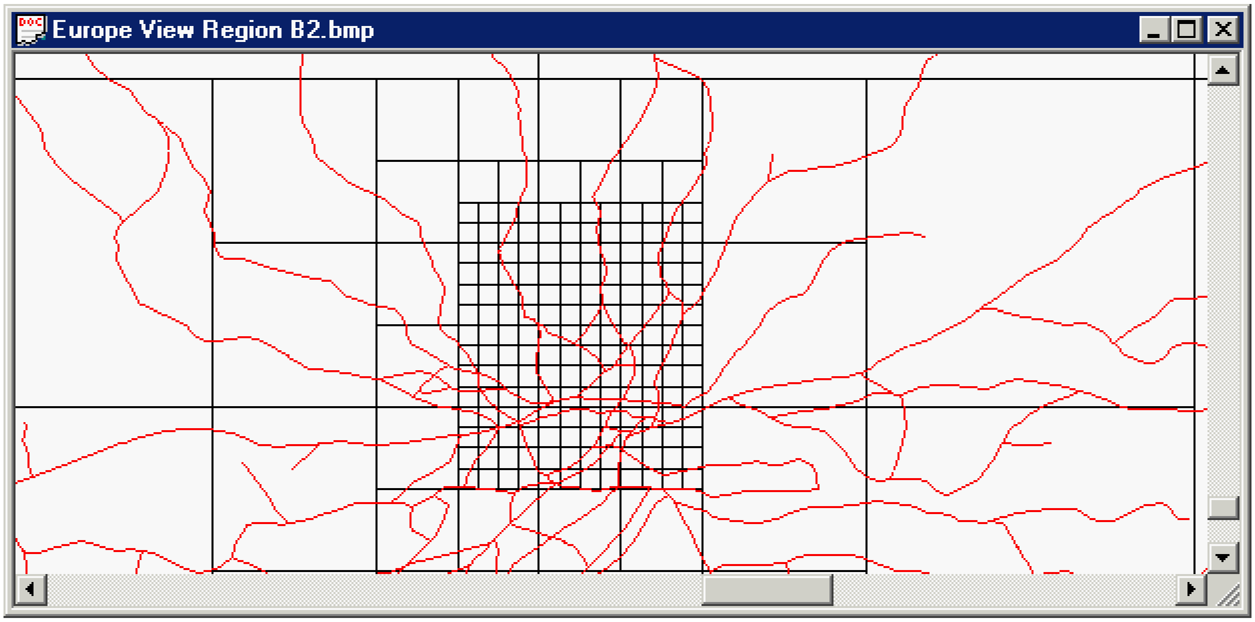

- click **OK**:  after a while, a new window tells you "New route created" and lists its folder (directory);
- click **OK**;
- click **File**, then click **Select Route**;
- in the Route list box, click the route directory (folder) name that you just created (FirstRoute), then click **OK**;
- click **File**, then click **New Quad-Tree**:  [FirstRoute Normal Terrain] appears at bottom left;
- now you must point out where the London area is:  shift the map to the right so it shows the UK and right-click near London;  this opens a pop-up menu:




- select **Zoom region**: a map of Europe appears, with major railway lines shown in red; you may maximize the map by clicking on the little square at its top right corner;
- do this once more: shift the map, right-click near London, and select **Zoom region**; a map of the UK appears;
- repeat this a bit differently now: shift the map, right-click near London, and select **Zoom window in**; a larger map of the UK appears; (if you are lost, notice that the coordinates of the mouse pointer are shown at bottom, so you can tell where you are pointing; read off the coordinates of London on an earlier map, accessible through the Window menu option, by pointing at London: these coordinates are about Lon = 0, Lat = 51.5; then shift the latest map toward those coordinates);
- repeat this again: shift the map, right-click near London, and select **Zoom window in**; a similar map of the UK appears, but showing only railway lines; at this zoom level, the map spans about 1° from top to bottom, which is about 111 km or 69 miles;
- zoom in once more the same way: right-click near London, and select **Zoom window in**; the map spans about 0.5° from top to bottom, which is about 56 km or 35 miles;
- focus on the region where you want to lay tracks; it looks something like this (without the gray box):

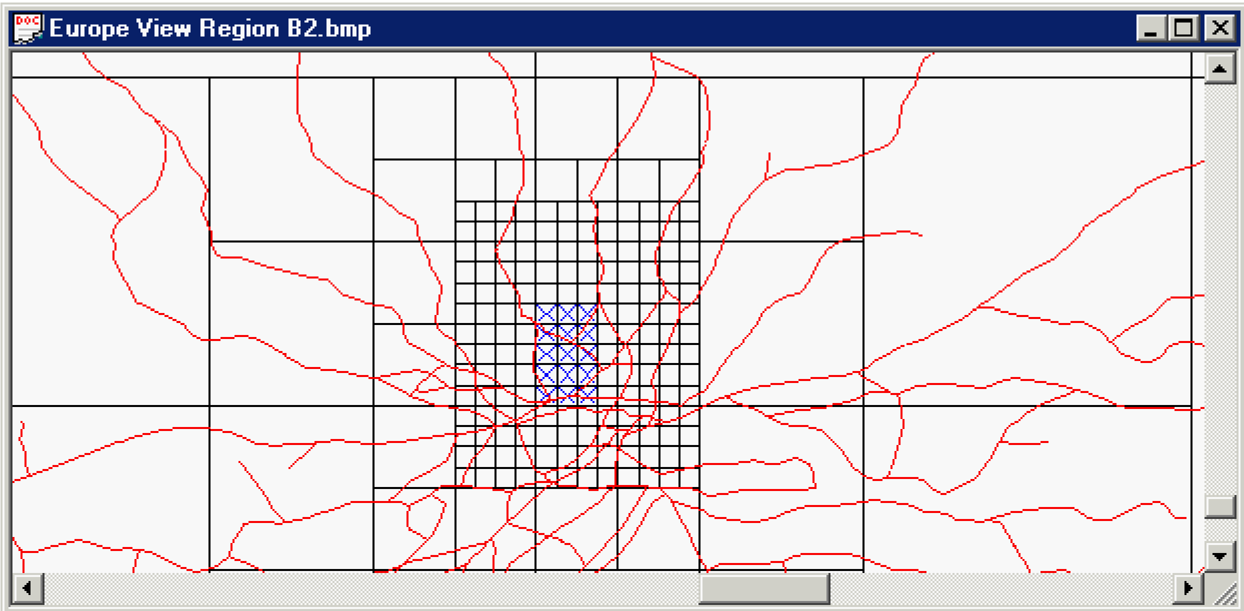



- drag a rectangle around the section of the map you want the route to occupy (drag from top left to bottom right): that is the gray box shown in the screenshot above;
- right-click inside the selected area to open the pop-up menu;
- click **Add All Selection Tiles**: the map will be covered by a grid (shown in the next screenshot): the smallest squares are the tiles with which you will be working. These tiles are about 2km by 2km in size;




- for our First Route you will have to select 3 x 5 tiles among the smallest squares; since the Route Geometry Extractor will give you neither those real-life railway lines shown in red, nor any terrain elevation (no hills, etc.), you can place your route anywhere you wish!

- if you think that your route will not fit within the shown smallest tiles, click **File**, then **New Quad-Tree**, then **No**;  next draw a new gray box and click **Add All Selection Tiles**, until you are satisfied;
- select the 3 x 5 tiles that you want to use by dragging a (slightly enlarged) rectangle around them;  it may be tricky to get just the right set of tiles;
- right-click inside the selected region, then click **Toggle Populated State**:  this marks the selected tiles with blue crosses, as shown in the next screenshot;




- if you have not selected the desired set of tiles, repeat the last step to untoggle them, then try again;  you may also toggle individual tiles on or off:  right-click a tile, then click **Toggle Populated State**;
- click **Edit**, then click **Minimize quad-tree**:  to the question "This will remove empty quad-tree nodes. Continue?", answer **Yes**;  this will reduce the number of grid lines, leaving your selected tiles untouched;  if this fails, delete the grid (click **File**, then **New Quad-Tree**, then **No**), and try again generating the grid and selecting your tiles (I find the RGE somewhat unpredictable at this stage);
- click **File**, then click **Save Quad-tree**;
- click **Edit**, then click **Generate Flagged Tiles**:  you should see "Number of tiles to generate: 15. Number of tiles flagged: 15. Generate these tiles?";  if true, click **Yes**;
- at bottom left you should see "Creating tile: x/15", where x increases from 1 to 15 as the tiles are being generated for our First Route;
- **IMPORTANT**:  right-click the middle tile in the bottom row of blue-marked tiles to show where we will start laying track (it will position the camera of the Route Editor there);  this opens the pop-up menu:  click **Route Editor Start Tile**, then click **OK**;
- click **File**, then click **Exit**:
- **IMPORTANT**:  click **X**, then **Yes**, to exit from Editors and Tools:  if you don't exit from Editors and Tools, but start up Route Editor immediately, our First Route will not show up yet!

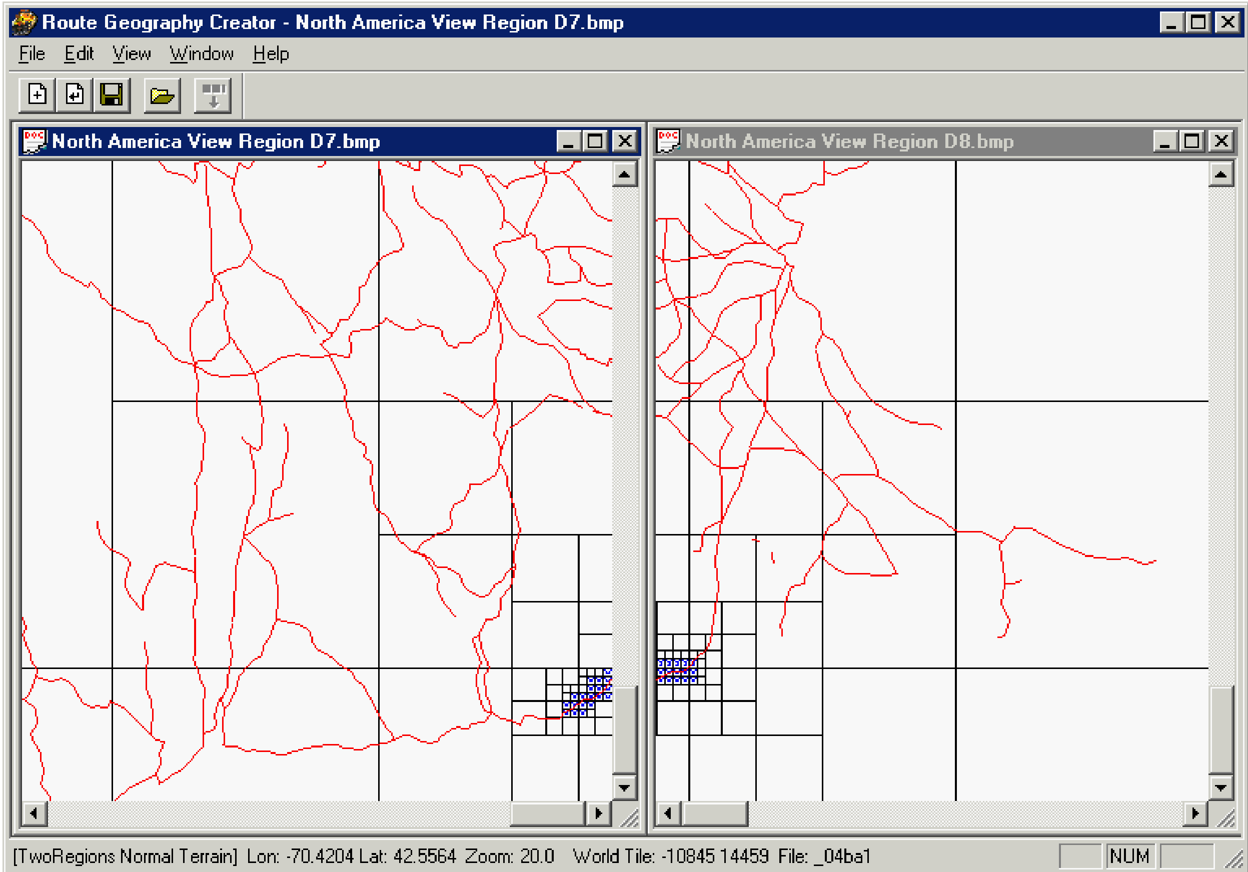
You have now created the geographical space in which our First Route will be laid, as well as a new folder that is reserved for this route alone.

I recommend that you **make a backup of this folder**, so that you could come back to it in case of difficulties with track laying, as follows.

**To back up your route, do this:**

- using Windows Explorer or My Computer, copy the entire folder called "FirstRoute" into some other folder, preferably outside the Train Simulator folder;  for example, you may have or could create a folder c:\backup;
- rename that backed-up folder to distinguish what version it is (you will back up later versions as you build the route, so you need to identify each version):  for example, you might rename this backup folder to "FirstRoute (fresh from RGE)" or "FirstRoute - todaysdate_time - fresh from RGE".

NOTE - **merging multiple regions**.  It may happen that your route spans more than one of the regions defined in the RGE.  For instance, if you make a route from New York to Boston, it will cross from one region to the next.  This is easily done in the RGE.  First you create the tiles that exist in one region, just as described above until and including the step that tells you to **Toggle Populated State**.  Then you go back to an earlier map that shows the missing region (or regions), and zoom into the next region, likewise creating tiles in that region.  Repeat this again if you need a third region, etc.  Now you can click **Edit**, then click **Minimize quad-tree**, and continue as before.  This is illustrated in the figure below, with just a few tiles selected near the region boundary.




Next, you will start up the Route Editor, and load the First Route that you created with the RGE.

**To continue building our First Route, jump to the next blue box.**

## 2.3 Using RGE

**To create a new route**, **follow the instructions in section A (the blue box that precedes this section).** Of course, use the route name that you wish, and select the region and tiles that your route will need. Also, you may make several non-default optional choices, discussed next.

There are several options that you can set in RGE (you can change their settings later by passing your route through the RGE again, except for the route's Directory name). These include:

- **Route name and Directory name**: for simplicity you can use the same name for both, but it is recommended that the Directory name include no spaces and be short; for example you could choose the Route name "Forest Valley" and the Directory name "ForestValley" (also without quotes); in this guide we will use the generic name NewRoute for both the route name and the directory name (the project in this guide uses the Route name "First Route" and the Directory name "FirstRoute");

- you can declare the track electrified and set the height of electrified cable;

- one option is of some importance for shaping terrain: **select the Terrain Detail Scaling Factor** to control how finely shaped you want your terrain to be; for flat terrain or smooth hills use 1.0 or 2.0; for sharp terrain use small values (I use 0.0625 for mountain tracks); the sharper the terrain, the larger will become the resulting file sizes;

- you can set two speed limits for your route:
  - the **Route maximum speed limit** applies everywhere on your track by default (if in Train Simulator you enable the Derailments option, the trains will derail when they go faster than this speed limit);
  - the **Route restricted speed limit** applies to specific parts of the route (such as where track repairs are in progress; this limit is used in Activities that impose a temporary speed limit);

[NEW SINCE V1.106] Another useful option is found by selecting **Edit**, and then **Set Height Offset**. This allows you to preset the terrain height to an altitude of your choice (in meters); instead of starting with terrain at an altitude of 1m, the terrain will initially be at the altitude which you type in here.
[NEW SINCE V2] (**WARNING: You must set the Height Offset BEFORE starting to Generate Flagged Tiles. You can't change this offset later.**)
[NEW SINCE V1.106] You could choose the minimum altitude relevant to your route and then shape hills and mountains up from there in the RE ("bottom-up"), or you could choose the altitude of the highest hill or mountain in your route, and then shape valleys by digging down from there ("top-down"): see section 3.5 for more details.

[NEW SINCE V1.106] Other options are reached by selecting **Edit**, then **Route Values**:

- Route name;
- Route description;
- whether the route will be electrified, together with cable height;
- whether speed limits are to be shown in miles per hour or kilometers per hour.

After selecting your tiles, **choose a Route Editor Start Tile!** This tile is where the RE camera will initially be positioned each time you open your route in RE (this choice can be changed later by loading your route again into RGE). If you don't choose a start tile, you will only see white space when you open the Route Editor later, instead of green terrain. Choose any essential tile as a start tile (avoid a fringe tile that may be deleted later), but choose a convenient one, centrally located so you can move your camera easily anywhere along your route. You can lay your first track on any tile that you wish, not only on the start tile.

You can **use RGE later at any time to delete fringe and/or unused tiles**; you may also use RGE at any time to add tiles if you need to: see next. But planning ahead properly will minimize those extra steps.

**After finishing with RGE, close RGE <u>and</u> also close the Editors and Tools, before starting the Editors and Tools again in order to launch the Route Editor.** Otherwise RE will not find your route!

If you want **to make a corresponding test route** (as I suggest in section 1.3), produce a similar set of tiles with RGE (they may be in the same location on Earth, and may use fewer tiles): repeat the above steps in RGE, but produce a route with **different names** (both the Route name and the Directory name must be different from any existing route).

NOTE: If you find in the Route Editor that a part of your tiles don't show up (they are white), it appears to be sufficient to choose a Route Editor Start Tile in the part that does not show up. A similar problem is reported while running MSTS and crossing the lines between large sections of the quadtree: no proper solution is known, but, when this happens, it seems to help to save an activity (press F2 in MSTS when you no longer see the terrain) and restart it (instead of selecting Drive a Train in MSTS, select Start a Saved Activity).

## 2.4 Deleting and adding tiles

You can **delete or add tiles in your route through the RGE**. This can be done at any time during your route building. For example, you could reduce the 3x5 tiles of First Route to 2x4, by keeping only the tiles that are useful, with the result illustrated in section 2.7.

You need to first determine which tiles you want to delete or add. To do this, you could use the Route Editor to navigate around your tiles, so you can hand-draw a map showing which tiles you want to keep and which you will delete.

Note that before deleting any tile, you should verify that it does not include objects that you may have added. To do this, move the Route Editor camera around the edge of the tiles to keep, viewing in wire mode - press W - and in terrain texturing mode - press F7 - to see the blue tile edges. For example, forests can easily spill over onto neighboring tiles. That is not a serious problem, but if you do not avoid spillovers, you will be left with more files than necessary in the WORLD subfolder of your route.

Amazingly, if you accidentally delete a tile upon which you have installed tracks (an "essential" tile), MSTS still works! You will drive on your tracks in a white space with objects (like trees and buildings) floating around you.

WARNING: When you delete tiles from the RGE map, it is easy to lose sight of where you are! Keep a good record of which tiles are left (for example by crossing out tiles on a hand drawing), so you know which ones are still to be deleted!

**To delete tiles from a route, do the following** (WARNING: **the RGE Help instructions are incomplete**):

- **make sure you have backed up your route** (make a full copy of your route's folder outside the Train Simulator folder);

- start RGE;

- click **File**, then click **Select Route**;

- click **File**, then click **Load Quad-Tree**;

- zoom into your route tiles;

- you may delete tiles grouped as rectangles, all at once - do so now as follows:

    - drag a rectangle around the group of tiles you want to delete from the route;

    - right-click inside the rectangle;

    - click **Delete Tiles**;

- you may still need to delete single tiles - do so now as follows:

    - right-click on a tile you want to remove;

    - click **Delete Tiles**;

- when all tiles have been deleted, click **Edit**, then click **Minimize Quad-tree**, and answer **Yes**;

- click **File**, then click **Save Quad-tree**;

- right-click the blue tile which you want to become the start tile;

- click **Route Editor Start Tile**, then **OK**;

- click **File**, then **Exit**, to leave the RGE;

- exit the Editors and Tools.

**To add tiles to a route, do the following:**

- **make sure you have backed up your route** (make a full copy of your route's folder outside the Train Simulator folder);

- start RGE;

- click **File**, then click **Select Route**;

- click **File**, then click **Load Quad-Tree**;

- zoom into your route tiles;

- drag a rectangle outside the set of already existing tiles (marked with a blue cross) to define a region within which you want to add one or more tiles;

- right-click inside the rectangle;

- click **Add All Selection Tiles**: this creates a new grid of empty tiles;

- you may add tiles in exactly the same way you created new tiles originally, as follows:

  - select a rectangle of tiles by dragging a rectangle around them, or select a single tile;

  - right-click inside the rectangle or individual tile;

  - click **Toggle Populated State** (make sure you don't cancel the blue marks of the older tiles);

  - repeat these steps to add more tiles, if you wish;

- click **Edit**, then click **Generate Flagged Tiles**, and answer **Yes**;

- click **Edit**, then click **Minimize Quad-tree**, and answer **Yes**;

- click **File**, then click **Save Quad-tree**;

- you may change the Route Editor Start Tile, if you wish: to do that, right-click the desired blue tile, then click **Route Editor Start Tile**, then **OK**;

- click **File**, then **Exit**, to leave the RGE;

- exit the Editors and Tools.

NOTE - **merging multiple regions**. It may happen that your route spans more than one of the regions defined in the RGE. How to deal with this situation is shown with a specific example at the end of Section A.

## 2.5 Changing the Route Editor Start Tile

The Route Editor always starts up with the camera placed near the center of the tile that you defined as the Route Editor Start Tile. When building long routes, you will soon find that you are working much of the time far away from that tile. So, each time you start the Route Editor, you have to either jump to a set of new coordinates or a new tile number (which requires that you know those coordinates or tile number, see section 3.1.4), or you have to move the camera a long way to your work location along the track.

Instead, you can change the Route Editor Start Tile at any time, so the camera will be initially positioned close to your current work location. This is done by loading your route into the RGE: do the same steps as if you were deleting tiles (see section 2.4), but skip those steps that deal with deleting tiles, minimizing and saving the quad-tree; in particular, select a tile as the new Route Editor Start Tile.

[NEW SINCE V1.106] Alternatively, you can do this without using the RGE, by editing the route's *.trk file with Wordpad. This file contains a line that may read, for example,

```
RouteStart ( -6079 14928 136 -136 )
```

The first two numbers (`-6079 14928`) label the start tile and can be changed (the second two numbers give the camera start position within the start tile, as x and z in meters relative to the tile center; they must be within the interval from -1024 to +1024). To find the label for the new start tile, you should move to the desired tile in the Route Editor, and then read off the "tile x" and "tile z" values in the Camera window (see section 3.10.9 for more details), or you can get such values from the RGE (see section 2.4).

[NEW SINCE V1.106]

## 2.6 Importing real terrain elevations (DEM)

After RGE has created tiles for your route, it is possible to replace those tiles with real-life terrain data that give actual terrain elevations (such data are usually called "Digital Elevation Model", or DEM): this is done after exiting RGE and before starting the Route Editor (RE), by using other third party software not provided with MSTS.

The process for importing real terrain data starts by obtaining the DEM data. An excellent source for the USA is MapMart (http://www.mapmart.com/). DEM data can be selected from an interactive map of the USA and downloaded for free (MapMart has a limit of 10 free DEMs per download session, however one can perform multiple download sessions per day). Data are available for the entire USA at 30 meter resolution which is fully acceptable for use with MSTS, and some areas at 10 meter resolution. Another source for US data is the GIS Data Depot (http://www.geocomm.com/): most of the data files are available for free download, or on CD for a fee. DEM data are also available for other countries at the GIS Data Depot for free (at http://www.gisdatadepot.com/catalog/index.html), but the resolution in many cases may not be sufficient for route design. Higher resolution data for other countries is generally only available for a substantial fee, however sometimes may be found for free at universities or other educational institutions.

Once the DEM data is obtained it must be converted into actual MSTS terrain on the tiles for your route. There are several third party programs available that perform this task, both free and payware.

One freely available program is DEMEX, which can be downloaded from http://www.dem-ex.com/. A detailed tutorial is also available from the DEMEX site which walks the user through the entire terrain generation process from downloading the DEM data to viewing the final product. All the basic required terrain generation features are available for free in DEMEX, however paying a registration fee for the program will unlock additional features, including the creation of Distant Mountains from DEM data or existing route terrain.

Another freeware program is TSTerraform which is available from the file library at http://www.train-sim.com/. TSTerraform may be used to import small DEM areas into MSTS, or to trace contour lines from scanned topographic maps.

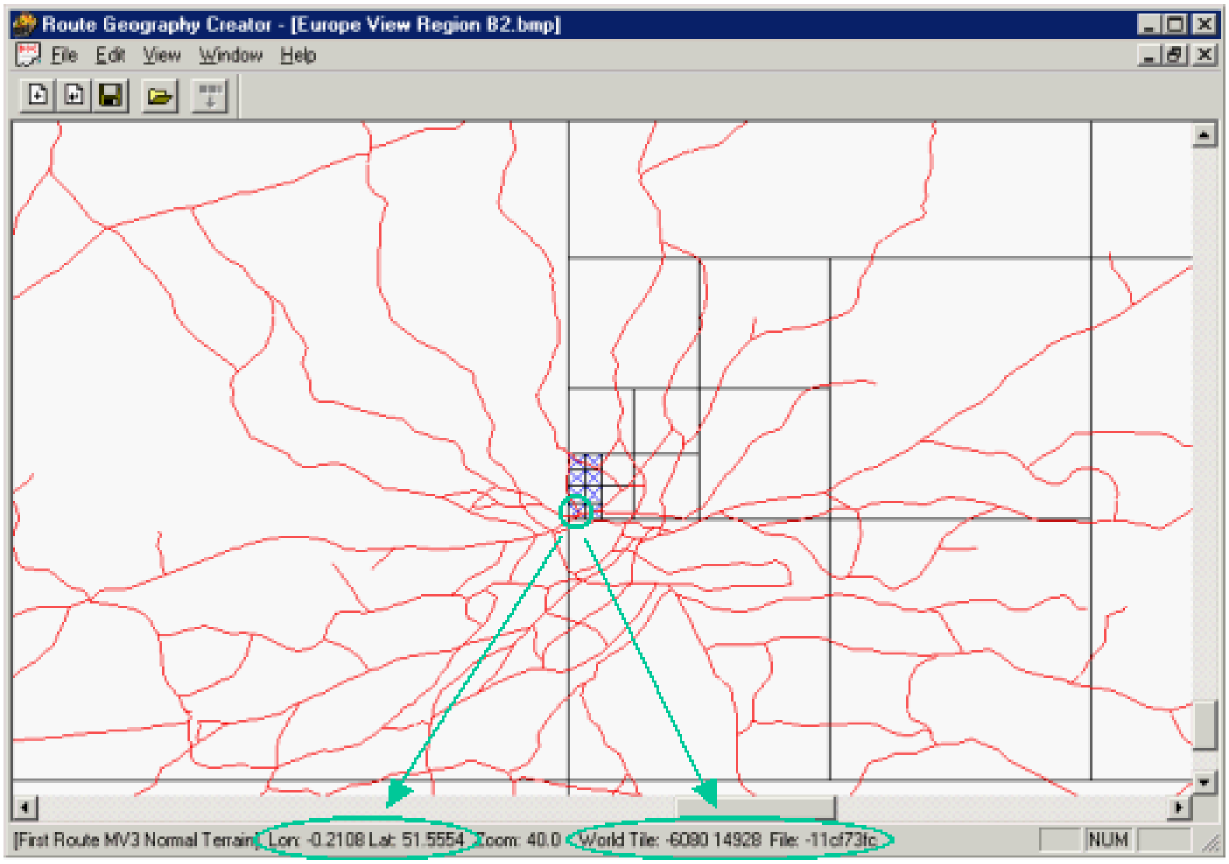
A payware toolkit called TsTools is also available from http://www.ts-tools.com/. It allows importing DEM data as well as converting contour lines from topographical maps into data usable by MSTS. Before you buy, you may consult its help file as http://www.train-sim.com/dcforum/User_files/3c1696a40efece0b.zip.

You are referred to the Train-Sim.com Message Forum for many discussions and tips concerning the use of DEM data.

[NEW SINCE V2]

## 2.7 Tile numbering

The following picture shows some useful information given by RGE about your route and tiles.




If you point at a particular position with the mouse, the RGE will tell you its **longitude and latitude** (in decimal form, where 51.5 = 51° 30', as also used in the Camera window - see section 3.1.4).

If you point at a tile, the RGE will show several numbers (marked in green outline in the picture) that it uses to label that tile:

First, the numbers -6080 and 14928 are the **"tile x" and "tile z"** values that show up in the Camera window in the Route Editor when the camera is within that tile. They are also used to label that tile's **world tile files** (these files reside in the route's WORLD folder and contain details about all objects placed in that tile): in this example `-006080+014928.w` and `-006080+014928.ws` (the second of these contains only sounds). These tile numbers vary simply by adding or subtracting 1 as you go from one tile to the next.

Second, the file label -11cf73cf appears in the name of **tile files** (these reside in the route's TILES folder and contain the terrain shape and textures for that tile): in this example `-11cf73cf.t` and other similar file names. This tile label varies in a relatively complicated way from one tile to the next. It is easier to read these labels from the RGE (as discussed here) than to try to calculate them. (I am grateful to Charles V. Beckers, Jr., for helpful discussions about this.)

It may be useful to record these tile labels for each important tile in your route on a map that shows all your tiles:  you can then enter the "tile x" and "tile z" values in the Camera window to jump to that tile (see section 3.1.4), or use them for the Route Editor Start Tile (see section 2.5); you can then also identify world tile files for manual editing of objects placed within a given tile (see section 3.10.9);  and you can identify tile files to recover older terrain shapes if you change your mind or damage the terrain (see section 3.5.6).

# 3. THE ROUTE EDITOR (RE)

## 3.1 Using RE

### 3.1.1 ORDER OF OPERATIONS

**I recommend the following order of operations** (details are given later). Build a route in two major stages: first lay track and shape the terrain; then add roads, objects, etc.

In the first stage, you perform the following steps:

- define tile and patch properties;
- place markers to guide later track laying;
- lay tracks, while adjusting local terrain only;
- test by running a train using simple activities;
- shape all terrain;
- run a final test: do this only after all preceding steps are finished for the entire route.

You may complete each of the above steps first for the whole route, or work piecewise as the route progresses (for instance, you could define the tile and patch properties for the entire route before moving to the next step, or first work on one tile or region alone, laying track and shaping terrain there before moving to the next tile or region).

In the second stage, you perform the following steps, roughly in the following order:

- add interactive track objects (such as signals, mileposts, etc.);
- add roads, road traffic and level crossings (adding roads earlier can cause severe problems);
- change terrain textures;
- add objects (such as tunnel entrances, stations, bridges, buildings, forests, sounds, etc.);
- electrify the route and add gantries;
- make activities.

In this second stage, it is preferable to first place those objects that interact with the tracks, before working on the other aspects.

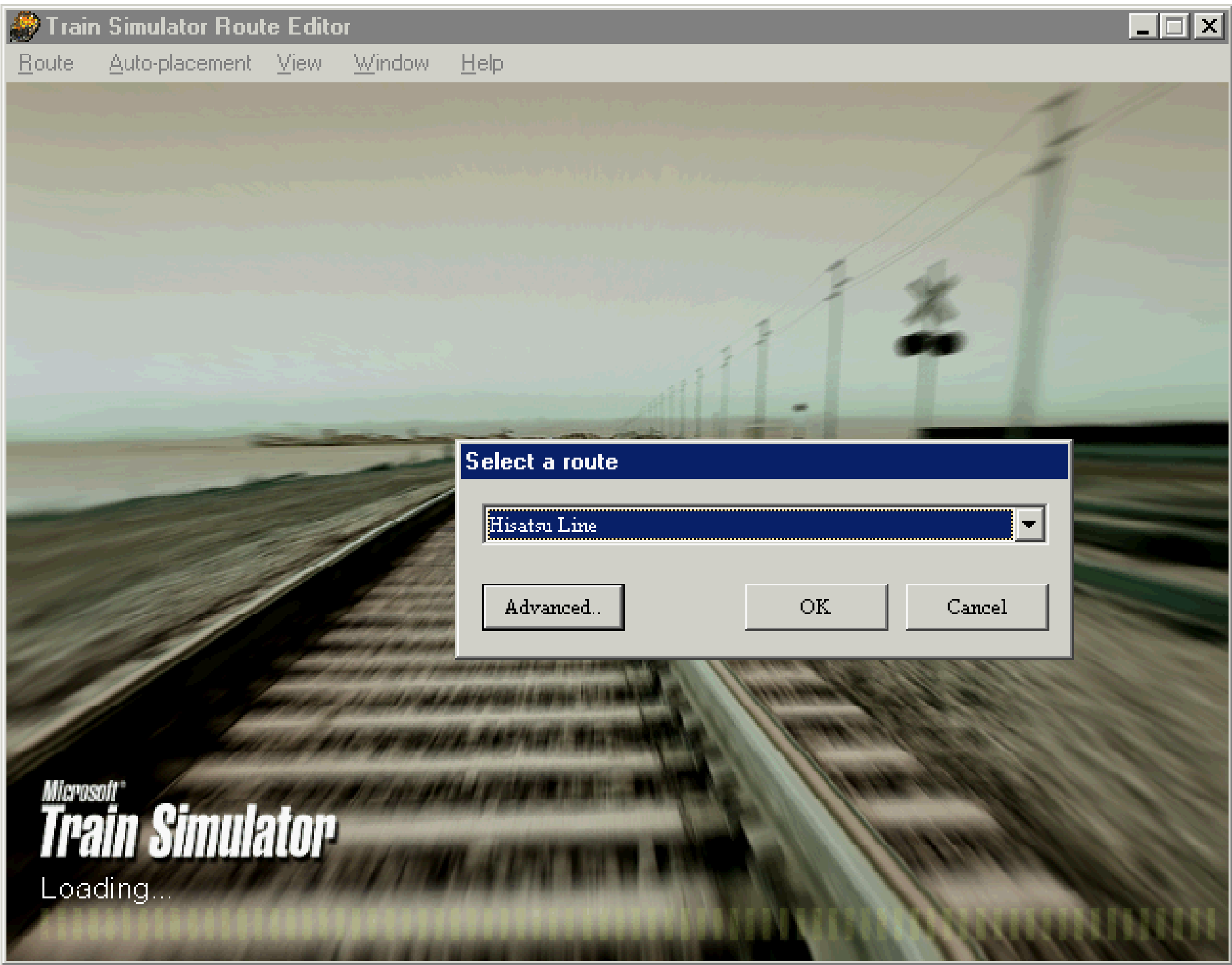
NOTE: The abovementioned order is not compulsory, but it reduces the risks of making fatal mistakes. An example of a different order is the change of an existing completed route (such as one of the default MSTS routes): you can add or remove new tiles, add or change terrain shapes, tracks, etc. in an already completed route.

## B. Building "First Route": starting the Route Editor

After you have created the geographical space in which our First Route will be laid (as well as a new folder for this route), you must start up the **Route Editor**, which allows you to place tracks, shape terrain, etc. For more details, see section 3.1.2.

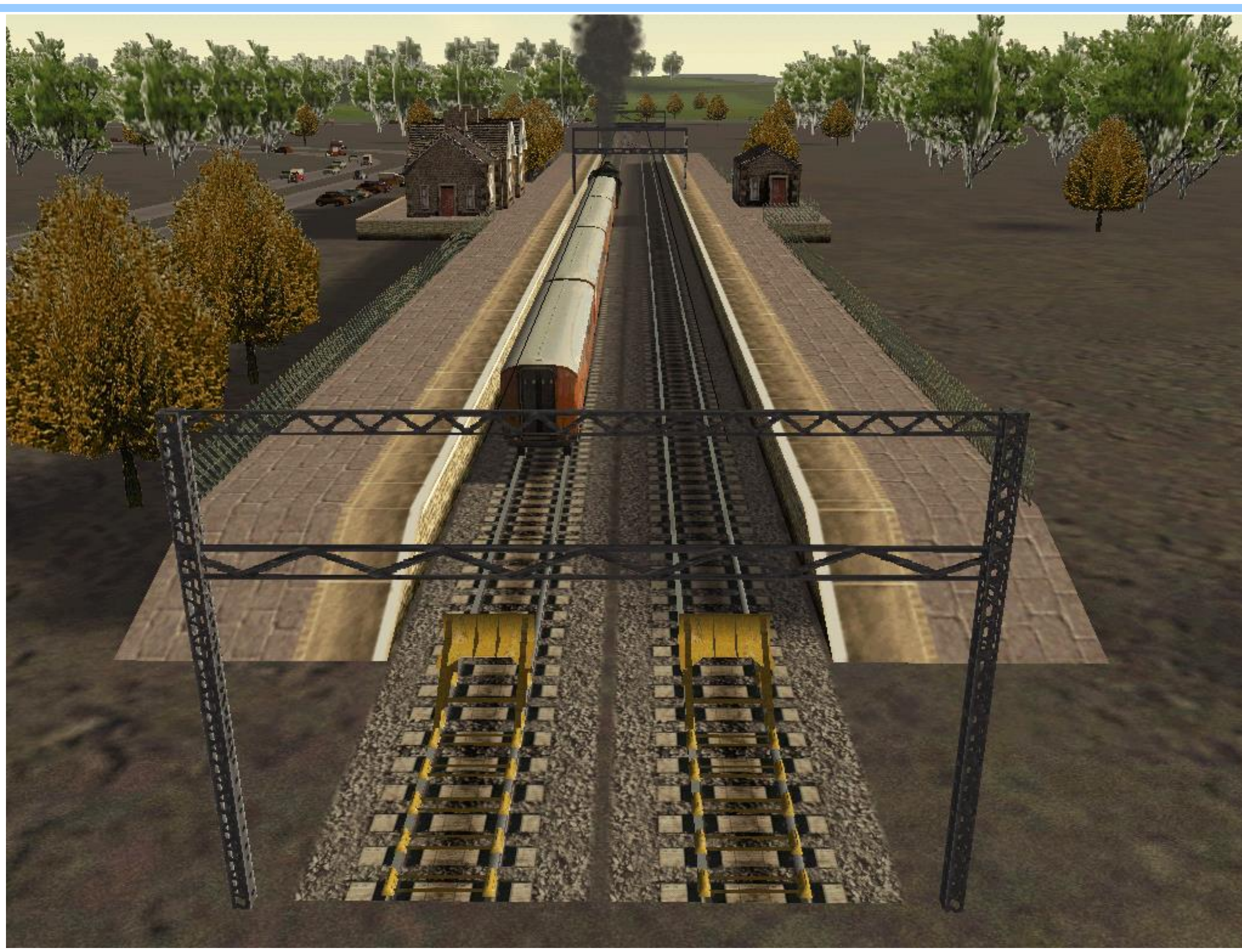
### Do the following:

- start the Route Editor through the Windows Start button: select **Start - Programs - Microsoft Games - Train Simulator - Train Simulator Editors and Tools**;
- select **Route Editor**: you should see a window with a picture of tracks, and a small window called "Select a route", as shown below: using the drop-down box, find and select your route (First Route), then click **OK**.




You should soon see an absolutely flat, bare, green terrain, with clouds blowing overhead, as shown in the next figure: this means that you have successfully created the geographical space (defined by tiles) for First Route, and loaded it into Route Editor. Certainly the rewards for your efforts may seem meager so far: just flat, bare, green terrain!


You will gradually transform this bare piece of land into something like the "Central London" railway station shown in the next picture (this view is taken from my final version of First Route, included with this guide, as are many other "finished" views in this guide). However, we will lay all the tracks and shape all the terrain before adding platforms, gantries, roads, trees, etc.



If you could not find First Route, or if it did not give you the flat, bare, green terrain, you must have made an error in its creation.

**One common mistake is to forget to define a Route Editor Start Tile. If you've made this mistake, it is easy to fix, using option 2 offered below.**

You have two options to fix a bad route:

Option 1. You can easily delete the First Route, if you know how to manipulate folders, and then start over from the very beginning, as follows:

- first quit Route Editor and Editors and Tools;
- find the folder called FirstRoute in the ROUTES folder of Train Simulator (using Windows Explorer or My Computer);
- delete the entire FirstRoute folder (and nothing else);
- now start again creating the First Route with the Route Geometry Extractor (see the blue box in section 2.3).

Option 2. Alternatively, you can use the Route Geometry Extractor to try to fix the First Route that you created, as follows:

- start Route Geometry Extractor (as in section 2.3);
- click **File**, then click **Select Route**;
- in the drop-down box, find and select First Route, then click **OK**;
- click **File**, then click **Load Quad-Tree**, if available;
- if the **Load Quad-Tree** option is not available, you will have to recreate everything: click **Edit**, then **Route values..**; check your previous entry here (remember that the **Directory Name** can't be changed), then click **OK**; then continue creating your route as before (but hopefully correctly this time): jump to the instruction to "click **File**, then click **New Quad-Tree**" in section A of this guide; once you have completed the new version of the First Route, get out of the Route Geometry Extractor and out of the Editors and Tools, and start again at the top of this blue box;
- if the **Load Quad-Tree** option was available, the same grid that you created previously should become visible: it is densest around the area where you created a set of tiles (London in our case);
- zoom in to that area like you did before, in several steps, and check that you see clearly again the blue-marked tiles that you selected earlier;
- if you don't see blue-marked tiles, create them again as you did before: jump to the instruction to "click **File**, then click **New Quad-Tree**" in section A of this guide; once you have completed the new version of the First Route, get out of the Route Geometry Extractor and out of the Editors and Tools, and start again at the top of this section B;
- **IMPORTANT**: right-click the middle tile in the bottom row of blue-marked tiles to show where we will start laying track (it will position the camera of the Route Editor there); this opens the pop-up menu: click **Route Editor Start Tile**, then click **OK**;
- click **File**, then click **Exit**:
- **IMPORTANT**: click **X**, then **Yes**, to exit from **Editors and Tools**;
- go now to the top of this blue box and start up Route Editor again.

You should know the following about what you see after a successful load of the First Route:

- you are looking through a "camera", which you can move and turn to view anything you like from any angle;
- initially the camera is located roughly in the center of the tile that you selected as start tile;
- initially the camera is located close to sea level, a tall person's height (2m or about 7 feet) above the terrain;
- initially the camera looks to the North, horizontally.

Next, you will select working windows, which are needed to operate the Route Editor (RE).

**To continue building our First Route, jump to the next blue box.**

## 3.1.2 STARTING IN RE

What you see (or should see) the first time you start your new route in RE is flat, green, bare terrain:  there is nothing else to see or use.

If you have trouble opening your route, see the options in section B of this guide.

The camera (your viewpoint) is roughly in the middle of the start tile (the camera's coordinates are shown in red just below the compass which is at the top of the screen);  it is 3m (about 10ft) above sea level and 2m (about 7ft) above the ground level;  and it looks due North (the red compass at the top marks 0°, East is at 90°, South at 180°, West at 270°).

When you start the RE again, the camera will be positioned the same way and at the same place, except higher if you have raised the terrain.

Note that if you add and/or delete tiles and retain the same start tile, the camera's default start position may change.

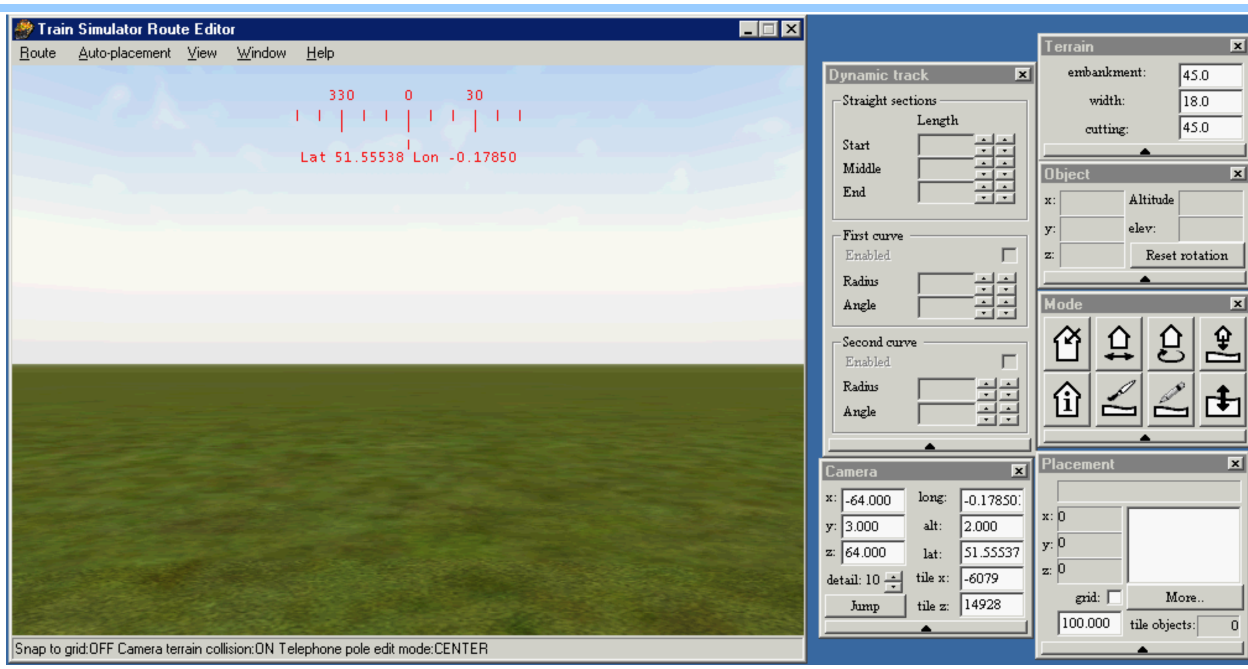
## C. Building "First Route":  selecting working windows in the Route Editor

After you have opened our First Route in the Route Editor, you should open a few windows that you will need for working (some may already be open by default).  For more details, see section 3.1.3.

**Do the following**:

- if you see no small window called "Object" on your screen:  click Window, then click **Object**;
- if you see no small window called "Placement" on your screen:  click Window, then click **Placement**;
- if you see no small window called "Terrain" on your screen:  click Window, then click **Terrain**;
- if you see no small window called "Camera" on your screen:  click Window, then click **Camera**;
- if you see no small window called "Mode" on your screen:  click Window, then click **Mode select**;
- if you see no small window called "Dynamic track" on your screen:  click Window, then click **Dynamic track.**

Your desktop may now look something like this (the RE windows can be moved, although not as easily as normal windows, because they resist overlaps):




From now on, these windows should open automatically each time you start Route Editor (unless you change them).

Next, you will learn to move and rotate the camera.

**To continue building our First Route, jump to the next blue box.**

## 3.1.3 SELECTING WORKING WINDOWS IN RE

When you start RE, you get at least one window: it shows the view from the camera.

To work with trains, terrain, objects, etc., you'll use several smaller windows. These are controlled by the Window option of the main window's menu. I suggest that you have the following windows permanently open: Object, Placement, Terrain, Camera, Mode select, and Dynamic track. You can move them to some extent around your desktop.

When you work with textures, also open the Texture select window (and perhaps the Texture window).

If your screen does not have enough space for all these, you can shrink each by pressing on its bottom bar, and enlarge it again later the same way.

**D. Building "First Route": moving and rotating the camera**

You will be using the camera a lot in Route Editor: so you should become very agile in moving and rotating it.

You move and rotate the camera by means of **arrow keys of the arrow pad** of your keyboard (**not the key pad = num pad**). This may vary with your keyboard. For more details, see section 3.1.4.

Try the following **to move the camera**:

- press the UpArrow: the camera goes forward;
- press the DownArrow: the camera goes backward;
- press the LeftArrow: the camera goes left;
- press the RightArrow: the camera goes right;
- press the UpArrow while holding the Ctrl key: the camera goes upward;
- press the DownArrow while holding the Ctrl key: the camera goes downward.

A short arrow-key press gives a short jump; a continuous arrow-key press makes the camera run; pressing an arrow-key while pressing **Shift makes the camera race fast** (this also applies to Ctrl-UpArrow and Ctrl-DownArrow); pressing an arrow-key while pressing **End makes the camera move slowly** (this also applies to Ctrl-UpArrow and Ctrl-DownArrow).

**To rotate the camera**:

- drag the mouse left, right, up or down, while pressing the right mouse button: the camera should turn left, right, up or down.

Try the following exercise: focus on a particular feature on the ground, like a dark spot; now "walk around" that spot, keeping the camera pointed at the spot, so as to see it from all sides. It helps to handle the mouse with the right hand, and the arrow keys with the left hand.

You will do an awful lot of this in Route Editor!

TIP: Look at the "y" value in the Camera window: that is the current altitude (in meters) of the camera. It will normally stay at least 2m above the terrain (which by default is 1m above sea level).

Now you are ready to lay your first track.

**To continue building our First Route, jump to the next blue box.**

### 3.1.4 HANDLING THE CAMERA

**The camera is your primary tool for laying tracks and shaping terrain**: almost everything happens visually, so you will be moving the camera all the time.

This section deals with the camera used in RE. By default it gives similar views to what you get in MSTS. However, it is possible to modify the views that you get in MSTS (for example to produce a "fish-eye" or "zoom" lens effect: that is described in Appendix J. Those changes, however, have no effect in RE.

Each time you open a route in RE, the camera looks North and horizontally. It is 2 meters above the ground, or 3 meters above sea level (its altitude is shown as "y" meters in the Camera window).

You **move the camera** with the four keys of the arrow pad (this may vary with your keyboard):

- **left/right with the left/right arrows**;

- **forward/backward with the up/down arrows**;

- **up/down by holding Ctrl while pressing the up/down arrows**;

- each of these motions can be **accelerated by** also holding **the Shift key** (this is especially useful to quickly move over large distances);

- each of these motions can be **slowed down by** also holding **the End key**;

- you **rotate the camera left/right and up/down by dragging the mouse** in the window **while pressing its right button**.

The camera will normally stay at least 2m above the terrain (it can go below water if the terrain there goes below the water level).

**To allow the camera to go below ground, press /**; pressing / again lifts the camera above ground.

You can **move the camera to a distant place** in one hop by entering its coordinates or the tile number in the Camera window and clicking on Jump.

TIP: To make the camera Jump, you need to know coordinates or tile numbers along the route. One way to find coordinates is to open an activity in the Activity Editor: as you point with the mouse at a place along the tracks, you will see its latitude and longitude displayed. Another way is to run the route in MSTS with the "head-up display" turned on (pressing 0 - zero - displays latitude and longitude on-screen); if the route has Activities, you can often start in various locations without having to drive there.

[NEW SINCE V1.106] To find the tile number of the current tile, look at the current value of "tile x" and "tile z" in the Camera window ("tile x" increases to the east, while "tile z" increases to the north): I find it convenient to write the tile numbers on a map of the tiles for the route (see section 2.7), so I can jump to that tile by entering its "tile x" and "tile z". If you need to frequently jump to distant locations along a route, another method is to move the camera once in RE along the track to the desired locations, and to copy the "long" and "lat" values shown in the Camera window to a file (with an identifier to remind you what is there). Then, the next time you need to go to one of those locations, copy the coordinates back into the Camera window and press Jump.

### 3.1.5 DEFINING TILE PROPERTIES

There are some settings that you can choose for each tile separately and which can be changed at any time. **To see the tile and patch grid, press F7**: the blue grid lines define the tiles (they may be far away, since they are about 2km apart), while the black grid lines define the patches (16 x 16 of them in each tile).

After you **left-click within a tile** (a tile has blue borders; the borders of one patch in that tile will turn red), **right-click to open a pop-up menu**. You can now set:

- **altitude limits** ("Set tile floor and scale"; by default -63m and +65m): the terrain within the selected tile will not drop below or rise above these limits; it has been written that if you choose larger limits, the altitude resolution may suffer: you may not be able to tune the terrain as finely any more; nevertheless, the Innsbruck - St. Anton route uses lower and upper altitude limits around +1100m and +2600m near St. Anton, with no apparent loss of altitude resolution; it has been reported that MSTS has an upper altitude limit between 3500 and 3600m; it is best to keep the lower and upper limits as close together as possible;

- **water level** ("Set tile water height"; by default 0m): you can define the water level at the 4 corners of each tile (sloping water may seem unnatural, but rivers and ground water actually do slope); to see water in MSTS (as opposed to RE), make sure the water option is checked on in the MSTS Advanced Display options;

- **error bias** (by default 1): set this to 0 to enable finer detail in the terrain.

[NEW SINCE V1.106] NOTE: If you redefine a tile's altitude limits to include the current limits, the current terrain will not change. But if you redefine the limits to be totally above the current limits, the terrain height will be set to the new upper altitude limit. For example, by default the lower and upper limits are -63m and +65m; if you define new limits of -100 and 800m, all existing terrain in that tile will stay unchanged; but if you define new limits of 1000 and 2000m, all terrain within that tile will be set to 2000m, producing a flat "terrace" at that altitude (since this involves changing a lot of altitude values, you should save the result after each change of limits in a tile, at least once per tile).

NOTE:  Make sure that the minimum and maximum altitude limits are compatible between neighboring tiles.  The min-max range of one tile should overlap the min-max range of each neighboring tile.  Otherwise you will get big steps in the terrain.  In any case, when you redefine tile altitude limits, especially in mountainous terrain, you may create large terrain "walls" between adjacent tiles.  These can be removed like any terrain feature (except perhaps along the free edge of tiles):  for example, by laying track and pressing Y, or by selecting an area and setting its height.

### 3.1.6 DEFINING PATCH PROPERTIES

Among other options, each 128m x 128m patch in each tile can be set **to show water** or not (this can be changed later if you like):

- **press F7 to see the tile and patch grid;**

- **left-click within a patch** (with black borders);  if you want **to select a rectangle of patches, press Shift while selecting the patches at two opposite corners of the rectangle**:  the patch border(s) will turn red;

- **right-click** to open a pop-up menu;

- **toggle water on (or off)**.

You will only see water after you depress the terrain so that it falls under the water level.

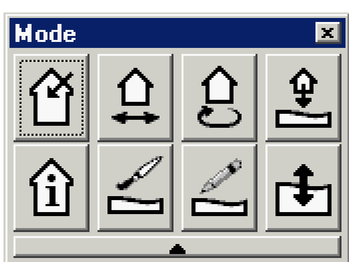
There is a disadvantage in having water toggled on:  it makes it harder to see tracks and objects in the wire frame mode, so you might delay this step (or toggle water off and on again later).

### 3.1.7 USING THE MODE WINDOW

The mode window allows you to switch between various important functions of RE, such as selecting a track or object, moving it, rotating it, placing a new one, etc.

You can select a mode either with a Fn key, or by clicking on the corresponding icon in the Mode window.

The 8 mode icons are arranged in a 4x2 rectangle, as follows:



The corresponding Fn keys and their actions are the following:

| | | | |
|---|---|---|---|
| **F2**<br>Select Object | **F3**<br>Move Object | **F4**<br>Rotate Object | **F5**<br>Place Object |
| **F6**<br>Edit Object Properties | **F7**<br>Terrain Texturing | **F8**<br>Draw on Textures | **F9**<br>Alter Terrain |

### 3.1.8 SAVING AND BACKING UP

**It is important to save and back up frequently**, to avoid losing a lot of work and time (it is easy to make fatal mistakes in RE, so you will likely need to use a back-up copy at some stage). Here is what I do:

- **save every 5-10 min** (press Ctrl-S or select Route | Save); **make sure no track section or other object is selected when you save**;

- **back up every 30-60 min** (put a copy of your route's folder outside the Train Simulator folder or on a CD), using Windows Explorer or My Computer; distinguish versions for easy recognition by adjusting their folder name, for instance as

NewRoute - date_time - last action - status

where "NewRoute" is your route name, "last action" reminds what you changed last, and "status" tells whether the route functioned OK or failed a test run; you can retrieve a backed-up route by first deleting the entire "NewRoute" folder in MSTS; then copy the backed-up version into the ROUTES folder of MSTS (remember to rename that folder within MSTS to its original name, such as NewRoute, or else it will fail); it is wise to already back up the bare route coming out of RGE! you will not regret keeping many backup copies (if your hard drive is big enough).

TIP: You can halve the disk space required for saving by zipping your route (using WinZip, for example). You can also much reduce the needed disk space by backing up only the following folders of your route: main folder, Activities, Lo_Tiles (if it contains files), Paths, Services, TD, Tiles, Traffic (if it contains files), Tutor (if it contains files), and World.

- **reboot your computer every 1-2 hours**;

- **every few hours** (or whenever you encounter track problems), **load the route into RE with the Advanced option "Rebuild track database"**;

- **if the RE screen flickers or goes black** (especially while saving), **stop immediately, save and reboot** (this happens when Windows has used up your computer's memory and cache space).

### 3.1.9 RECOVERING FROM "TRACK DATABASE" ERRORS

Track database errors may be caused in various ways. You may have made an error in placing tracks. But these errors are also often due to Windows memory overflows: you can minimize them by rebooting your PC more often.

You will notice such errors mainly when loading your route into the Route Editor. It may say, for example:
- "Failed to load track database";
- "Track database read failed";
- "Error in Track database: 'remove_end_end' entry is no vector";
- "Track database error: 'remove end_end' one non-vector, one vector";
- "A lock point on this track piece coincides with a lock point on an unloaded track piece. Track database entry will not be made";
- or it may ask you to remove signals.

If "Failed to load track database" appears when you try to load your route into RE, do this:

- **delete** the following one or two files in your route's folder (assuming it is called NewRoute): **NewRoute.tdb and NewRoute.rdb** (this second file only exists if you have installed roads: it seems to contain information on roads as opposed to tracks);
- **do not delete** the backup files (their names end in .bk);
- **load the route into RE with the Advanced option "Rebuild track database"**;
- save the route.

If you continue having the same problem, it sometimes helps to repeat the process again.

It seems that adding roads before finishing the tracks is one source of this problem: **first lay all your tracks, then add the roads**.

In some cases, you will have to repair some tracks. For example, in the case of "Error in Track database: 'remove_end_end' entry is no vector", you may find that the editor locks up when you try to select a track section. Then you must restart RE and delete the surrounding track sections, before laying new tracks to fill the gap. The same error message can mean that there is a bad joint between track sections: such a bad joint may show up as blue poles where they should not show up, but a bad joint may also not be visible in any way. It is then best to lay tracks in the affected tile again (the tile where the problem exists is given by tile x and tile z in the Camera window at the time the error message appears). Note that the same error may occur with road sections.

[NEW SINCE V2] It appears that the "Error in Track database: 'remove_end_end' entry is no vector" can occur especially with track sections that cross a tile boundary, or two tile boundaries near a tile corner. In particular, dynamic track that is adjusted (or deleted) in such a place has been seen to cause this problem. One possible solution is to readjust or replace such dynamic track. Another is to repeatedly rebuild the database (after backing up!) until the problem disappears: the problem may shift from place to place, and end up being resolved.

## 3.2 Guides for laying tracks

When you lay tracks, you will constantly ask yourself: "Where do I head now? In which direction should I lay my tracks? What track length and curvature do I need?" To show you the way and suggest track curvature, it is useful to insert some visual guides in the landscape: then you can lay tracks following those guides. We here sketch three different methods, which are useful in different circumstances.

### 3.2.1 PLANTING TREES

You may have a good drawn map of your planned route, showing where your tracks will be with respect to the tile and patch grid: then a very convenient guide is to plant trees to mark the planned route.

To do this in RE, make sure you know where you are (that is where the camera is), relative to the tiles and patches that you already have on your drawing of the route.

Now place a tree every 100 or 200m along the planned route, for example every time the route crosses a patch border (these are 128m apart).

To place a tree, do the following (this is the same procedure as for any object):

- press More.. on the Placement window: this opens the Object selector;

- under Object class, select Vegetation;

- select any tree in the Objects list;

- press W to show the wire frame, which defines the patches (to be sure where the tiles and patches are, press F7);

- move the camera close to where you want to place a tree;

- press F5;

- point the mouse (as the + symbol) at the desired location, and left-click;

- repeat the last three steps for other trees.

Once you have placed trees, lay your tracks so that they pass close to the trees.

You may delete the trees any time later, but first test the tracks in MSTS. (The trees don't cause any problems in MSTS, even if they are located on the tracks.)

### 3.2.2 USING MARKERS

You may know the coordinates of points along your tracks, for example for a real-life railway line: then you can use the RE "markers" as a visual guide. These markers are visible in RE as large flags or "advertisement" posts. There are three kinds of markers, so you can distinguish, for example, between stations, switches and tunnel entrances.

Markers are listed in a *.mkr file, located in the main folder of a route (such a file is optional). Its name should match that of the other files specific to your route: for example, if your route has a file NewRoute.ref and NewRoute.ace, the marker file should be called NewRoute.mkr.

A typical marker file looks like this:

```
SIMISA@@@@@@@@@@JINX0I0t______
Marker ( -2.92978 54.8882 carlisle_station 2 )
Marker ( -2.28325 54.0673 SettleStation 2 )
Marker ( -2.27939 54.0736 L1 0 )
Marker ( -2.27862 54.0772 L2R1 0 )
Marker ( -2.27939 54.0807 R2 1 )
```

(This is the beginning of the SettleCa.mkr file of the Settle and Carlisle default route; even though I have only listed a few lines, this is enough for a complete file: no other lines are needed to make this list a valid marker file.)

Each line in a marker file gives a marker's longitude, its latitude, a name (that you choose), and a number (0, 1, or 2, which you choose to distinguish between the three types of flags).

WARNING: The *.mkr file must be in Unicode format. This means you won't be able to create it with an older NotePad, but you must use a newer NotePad, WordPad or Word, and save as a text file in Unicode format. Also, if you use a name that includes a space, you must enclose it between quotes, as in "Settle Station".

WARNING: Do not cross tile boundaries while in F7 mode (which shows tile and patch boundaries), or markers could shift by large amounts.

### 3.2.3 USING WATER

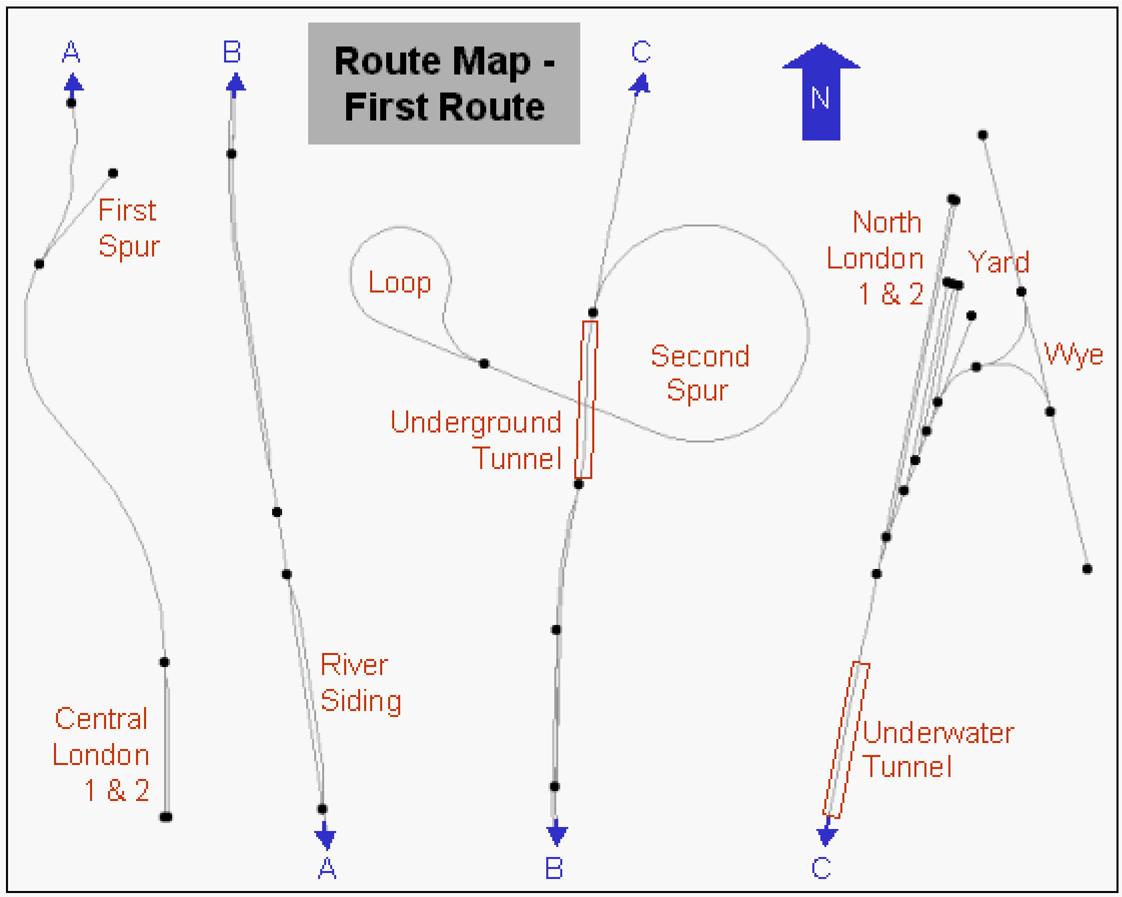
If you plan to have a lot of water (sea, rivers, lakes), it is convenient to make it visible early on, because it will help you as a guide to laying track (in the absence of markers).

Prepare a rough coast/river/lake outline with the Alter Terrain Height tool (press F9): see section 3.5.2. (You can fine-tune the coast details later, after the tracks have been placed and the on-land terrain is shaped.)

## E. Building "First Route": laying the first tracks

We will next lay the first tracks of "First Route", and make a simple station ("Central London"): it will consist of two parallel tracks (imagine two platforms alongside these two tracks, which we will place later) and a switch (called points in the UK), which leads to a single track heading North, toward "North London".

The "First Route" will ultimately look something like the following diagram (in which the route has been cut up in four parts for easier display): you will start building the Central London station at bottom left, and lay track toward point A, then B, C and North London. As you progress, you will be learning new techniques at each stage. The distance from Central London to North London is about 3 miles (5 km).

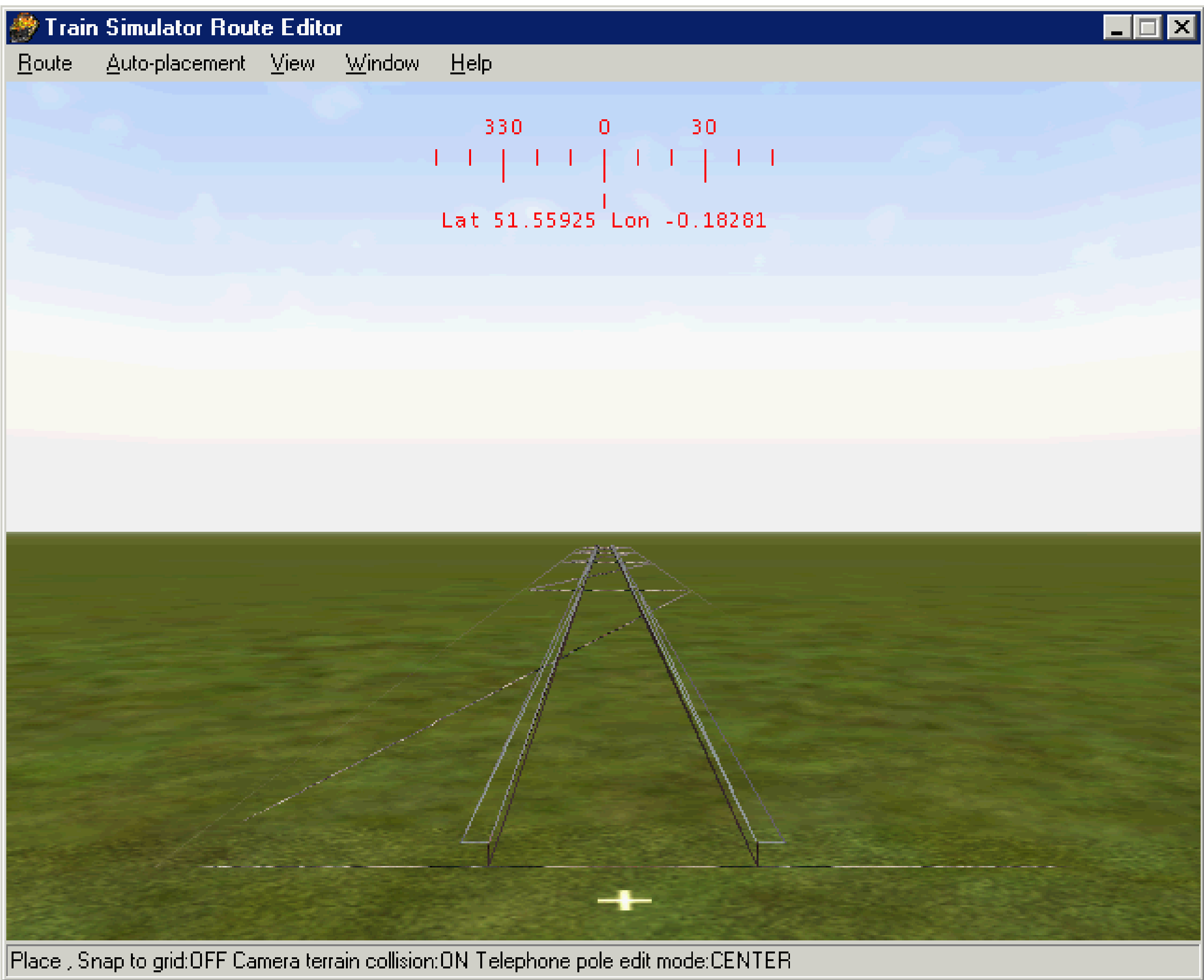



See section 3.3 for more details on laying tracks.

IMPORTANT: We will make a fresh load of our First Route into the Route Editor (RE for short), so the camera is positioned in its default start-up position in the start tile of the route (that way your first track will always show up immediately when you load First Route, thus helping to find your way).

Do the following **to lay the first track section**:

- Click **Route**, then click **Open**; select First Route in the drop-down list, and click **OK**;
- CAUTION: **don't move or rotate the camera** until you have laid your first track!
- click **More..** in the Placement window: the Object selector window opens up;
- in the drop-down list, select Track sections; a long list of strange names appears: they label a variety of track sections that you can choose to put on the terrain;
- near the bottom of the initially visible list of names, you will see one called A1t100mstrt.s: this is a single-track (1t) section that is 100m long (100m) and straight (strt);
- click on A1t100mstrt.s, then click **OK**: this name now appears in the top box of the Placement window; this track section is now ready to be placed on the terrain;
- press F5: the top right button in the Mode window activates (you could have clicked on it to achieve the same result); this puts RE in its "Place object" mode;
- move the mouse over the terrain: you should see a white + symbol as the mouse pointer; it points where you will place a track section;
- place the + symbol "as close to your feet as you can see", namely at the center near the bottom edge of the screen: see the screenshot below;
- left-click: you should now see the following:



**That is your first track!** It does look "wiry", but that is only because it is in its "selected" state, showing its wire frame. Continue as follows:

- press F2, then click anywhere away from the track: now you see a proper track, with ties and ballast.

The only abnormal things about this track section (when it is not selected) are the blue poles at both ends (they mark where other track sections will be attached), and the white/gray line that joins the two poles overhead (it will help show how tracks connect to form a railway line).

Now admire your first track from all sides, from high above, from close to the side, etc: that is just another exercise in camera handling.

Three important things to note:

- the track section lies on the ground;
- it is horizontal (not sloped);
- it points straight North (see the red compass showing 0º for North).

(Note also that when you move the mouse across the track, it turns red: when it is red, you can left-click to select the track section again.)

If by mistake you have placed another track section on the ground, **delete** it as follows:

- press F2;
- select the track section to be deleted (point at it so it turns red, then left-click);
- press Delete.

Before you continue, **save your achievement for posterity** (you will still be able to remove this track section later if you should ever wish!):

- deselect any selected track section or other object;
- press Ctrl-S; or click **Route**, then click **Save**;
- to the question "Save world placement changes?" answer **Yes**;
- to the question "Save terrain changes?" answer **Yes**.

To check that this track section will indeed appear in front of the camera when you load First Route, do this:
- Click **Route**, then click **Open**; select First Route in the drop-down list, and click **OK**; you should see the track the same way it looked when you placed it.

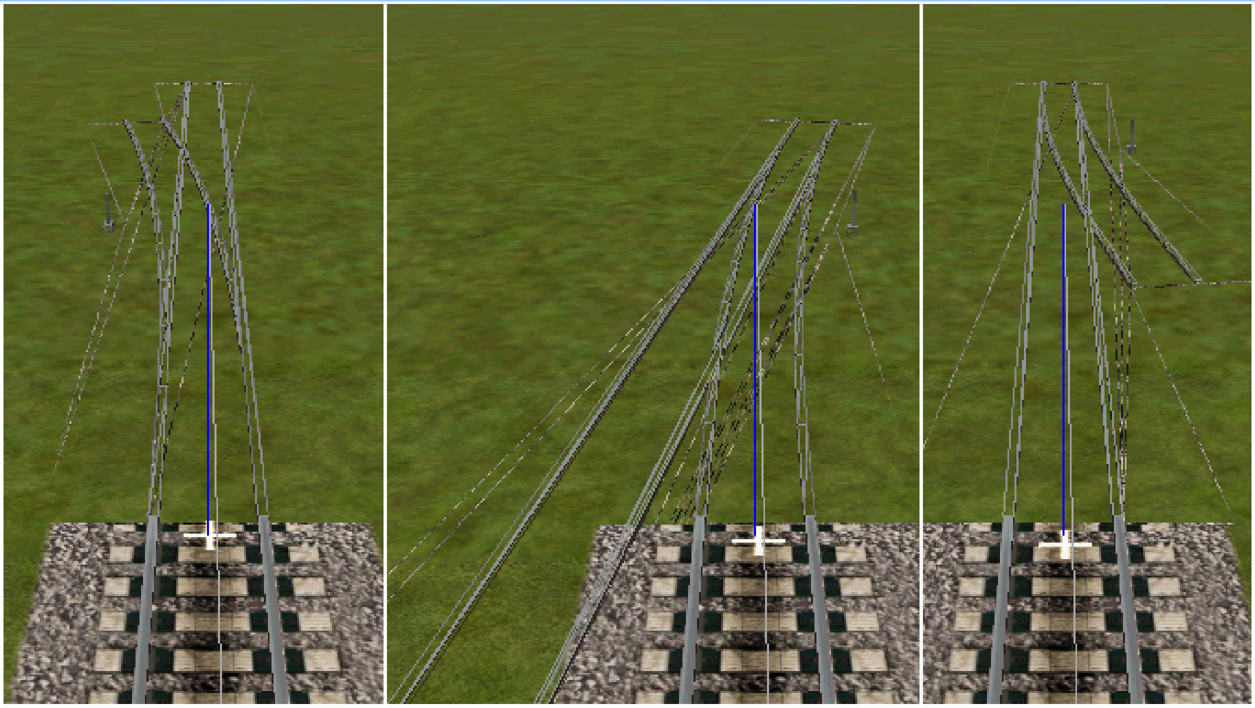
Now that we started laying track, let's continue: it's so easy!

**Our First Route starts with a pair of tracks in a station: they will join up through a switch (called "points" in the UK) into a single mainline. To make this layout, follow these steps:**

- move the camera forward, so it is near the far end of the first track;  make sure you see the end of that track section;
- select again the A1t100mstsrt.s track section (click **More..** in the Placement window, select the Track Sections, and find this particular track section again;  select it so it appears in the Placement window);
- press F5 (to activate Object Placement mode);
- point the + symbol over the visible far end of the existing track;
- left-click:  your second track section appears, neatly connected to the first!
- press F2, and left-click away from the second track section, to deselect the new track section: you can now see how perfectly it is connected to the first;
- note also that the blue poles have disappeared where the two track sections have joined;  also the white overhead wires have joined up to form a continuous white line:  these two facts tell you that RE has accepted the joint as being a good joint.

We have laid 200m = 2 x 100m of track:  that should be plenty for a platform;  now let's prepare to **add a switch**, which will allow a second track to come back along the right side of our first track sections, along a second imaginary platform.  Do this:

- in the Object selector (press **More..** in the Placement window), select, among Track Sections, one called A1tPnt10dLftMnl.s (note that the track sections are listed in alphanumeric order): that ugly name stands for a point (switch) turning 10º to the left, with manual control (as opposed to computer control:  manual control allows the train driver to throw the switch);
- make sure A1tPnt10dLftMnl.s shows in the top box of the Placement window (A1t100mstrt.s has moved into the larger box below that);
- move to the far end of the latest track section, but not too close;
- press F5;
- point the + symbol over the end of the track, and left-click:  you should see a switch turning off to the left, as shown in the left part of the next figure (in this figure, the point of view is a bit higher):



That is not too surprising with a left (Lft) switch!  However, the intention was to place it as shown at the right in the figure, turning back to the right toward the beginning of our track.  That is easy:

- press T:  this produces the arrangement shown in the middle of the figure, still not correct;
- press T again:  we get what we want - the right part of the figure.

What has happened is that a switch, because it has three ends, can be attached three ways to existing track:  pressing **T toggles** between those three **orientations** (if you pressed T again and again, the switch would swing around and around).

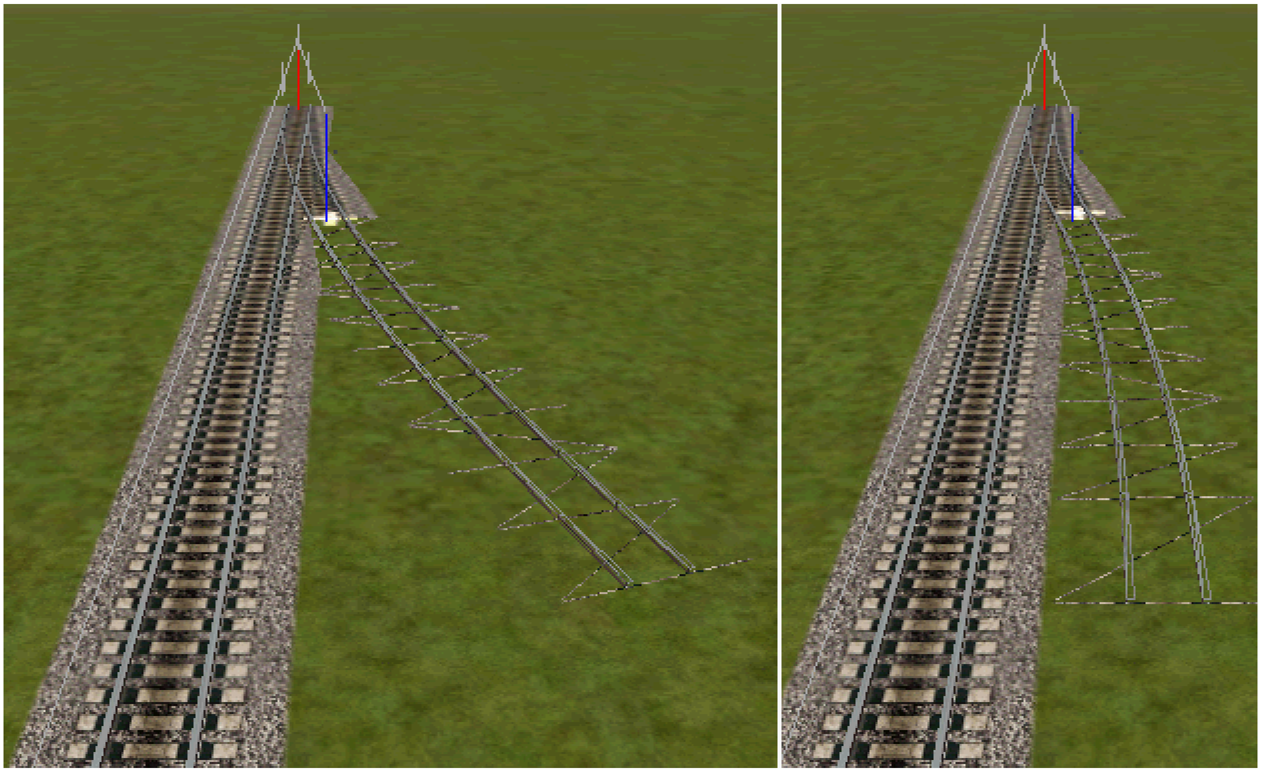
NOTE:  You could not have obtained this result with a right-handed switch (A1Pnt10dRgtMnl.s)!

NOTE:  After deselecting the switch, a red pole shows up (see below);  that one will always remain:  it identifies a switch.

Now that the switch is correctly attached, we want to curve the track back to being parallel with our original straight track.  For that a special curved track section is available, called A1tEndPnt10dLft.s:  it is made to complement A1tPnt10dLftMnl.s in exactly the way we need.  To place it:

- find and select A1tEndPnt10dLft.s;
- place it at the near end of the switch:  you get what is shown in the left part of the next figure.

CAUTION:  It is important to remember that **you must attach tracks to all three exits of a switch**:  any unconnected switch exit will cause the route to fail in MSTS.



That is again not what we wanted!  We wanted what you see in the right part of the figure.

Again that is easy to achieve:

- press T:  this reverses the left turn (as seen from the switch) into a right turn.

What is a bit confusing is that there is a "right-handed" curve available (A1tEndPnt10dRgt.s), which would make the correct turn immediately:  however, it is meant to accompany the right-handed switch (A1tPnt10dRgtMnl.s)!

NOTE:  You will often find that curves, when first placed, turn to the left:  you will have to press T to make them turn to the right.

For safety:
- deselect any selected track section or object;  press Ctrl-S;  or click **Route**, then **Save** (always answer **Yes**, and **Yes** if asked a second time).

Next we will **add straight tracks** to the second platform line to bring it **back to the beginning of your first track**:  the question is whether the two tracks will end together, or one will be longer than the other.  Let's see:

- select the A1t100mstrt.s straight section:  just left-click on it where you see it in the Placement window box;
- place it at the near end of the curved track section that you just laid:  the new section should align itself perfectly parallel to the first platform line that you laid earlier;
- move back to the free end of the last-laid section;
- place another copy of the A1t100mstrt.s section;
- move back to see its end:  it comes back too far!

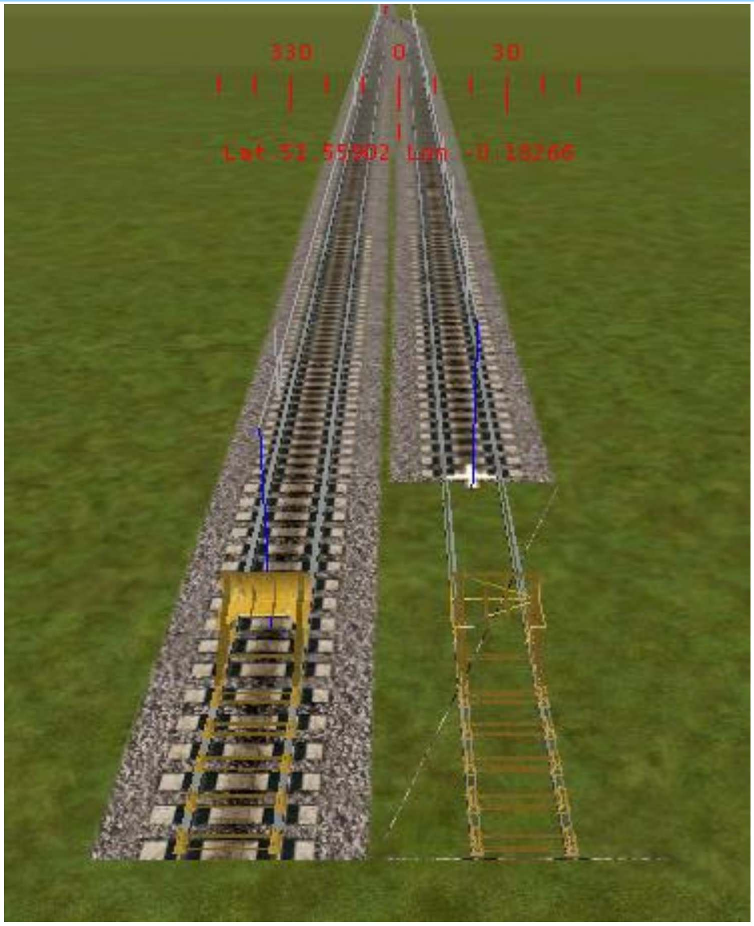
**How do we line up the two track ends?**  Let's try placing shorter track sections (they exist with 50m and 10m lengths) and hope that they will fit in some combination:  we can place one 50m section, and then add a few 10m sections, to see whether we get a good line-up.  Do this:

- press F2;
- select the last 100m section that you placed (the one that was too long);
- press Delete: it should disappear completely;
- find and select the track section A1t50mstrt.s;
- press F5;
- place it where the just deleted 100m section was attached;
- find and select the track section A1t10mstrt.s;
- attach it to the 50m section;
- attach more 10m sections until you get close to the end of the parallel track.

This will succeed pretty well with 3 sections of 10m length.  In fact, this good fit is not an accident and it is also extremely precise:  the MSTS switches and their associated curved tracks are designed in such a way that they create an offset that is an exact multiple of 10m.

To finish off our two platform lines, we are going to **add a buffer** (bumper) **to each line**:

- select, among the Track sections, the one called A1tUSBuffer.s;
- place it at the free end of your very first track section;
- place another buffer at the end of the parallel line, resulting in this (before deselecting the right buffer):




Now you have laid tracks for a simple station:  you can recognize the beginnings of the station illustrated in Section B!  Let's **add the beginning of the main line leaving this station** - we will use curved track:

- go to the far end of the switch;
- select track section A1t500r20d.s;
- press F5;
- point the + symbol at the far end of the switch;
- left-click:  a long track section appears which curves to the left;
- go to the end of that new curved track section;
- add another curved track section.

Each of these two curved track section turns left by 20°, so the far end of your track is now heading 40° to the West of North, which is 320° on the compass:  confirm this with the compass at the top of the RE window.

NOTE:  You could have made a right turn, by pressing T after laying each curved track section.

**Save your work, after deselecting any selected track or objects!  It would also be good to back up your FirstRoute folder again.**

The next task will be to extend the track over a hill.

**To continue building our First Route, jump to the next blue box.**

## 3.3 Laying track

A large number of tracks sections are available by default in any new route. A complete list is provided in Appendix B, together with useful information about each track section.

[NEW SINCE V1.106] Other track sections can be imported as add-ons from the web: see section 3.3.13.

### 3.3.1 SELECTING A LAID TRACK SECTION

To **select a track section that is already on the ground**, do this:

- press F2;

- point with the mouse at the track section or object;

- if it turns red, left-click: it should now turn to a wire-frame version;

- if it does not turn red, it must have a handle (a green shape near the track's end, characteristic of a dynamic track section); point at this handle so it changes appearance (it should become black), and left-click (this may be tricky to do: you may have to zoom in and/or view from a different side); this kind of track does not turn to a wire-frame version.

### 3.3.2 DELETING A LAID TRACK SECTION

You may delete a track section at any time: right after laying it down, or later when it is part of a complete track. In the second case, make sure that you reinstall track sections that connect properly (they do not have to be of the same type: for example, you could replace a straight track section by a switch).

To **delete a track section or object**:

- select it;

- press the Delete key.

If a track section refuses to be deleted, you should try to reload the route into RE, but asking for the track database to be rebuilt (select the Advanced option and check the box).

### 3.3.3 CHOICE OF TRACK SECTIONS

There are many track sections to choose from:  various straight and curved tracks;  single, dual, triple and quadruple tracks;  dynamic tracks;  switches;  tunnels;  a bumper (buffer);  etc.

A complete list of the available track sections is given in Appendix B.

The **track sections are named as follows** in the Object selector:

- "Dynamic track piece" is a very flexible, relatively complex track section (by default, this track section is in the Dynamic track class;  but if you import objects from other routes, it may end up in the Track section class;  see section 3.3.8 about the use of dynamic tracks);
- "A1t", "A2t", "A3t" and "A4t" designate single, dual, triple and quadruple tracks separated by a common standard spacing (5m);
- "strt" designates straight tracks;
- "100m" (and the like) gives the length in meters of straight track;
- "1000r" (and the like) gives the turn radius in meters of curved track;
- "10d" (and the like) gives the turn angle in degrees of curved track;
- "Tun" labels a track that is covered with a high tunnel;
- "RndTun" labels a track is covered with a low round tunnel;
- "wtr" specifies that the track has a water trough for steam engines;
- "Skew" specifies that the track gives a sudden small change of direction (avoid this kind of track section if possible:  see discussion in section 3.3.11a);
- "Lft" and "Rgt" specify that a curved track or switch turns left or right, respectively;
- "Pnt" indicates a switch ("points" in the UK);
- "EndPnt" labels a curved track section that complements the corresponding switch, by providing an opposite curve that allows continuing at the standard dual-track spacing;
- "Mnl" indicates that the switch can be operated manually by the train drivers (without "Mnl" a switch can only be operated by the computer);
- "Buffer" specifies that the track section includes a buffer (bumper);
- "X" labels a crossover of straight tracks;
- "Y" labels a symmetric switch, mainly used in a "wye" (for reversing trains);
- "LvlCrs" indicates that the track provides a level crossing with a road.

**Note that curved tracks often only come with a left curve!  Press T after laying such a section to make it curve to the right.**

A convenient track section for tighter turns is A1tEndPnt10dLft.s (its radius is about 172.7m): although this section is meant as a complement to a switch ("point"), it can be used anywhere.  It also is available with both types of tunnels (high vs. round).  You can make right turns by pressing T, or by using A1tEndPnt10dRgt.s.  (Note that A1tEndPnt10dRgt.s is not quite the same as A1tEndPnt10dLft.s after reversing with T!)

[NEW SINCE V1.106] If you use engineering data to lay track, you may find curves defined not by their radius of curvature and turn angle, but by their **degree of curvature**.  Appendix E helps you to convert such information to the MSTS practice of radius of curvature and turn angle.

[NEW SINCE V1.106] **It is recommended to initially use only manual switches** (labeled "Mnl"). The other switches are only computer-controlled, so the train driver can't switch them when driving an Activity. On the other hand, computer-controlled switches may be necessary to create more complex activities that have computer-controlled trains running along preselected paths: these trains often require the computer to freeze certain switches for them. A more complete discussion of this topic is given in Chapter 5.

Dynamic tracks offer much freedom and flexibility, but are more complex and time-consuming to use: they are discussed in section 3.3.8.

Not all track sections are available for tunnels: in particular, switches and dynamic tracks do not have tunnel walls.

Round tunnels (labeled with "RndTun") are low: they are too low for most American rolling stock. However, MSTS will let American rolling stock pass through them without a problem. It will just look strange!

### 3.3.4 PLACING TRACK ON THE GROUND

The **general idea** is to lay your very first track section, orient it properly, adjust the ground to it (or adjust it to the ground), and then add track sections to one or both ends, adjusting the ground below each section as you go; you continue this way until you have created the entire route (or large sections of your route). This will create a strange terrain with dikes and trenches for the tracks: the rest of the terrain (away from the tracks) can be adjusted later.

You **build a route by adding new track sections to existing track sections**, not by building independent unconnected routes to be connected later.

**To lay a track section**:

- click on More.. in the Placement window: this opens the Object selector window;

- select, from the drop-down list, either Dynamic track (if you want to select a section of dynamic track) or Track sections (if you want to select a track section with fixed dimensions);

- in the Objects list, click on the track section that you want (there is only one option in the case of dynamic track), and click on OK: it now appears in the top box of the Placement window (when you select another track section, the previous one will be placed in the bigger box to the right in the Placement window, available for reselection by a simple click there);

- press F5: a white + sign now shows on the ground where the mouse is pointing;

- point the + at the place where you want to lay the track section;

- left-click:  a wire-frame version of the track section appears (except for dynamic track);  the wire frame indicates that this particular track section is now selected, ready to be manipulated further (in the case of dynamic track, it is also now selected and ready for manipulation, even if it does not show a wire frame).

If you placed the new track section near the free end of an existing one (a free end is marked by a blue vertical post), **the new section will snap to line up and connect properly** (it will be horizontal, whatever the slope of the existing track section, and whatever the slope of the terrain: see section 3.3.6 to make it slope).  If you placed the new track section too far from the desired joint for it to snap on, do this:  select it, press F3, then left-click while pointing at the desired joint position (the free end of the existing track), so it connects.

**With a curve, switch or other more complex track section, press T to connect it differently to the existing track.**

To **check that track sections connect properly**, watch the white lines overhead each track section (after deselecting track sections):  those white lines should join without leaving blue vertical posts;  also the white lines should join up logically (the way a train would ride, without hopping randomly between tracks).  If you spot an error, you should delete and lay the last track sections again.

NOTE:  The elevation of the beginning of the selected track section appears as "y" (meters) in the Object window.  This elevation is 1m for the first track, unless you have changed the terrain.

To **delete a track section**:

- select the track section (press F2 then left-click on the track section);

- press Delete.

**It is OK to have free track ends**:  there is no need to finish off a line with a buffer (bumper), or anything else.

CAUTION:  It is important to **attach tracks to all three exits of a switch**:  any unconnected switch exit will cause the route to fail in MSTS.

[NEW SINCE V1.106] You may want **to replace (substitute) track sections in an existing line of tracks**, connected at their ends to other track sections.  For example, you may want to replace a manual switch with an automatic switch.  This can be easily done if the new track section has the same size as the old one.  Do the following:

- back up your route;

- make sure that no track objects are attached to the track section you want to change (signals, mileposts, speed limits, etc., all of which have "handles" in the middle of the tracks):  any track objects must be removed before you change the track itself;

- select one track section (press F2 then left-click on the track section);

- delete it (press Delete);

- insert the new track section;

- adjust its direction (press T), if appropriate;

- adjust its slope, if needed (see section 3.3.6).

If you replace existing track sections with sections of a different size, you may create a gap that you will have to fill with dynamic track (see section 3.3.11).

IMPORTANT: **Make running tests after each major track change**, especially after adding a loop or wye. See section 4.

### 3.3.5 PLACING THE FIRST TRACK SECTION

**The first track section can be positioned anywhere.** Use your drawn map, guides (see section 3.2) or known coordinates to position it. You can see the tiles and patches after you press W: then you can position your first track according to your hand-drawn map. To use known coordinates, move the camera so that its longitude (long) and latitude (lat), both shown in the Camera window, are about right: then position the track section.

You may want to prepare the terrain height or shape before laying the first track(see section 3.5), if you want to start at a higher altitude (the altitude of the beginning of the track section is shown as y meters in the Object window); or you can raise the track section to the desired height, and then adjust the terrain to match.

**That first track section will be positioned on the ground, oriented northward.**

You can **reorient** the first track left or right by pressing F4, then pressing the left or right arrow key, or dragging the mouse left or right (make sure the track section is selected). Unfortunately, the angle of the track orientation is not shown: you have to guess it roughly from the "compass" at the top of the window.

You can **move** the first track by pressing F3, and dragging it with the mouse. Usually, it will end up above or below the surface, unless you press Ctrl while dragging; press H to put it back on the surface.

You can **raise or lower** the first track section by pressing F3, and then the NumPad up- or down-arrow: you can see the altitude marked as "y" meters in the Object window.

You can also make the first track slope up or down, as discussed in section 3.3.6.

## F. Building "First Route": adding sloping tracks over a hill, and creating a spur

You will next extend the track that leaves our little station so it climbs and curves over a hill. Also, you will let the mainline branch off onto a spur.

For now we will only shape the terrain close to the track (see section 3.3.7 for details): the track will therefore ride on top of a "dike" that has little resemblance to a real hill. Later we will come back and shape a proper hill here.

So we need to first **lay track that slopes up**. See section 3.3.6 for details.

**Do the following:**

- load First Route into RE (the Route Editor), if it is not loaded now;
- move along the track to its end;
- select track section A1t50mstrt.s from the Object selector: this is a 50m straight section;
- place this track section at the end of the curved track (use F5): **do not deselect it** (keep it selected);
- press F4: this puts RE in its Rotate Object mode; you see the mouse pointer change to a pair of interlaced circles;
- drag the mouse downward: you should see the track section tilt up;
- drag the mouse upward: you should see the track section tilt down, even dipping underground;
- as you tilt the track up and down, look at the Object window: you will see the "elev." box display a slope that ranges between 3° (the maximum up-slope) and -3° (the maximum down-slope);
- now adjust the slope of your track section until its "elev." is 1.05° (a moderate slope).

Now your track section is sloping up into the air. We can **bring the terrain to the level of the track** very simply. Look carefully at the terrain as you do this, because the effect is very small this close to ground level:

- **press Y**.

After a moment the terrain will slightly jump up to fit the sloping track. (You can go back and forth by using the **EditUndo button - press the back-quote `** - then pressing Y again).

Now **add a second track section, with a steeper slope** (the intention is to raise the slope gradually, since a sudden change of slope from 0 to 3° is too abrupt and unrealistic; we will later make even gentler transitions with several shorter track sections, each one sloping a bit more than the last):

- go to the end of the last track section;
- press F5;
- place another identical track section;
- press F4;

- drag the mouse down until you see the track "elev." mark 2.1°;
- press Y: the terrain jumps a bit higher now, to the level of this higher piece of track.

NOTE: In the Object window is another useful piece of information. The "y" value is the altitude (in meters) of the beginning (not the end) of the selected track section: it should read 1.916m now, meaning that it is 0.916m above the ground level, which in turn is 1m above sea level (not much of a hill yet!).

**Save your work, after deselecting any selected track or objects!**

Next we will **turn to the right while climbing** at the maximum slope of 3°. But we will select a sharper curve than before, as may be necessary in hilly terrain. We will use a track section that is not obvious for this purpose: A1tEndPnt10dRgt.s (we have used its left counterpart earlier to complement the switch). It is convenient because it has a radius of about 172.7m.

Do the following:

- select track section A1tEndPnt10dRgt.s;
- press F5;
- place the track section at the end of the straight track;
- press F4;
- tilt it up to a slope of 3°;
- press Y.

After pressing Y, you may notice the terrain near the end of the last straight track section rising above that track level! That is normal: the Y function creates flat terrain for some distance around the selected track, which can bury existing lower track.

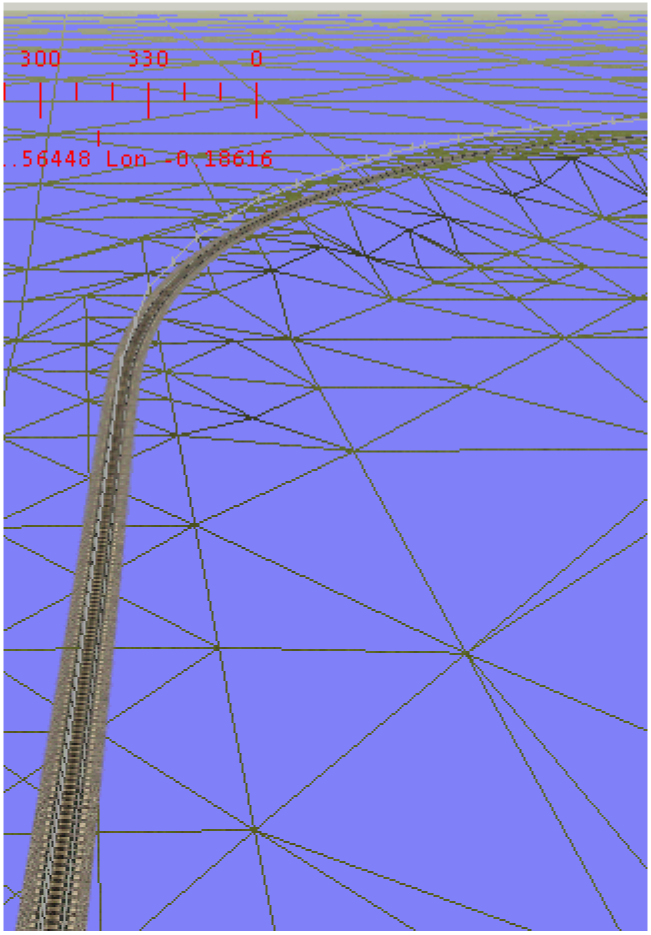
To unbury the previous track, do the following:

- press F2;
- select the partly buried track;
- press Y: this brings the terrain down to the level required by the lower track (it may leave the higher track slightly off the ground, but that is hardly noticeable from the train driver's perspective).

**Add three more sections of the same curved tracks** in the same way, with slopes of 3°, adjusting the terrain after each track (you will be pressing F5 and F4 often!).

Don't worry about burying the earlier tracks again: we will return to unbury them later.

By now you have created a clearly visible "dike" that slopes up toward an altitude of about 10m (9m above the ground). This dike is easier to see if you **vary the sun's illumination angle: press + or -** to sweep the sun across the sky and change the shadows. You can also see it well in wire-frame mode (press W to toggle it on or off), as the next figure shows (it already includes other tracks that we will lay in a moment.




The end of the track is now heading North again.

**Save your work, after deselecting any selected track or objects!**

You have learned an important command: **to get an inside view of the terrain in RE's wire-frame mode, press W**. (Press W again to go back to normal view.)

Now we **bring the track gently back to the horizontal**, by using several short 10m straight sections, as follows:

- select the A1t10mstsrt.s track section;
- press F5;
- place the 10m section at the end of the curving track, with a slope of 2.4°;
- press F4;
- press Y;
- repeat the last 4 steps, but with slopes of 1.8, 1.2 and 0.6°.

The next track section will be laid horizontally, so we don't need another 10m section with a 0° slope.

We next **make a horizontal right turn**, using three of the same curved sections that we used earlier, as follows:

- select the A1tEndPnt10dRgt.s track section again (it should show up in the larger Placement box);
- press F5;
- place the section at the end of the track;
- press F4;
- press Y;
- repeat the last three steps twice more.

Now go back and **unbury the earlier tracks** again, starting from the highest track section and working your way down the hill:

- press F2;
- select a track section;
- press Y;
- repeat the last two steps for the next lower track.

The next picture shows how this area may look when the route is finished: the train is just climbing through the right turn, and will level off on the short straight stretch, before turning horizontally to the right. The "dike" is no longer visible, due to the surrounding hills: later we will come back and create these hills and after that we will add all the other items that bring the scene to "life".

**Save your work, after deselecting any selected track or objects!**

We next **insert a short horizontal spur to the right**: that is a piece of track which branches off the mainline using a switch, and then ends (maybe at a factory). Do the following:

- select the track section A1tPnt10dRgtMnl.s (it is the right-handed counterpart of the left-handed switch that you used in the departure station);
- press F5;
- place the switch at the end of your curved track: it should be oriented properly to turn off to the right;
- press Y;
- select a 50m straight track section: A1t50mstrt.s;
- place it at the right exit of your switch;
- press Y;
- repeat the last two steps a few times to extend the spur line; (you may insert curved track sections also, but don't turn too close toward the mainline, because the mainline will start sloping down and needs space for the terrain to adjust);
- select the buffer: A1tUSBuffer.s;
- place it at the end of the spur;
- press Y.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again.**

Next, we will continue building the mainline as it slopes back down to near sea level.

**To continue building our First Route, jump to the next blue box.**

### 3.3.6 SLOPING AND BANKING A TRACK SECTION

A track section can be made to **slope up or down** (while it is selected). The slope is given in degrees in the Object window as "elev."; it can normally go up to 3° or down to -3° (a negative sign indicates a down slope). Steeper slopes and banked tracks are also possible.

CAUTION: The first laid track section will show slope values of opposite sign, even though the actual slope will be correct; also, when using the arrow keys together with the End key, the track not only slopes but also turns slightly to the right.

To **cause a normal slope**:

- select the track section;

- press F4;

- drag the mouse up to slope down, or drag it down to slope up (for a quick and rough slope); or press the up- or down-arrow keys while holding the End key (to fine-tune the slope).

Using the mouse changes the slope in steps of 0.15°. Using the arrow keys with the End key changes the slope in steps of 0.001°.

A 3° slope has a grade of 0.0524, or 5.24%, or 5.24 : 100, or 0.0524 : 1, or 1 : 19.08.

The grade (or gradient) is the tangent of the slope angle (angles are given in degrees, or sometimes in radians, or rarely in grads). For example, an angle of 3° gives a grade of tan (3°) = 0.0524 or 5.24% (rounded off from what the Windows Calculator gives you). This grade of 0.0524 means that the track rises by 0.0524m for every horizontal 1m, or 5.24m for every horizontal 100m, or 52.4ft for every horizontal 1000ft. For other small slope angles you can use simple proportionality: a 1° slope gives a grade of 0.0524 / 3 = 0.0175 or 1.75%, meaning a rise of 1.75m for every horizontal 100m. (With such small slope angles, you can ignore the fact that a sloped track section has a slightly shorter horizontal length than its actual length: for instance, a 100m track section sloped by 3° has a projected horizontal length of 99.9m.)

**To cancel a slope and go back to the original horizontal orientation, press O or click on "Reset rotation" in the Object window** (caution: the elev. value is however not reset to zero, a bug!).

IMPORTANT: You cannot "automatically" adjust a track section to slope according to the terrain. There is no key-press that will make the track follow the terrain slope (except for the very first track section, for which pressing N does give it the terrain's slope). You have to give the track a slope and make the terrain fit to it (by pressing Y).

NOTE: You cannot make existing tracks "automatically" follow changes in terrain height that you make after laying those tracks.

WARNING: Sloping curved track causes bad joints between tracks, especially for steep track with large turn angles: see section 3.3.12 for more details.

[NEW SINCE V1.106] You may want **to adjust the slope of an existing line of track sections**, connected at their ends to other track sections. For example, you may want to change its slope to better fit terrain. This is possible if you work in the same direction in which you first laid the tracks, adjusting one section at a time. The main difficulty will be that such changes create a gap between tracks that you will need to fill with dynamic track. Proceed as follows:

- back up your route;

- make sure that no track objects are attached to the track section you want to change (signals, mileposts, speed limits, etc., all of which have "handles" in the middle of the tracks): any track objects must be removed before you adjust the track itself;

- select one track section (press F2 then left-click on the track section);

- if this is the first track section that you want to adjust, change its slope; this will create a mismatch to the next track section;

- if this track section shows a gap to the previous track section, move it slightly (press F3 then press any arrow key): the track section should jump to connect to the previous track section; if it is too far to jump, left-click while pointing at the desired joint position, so it connects; if it still does not jump, it was originally reoriented by pressing T, so you should press T and then make the joint, before pressing T again;

- change its slope as before (it should have 0 slope after jumping to connect);

- repeat this procedure until you have adjusted all desired sections; the last section probably will need to be joined to the next unchanged section (see section 3.3.11).

To **cause a steeper slope** (this is only possible for connected track sections - an independent track section cannot be made to slope more steeply):

1) select the track section;
2) press F4;
3) drag the mouse down:  the track slopes up by 3°;
4) press T;
5) drag the mouse down:  the track slopes up by another 3°, to 6° (but the slope angle -labeled "elev." - does not reflect that increased angle:  it stays stuck at 3°;  more precisely, the "elev." can now vary again between 3° and -3°, but you should add 3° to this value, so it really ranges from 6° to 0°);
6) press T:  the track slopes down;
7) drag the mouse up:  the track slopes down by another 3°, to -9° (now the 3° to -3° range of "elev." really means -3° to -9°);
8) press T:  the track slopes down;
9) drag the mouse up:  the track slopes down by another 3°, to -12° (now the 3° to -3° range of "elev." really means -6° to -12°);
10) press T:  the track slopes up;
11) drag the mouse down:  the track slopes up by another 3°, to 15° (now the 3° to -3° range of "elev." really means 9° to 15°);
12) press T:  the track slopes up;
13) drag the mouse down:  the track slopes up by another 3°, to 18° (now the 3° to -3° range of "elev." really means 12° to 18°);
14) repeat steps 6-13.

Perform these steps until you have the desired slope (you can fine-tune the slope as before).  By pressing T once or twice at any time, you can change an up-slope into a down-slope, and vice versa.

There appears to be no limit on the slope angle:  the track may go vertical and even upside down (but the RE will flip it upside up) - even rollercoasters with vertical loops can be made!

It is also possible to **cause a track section to bank, as follows**:

1) select the track section;
2) press F4;
3) drag the mouse down:  the track slopes up by 3° (or any smaller angle);
4) press T repeatedly:  the sloping track will gradually turn into a non-sloping banked track, with a bank angle of 3° (or whatever the initial slope angle was), and then back again to a sloping non-banked track (while also switching each time between left and right turns);  you can cycle through these situations by continuing to press T.

Stop these steps when you have the desired bank and turn direction.  You may then fine-tune the slope by dragging the mouse up or down.

To create a steeper bank, first give the track section a steeper slope (as described further above), then apply the above steps.

The joints between track sections will show unrealistic up and down jumps between rails, but the trains don't seem to mind.

Unfortunately, trains do not bank on banked tracks, so this capability is of limited usefulness.

(It appears that pressing T has the following effect on a sloping curved track section: imagine the inclined plane that defines the initial track slope; pressing T rotates that plane around the vertical, keeping its inclination fixed, while the track section adjusts its inclination to that plane. A 10° curved track section takes 360/10 = 36 presses of T to return to its initial orientation, while a 20° curve requires 360/20 = 18 presses of T.)

NOTE: It is not known whether slopes steeper than 3° and banked tracks are "officially" accepted by Microsoft/Kuju, but they seem to work in MSTS.

### 3.3.7 ADJUSTING TERRAIN HEIGHT TO FIT THE TRACK

If the terrain is uneven or if you make the track slope up or down, you can adjust the terrain height and slope to match that of the track section: simply **press Y after laying or selecting each track section**. The terrain all along the track section should rise or fall to match the track bottom. In addition, terrain away from the track section will form a slope.

**The resulting terrain profile is governed by the numbers set in the Terrain window**: these can be changed to give a wide footprint in gentle terrain, or a narrow footprint in steep terrain. I like to use values of 45°/18m/45° (the default) in flat terrain, or 70°/12m/70° in steep terrain, like mountains - for more details, see section 3.5.4.

This will create a dike or trench around your selected track section when you press Y (press the back-quote ` to cancel your last terrain changes if you don't like them).

#### G. Building "First Route": snaking downhill with dynamic track

You will next continue the mainline as it descends the backside of the hill. Because of the need for sharp turns due to the particular shape of the (still imaginary) hill, we will need to make sharp snaky turns: dynamic tracks are convenient for that. See section 3.3.8 for details.

We start from the switch that led to the spur. **A gentle horizontal turn to the left** brings us away from the spur line, then we gradually dive into the descent, for which we will again use straight 10m sections, as follows:

- select the A1tEndPnt10dLft.s track section;
- press F5;
- place the curve at the straight exit of the switch: leave it horizontal;
- press Y;

- select the A1t10mstrt.s track section;
- press F5;
- place the section after the curve;
- slope it down -0.6° (press F4 and drag the mouse up to do that);
- press Y;
- repeat the last four steps with slopes of -1.2, -1.8 and -2.4°.

Next we **add a dynamic track section**:

- select the dynamic track piece (from the <Dynamic track> class of the Object selector);
- press F5;
- place the dynamic track at the end of your track: a short straight section appears, together with a green handle; its 2m length shows up in the Dynamic track window;
- reduce that 2m length to zero (that will let us start with a curve): click on the down-arrow that is just to the right of the 2.000 entry, to make that entry become 0.000;
- under "First curve", click the white box to Enable this first curve: you see a 25.000 appear for its Radius and a -0.010 appear for its Angle; the 25.000 means a 25m radius (which we will change), and the -0.010 angle means a left (negative) turn angle of -0.01 radians (which is about 0.6° to the left, and which we will also change);
- now decrease (make more negative) the turn angle to -0.5 radians (that is about 29° to the left): you do this by clicking on the down-arrow just to the right of the -0.010 entry; as you do this, you see the dynamic track grow into a sharp curve;
- next increase the radius to 80m: click on the up-arrow just to the right of the 25.000 entry; as you do this, the dynamic track increases its curvature;
- press F4;
- change the slope of the dynamic track to -3° (the same way you did it for all other track sections);
- press Y.

We next **add another similar dynamic track**, but make it turn to the right. The procedure will be just as for the previous left-turning section, but will require pressing T at the right time to make it reverse from left to right: the right time is after setting its turn angle and radius, but before making it slope, as highlighted below. Do the following:

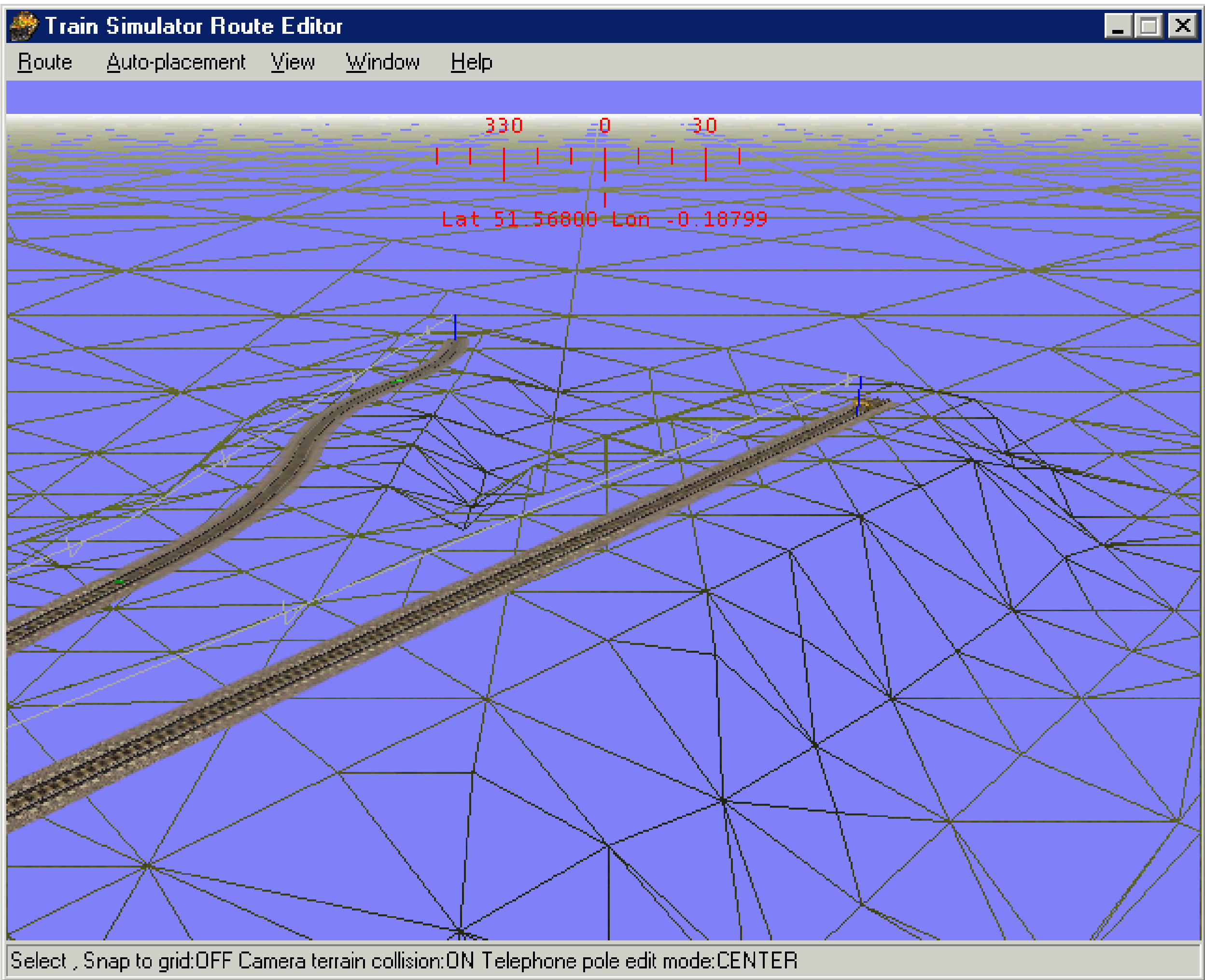
- press F5;
- place the dynamic track at the end of your tracks;
- reduce the 2m length to zero;
- click the white box to Enable the "First curve";
- decrease (make more negative) the turn angle to -0.5 radians (that is again about 29° to the left);
- increase the radius to 80m;
- **press T: this reverses the turn direction;**
- press F4;
- change the slope of the dynamic track to -3°;
- press Y.

CAUTION: **If you want to adjust the length or angle of a dynamic track after reversing it with T, you must first reverse it back to a left turn by pressing T again!**

CAUTION: **If you reverse a dynamic track by pressing T, it is reset to zero slope (even though the slope shown as "elev." remains non-zero!). So reset the slope after pressing T.**

Now **add a third curving dynamic track section, identical to the first**: left turn angle -0.5 radians, radius 80m, slope -3°.

As a result, we will have a snaky track that descends close to the original ground level, as shown at left in the next figure using wire-frame (the spur is visible in the foreground):




The next view shows how this same area may look later on.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again.**

Next we will create a siding.

**To continue building our First Route, jump to the next blue box.**

## H. Building "First Route": making a siding

You will next bring the mainline level again, before making it branch into a siding. A siding rejoins the mainline after some distance, so it will be important to get the two tracks to match up properly.

Starting from the end of the last curved dynamic track section, which slopes down at -3°, we **add four 10m straight segments that slope gradually less**, as follows:

- select the A1t10mstrt.s track section;
- press F5;
- place the section after the last track section;
- slope it down -2.4°;
- press Y;
- repeat the last three steps with slopes of -1.8, -1.2, and -0.6°.

Note that your last track section is about 3m above sea level (about 2m above the default ground). We will keep the track at that level for some distance now: that will require lifting the terrain under each new track section (by pressing Y each time).

Next we **add a 10° right-handed switch**, to start the siding:

- select A1tPnt10dRgtMnl.s;
- press F5;
- place it at the end of your track: it should be oriented correctly (branching off to the right), so we don't need to press T to reorient it;
- press Y.

**Add two 10m straight sections to the curved exit of the switch**, as part of the siding.

Next **add a section** A1tEndPnt10dRgt.s (reversed by pressing T) to bring the siding parallel to the mainline (remember to press Y after laying each track!). Thanks to the two 10m straight sections, the spacing between the siding and the mainline will be larger than between the tracks of our departing station.

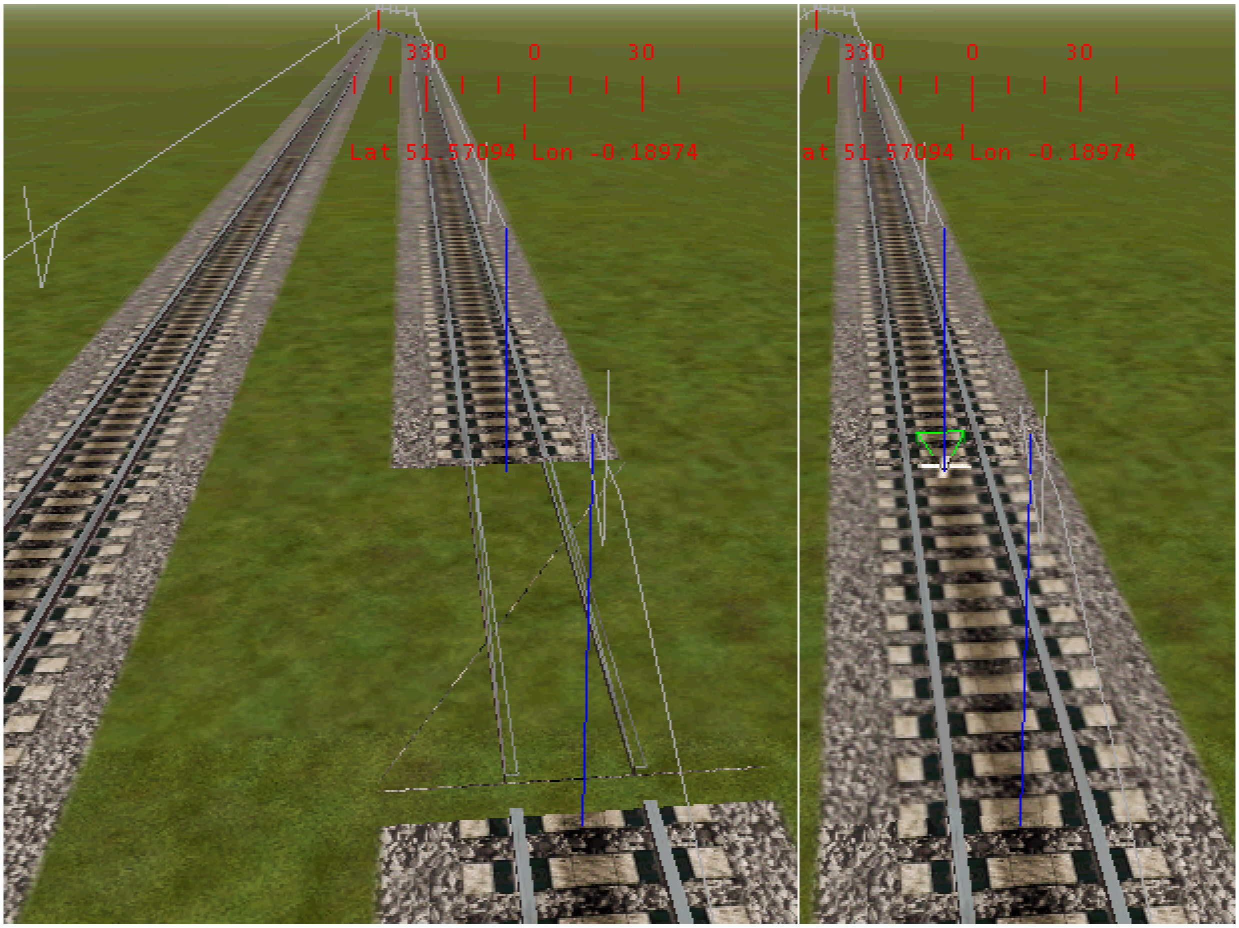
Now **continue the mainline, by adding two 100m straight sections to it** (press Y each time).

Next add **another 100m straight section** of a special kind: one **with a water trough** (for steam locomotives to pick up water while moving). For this use the A1t100mstrtwtr.s track section. Then press Y.

**Save your work, after deselecting any selected track or objects!**

We next **build the far end of the siding and back up** through the siding to its near end:

- add a switch of type A1tPnt10dLftMnl.s to the far end of the mainline track: as was the case back in the station, we need to reorient this switch, by pressing T twice;
- press Y;
- working backwards from that switch, add 2 straight 10m sections (equivalent to those you laid at the beginning of the siding);
- add a curve of type A1tEndPnt10dLft.s: press T so it comes back parallel to the mainline (and press Y);
- going backward, add two 100m straight sections;
- add two more 10m sections to try to close the gap: there remains a gap of less than one meter, as shown at bottom center in the next figure!




No standard track section can be used to fill this gap, even if we remove other track sections: **we need a straight dynamic track piece to fill the gap**! However a dynamic track must have a length of at least 2m: so we must remove the last 10m section and fill in the gap from there. That is shown in the right part of the figure above. To obtain this result, do the following:

- press F2;
- select the last 10m track section (the one shown in wire-frame above);
- press Delete;
- find and select a Dynamic track section;
- press F5;
- place it at the far end of the gap (where the + symbol is visible in the figure); (note that you could place it at the near end of the gap as well; in fact, there is no particular need to build the siding backward as we did: forward works equally well!);
- a short straight section appears (not as wire frame, because dynamic tracks do not show up as wire frame when selected), together with its green handle (as a wire frame handle while the dynamic track is selected!); its 2m length shows up in the Dynamic track window;
- increase that 2m length until you fill the gap (click on the up-arrow to the right of the 2.000 entry to get a near-closing of the gap, then fine-tune by clicking on the second up-arrow to its right): I find that 10.61m fills the gap well, as shown in the right part of the figure above; you

can move the camera for a close-up view to do the fine-tuning (remember to use the camera's fine-scale motion by pressing End while you move it);
- press Y.

If you now deselect the dynamic track, you should see its two blue poles disappear, and the gray/white overhead lines join up across the gap:  this indicates that MSTS has accepted the joint between the tracks.

You have now completed a siding.  Let us add a bit of straight single-track mainline after the siding:

- add a 100m straight track section after the switch.

This siding may ultimately look like this:

**Save your work, after deselecting any selected track or objects!  It would also be good to back up your FirstRoute folder again.  And you might as well reboot your PC after that.**

By now, you may be anxious to try out your route!  In fact, it is highly advisable to test a new route each time it changes significantly.  Next we will create an Activity to test your First Route.

**To continue building our First Route, jump to the next blue box.**

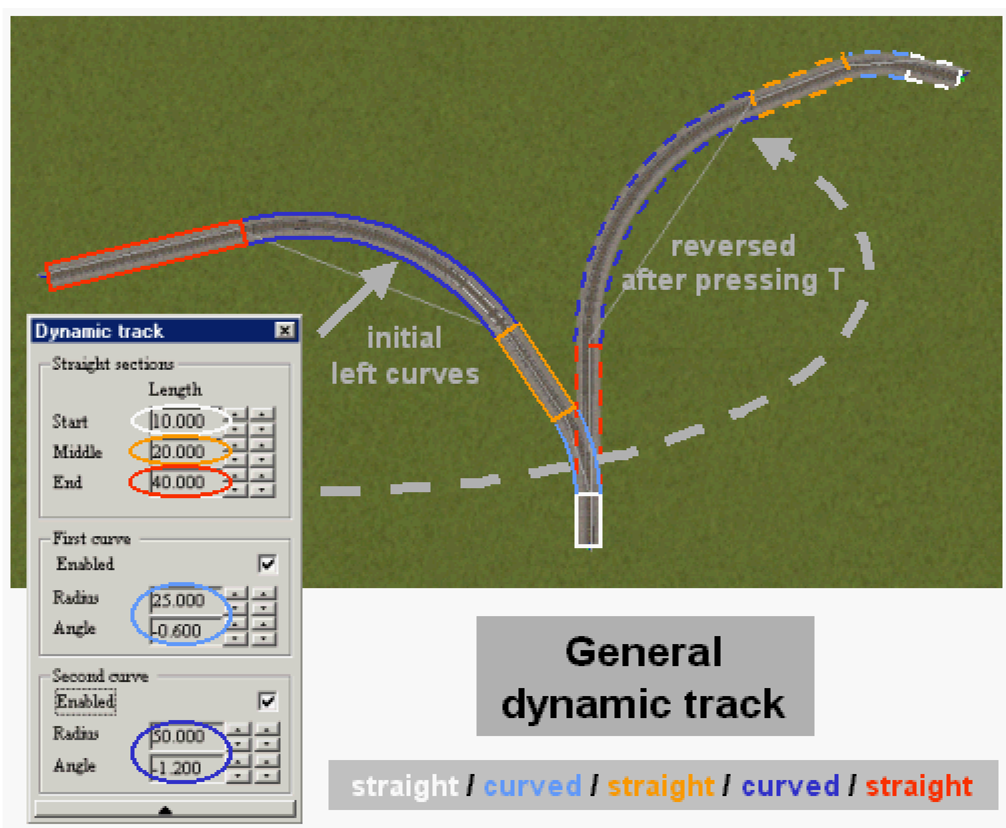
## 3.3.8 DYNAMIC TRACKS

Dynamic track sections are **useful when you need special turn radii and/or turn angles, or special lengths** not provided by the standard track sections. They are also essential when joining two tracks coming together: they can be fine-tuned to often produce a visually perfect fit between meeting tracks.

A dynamic track section is extremely flexible: it is **composed of up to 3 straight and 2 curved segments**. The straight segments have independently variable and fine-tunable lengths, while the curved segments have independently variable and fine-tunable radii and turn angles.

The most common use of dynamic tracks only utilizes a single straight segment or a single curved segment: these are illustrated further below.

The next figure shows the most general and complex dynamic track: it has 5 segments (here color-coded); the figure also includes the "reversed" case with right-hand turns (obtained by pressing T, see below). The view is almost straight down:



Unfortunately, **the two curved segments must turn in the same direction** (both to the left or both to the right, not one to the left and one to the right; if you want opposite turns, use two separate dynamic tracks, one turning left and the other turning right).

You can **reverse the turn direction (from left to right, or from right to left) by pressing T**, if the dynamic track section is selected. However, this does more that reverse the turns: **this also exchanges the ends of the track section** (the green handle goes to the other end), so that a straight/curved section becomes a curved/straight section: see the illustration above, where the white end of the initial track becomes the dashed white end of the reversed track, and similarly for the red end. You can reverse back to the original left turns by pressing T again (if you have deselected the track, first select it again - the green handle is now at the other end! - then press T to reverse it back to left turns).

Another limitation of dynamic track sections is that they are **not available as tunnels**.

Turn angles are given in radians, not degrees! (This is very odd, since degrees would be so much more convenient and precise! For example, there is no way to specify a 90° or 180° turn with radians, unless you could use an infinite number of digits.)

A few useful angles are the following (except for 180° = pi = π, the angles are rounded off to 3 digits after the decimal point, as you would use them in MSTS; I got pi = π from the Windows Calculator: just press on the "pi" button):

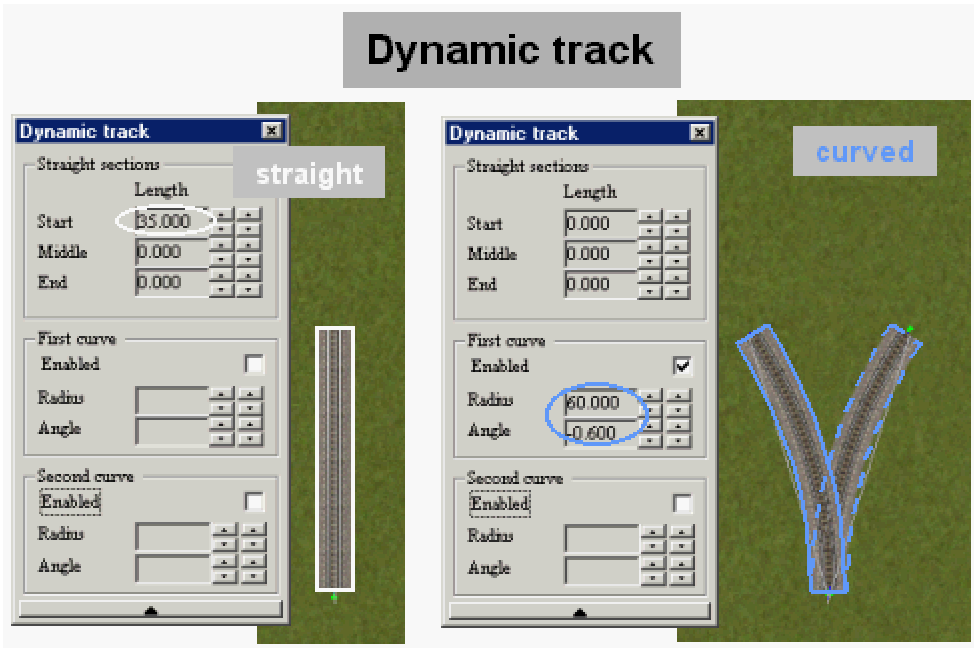
1° = 0.018 radians
2.5° = 0.044 radians
5° = 0.087 radians
10° = 0.175 radians
20° = 0.349 radians
30° = 0.524 radians
45° = 0.785 radians
60° = 1.047 radians
90° = 1.571 radians
120° = 2.094 radians
180° = 3.1415926535897932384626433832795… = pi = π radians

The 5 segments of a dynamic track are ordered as straight/curved/straight/curved/straight, as illustrated above. But they need not all be present. As a result, **you can make many combinations of segments in a single dynamic track section**; together with the option to reverse the tracks section (by pressing T), you have 18 combinations! (They are illustrated in the diagrams that follow, together with the Dynamic track window that defines them, and together with the dashed reversed track obtained by pressing T.)

**The 18 different types of dynamic tracks sections are the following:**

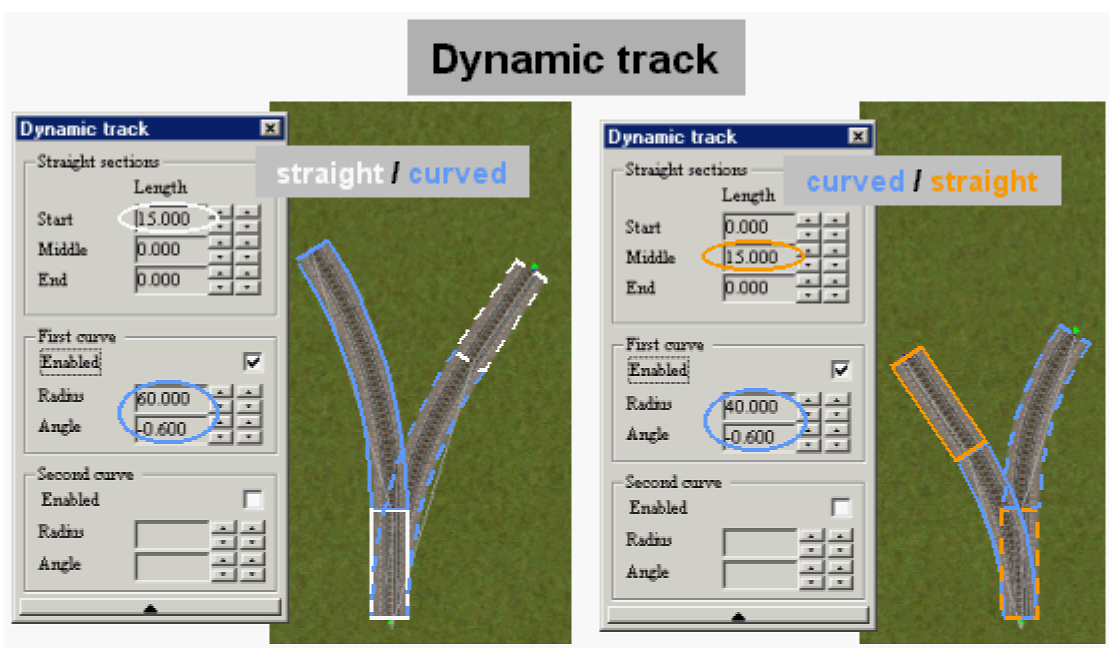
- **straight** (pressing T puts the green handle at the other end of this track section;  this is rarely useful);

- **curved**;

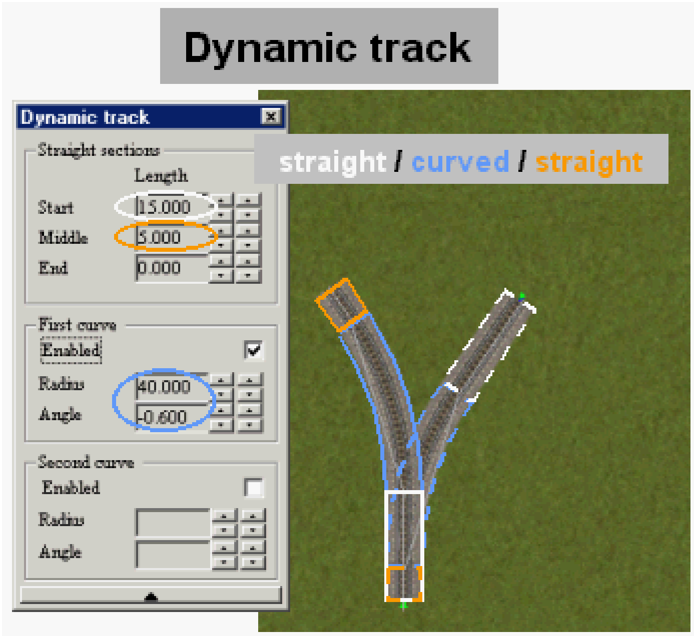



- **straight/curved**;

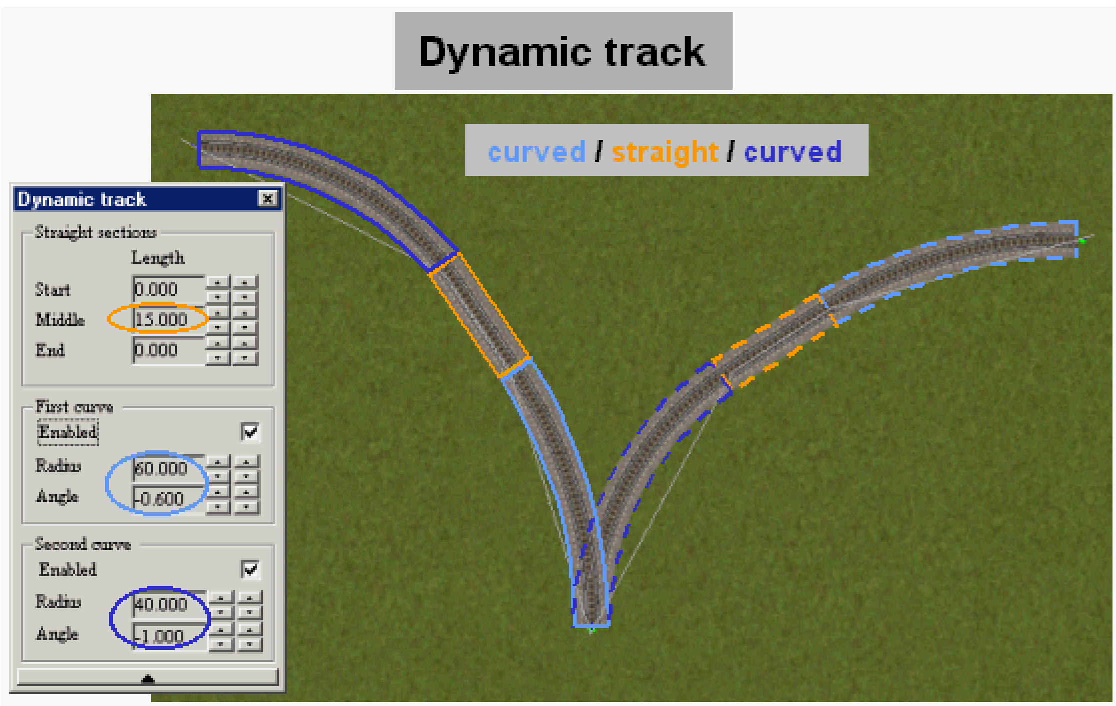
- **curved/straight**;



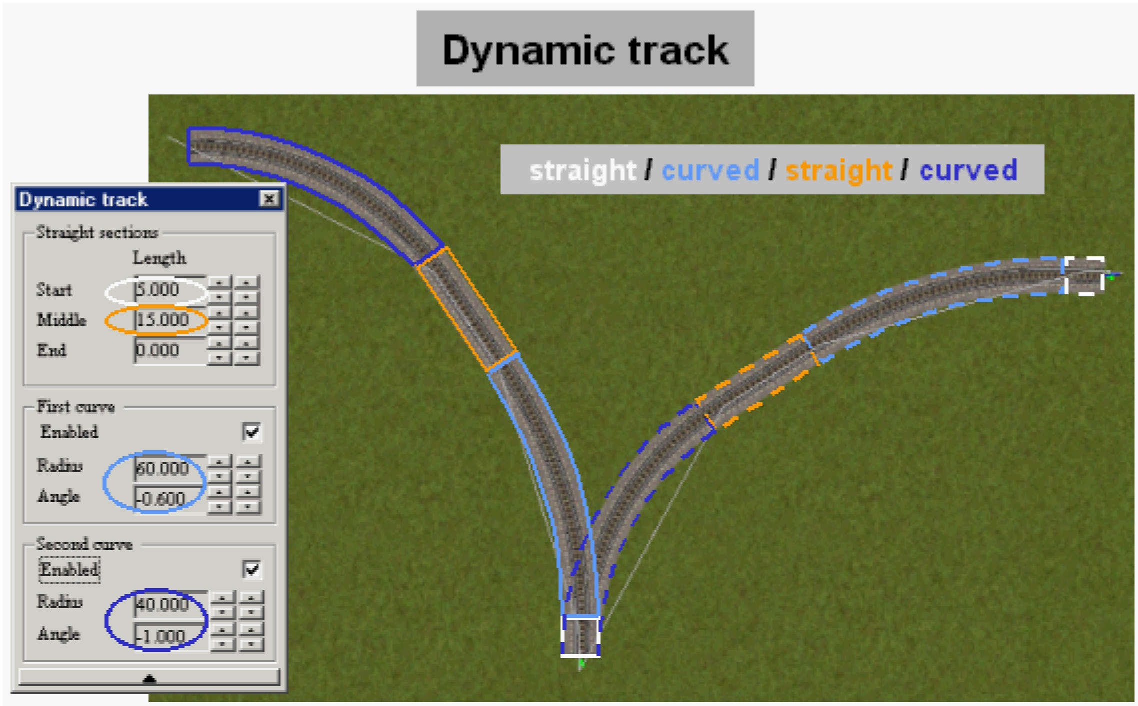
- **straight/curved/straight**;



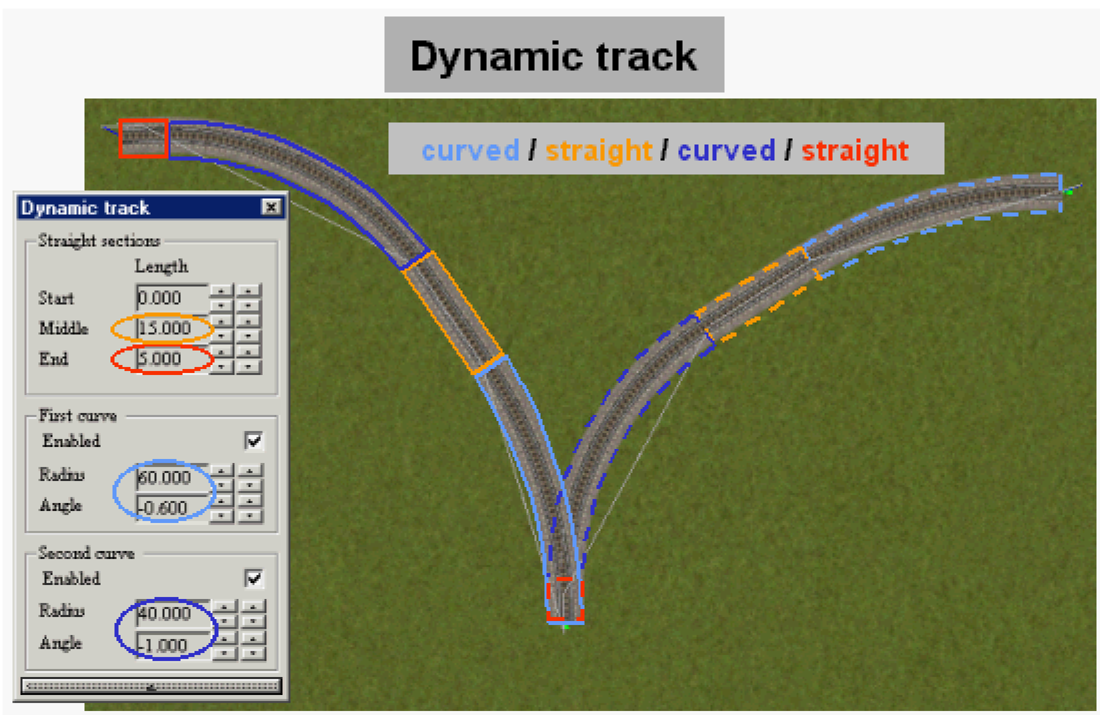

- **curved/straight/curved**;




- **straight/curved/straight/curved**;




- **curved/straight/curved/straight**;




- **straight/curved/straight/curved/straight** (see the illustration higher up, showing a general dynamic track section).

NOTES:

- to start with a curved segment, give the initial straight segment a length of 0m;

- to include a curved segment, you must enable it (click in the check box);

- any intermediate segment (such as the straight segment in a curved/straight/curved section) must have non-zero length;

- a non-zero straight segment has a minimum length of 2m, and a maximum length of 200m;

- an enabled curved segment has a minimum radius of 25m, and a maximum radius of 2000m;

- a curved segment has a minimum turn angle of 0.01 radians (about 0.6°), and a maximum turn angle of 1.57 radians (just under 90°);  the angles are negative to reflect the left turns!

The whole dynamic track section (with up to 5 segments) can be made to **slope up or down** like any normal track section.

WARNING:  The sloping rotation occurs around the beginning of the first segment.  For example, if the dynamic track turns twice (to the left) by 90° and comes back so that its end is lined up with its beginning (like a U turn), then sloping this track will raise or lower the round part of the U, but leave both ends of the U at the same level!  If the first half of this track slopes up, the second half will slope down.  If you want the second half to slope up, you must use a separate second dynamic track section.

WARNING:  **After you reverse the turn direction to right turns, don't modify the segments, unless you first press T** to reverse it back to left turns (otherwise, the track will disconnect and become useless).  To further modify the segments, you should reverse back to left turns, make your modifications, and then reverse again to right turns to see the result.

NOTE:  **To select a dynamic track section, you must left-click on the green "handle" at one of its ends**;  in fact, you must point at it so it turns from green to black and then click (this may require zooming in and even viewing it from another direction).  When a dynamic track section is selected, its green handle shows as wire frame, but the track itself remains normal:  it never shows as wire frame.

WARNING:  Placing dynamic tracks near water may cause varying (and even flashing) water colors, from blue to white and back to blue!

## I. Building "First Route": testing the route with an Activity

You will next create an Activity that allows you to drive on and test your First Route. This requires using the Activity Editor. For more details see sections 4 and 5.

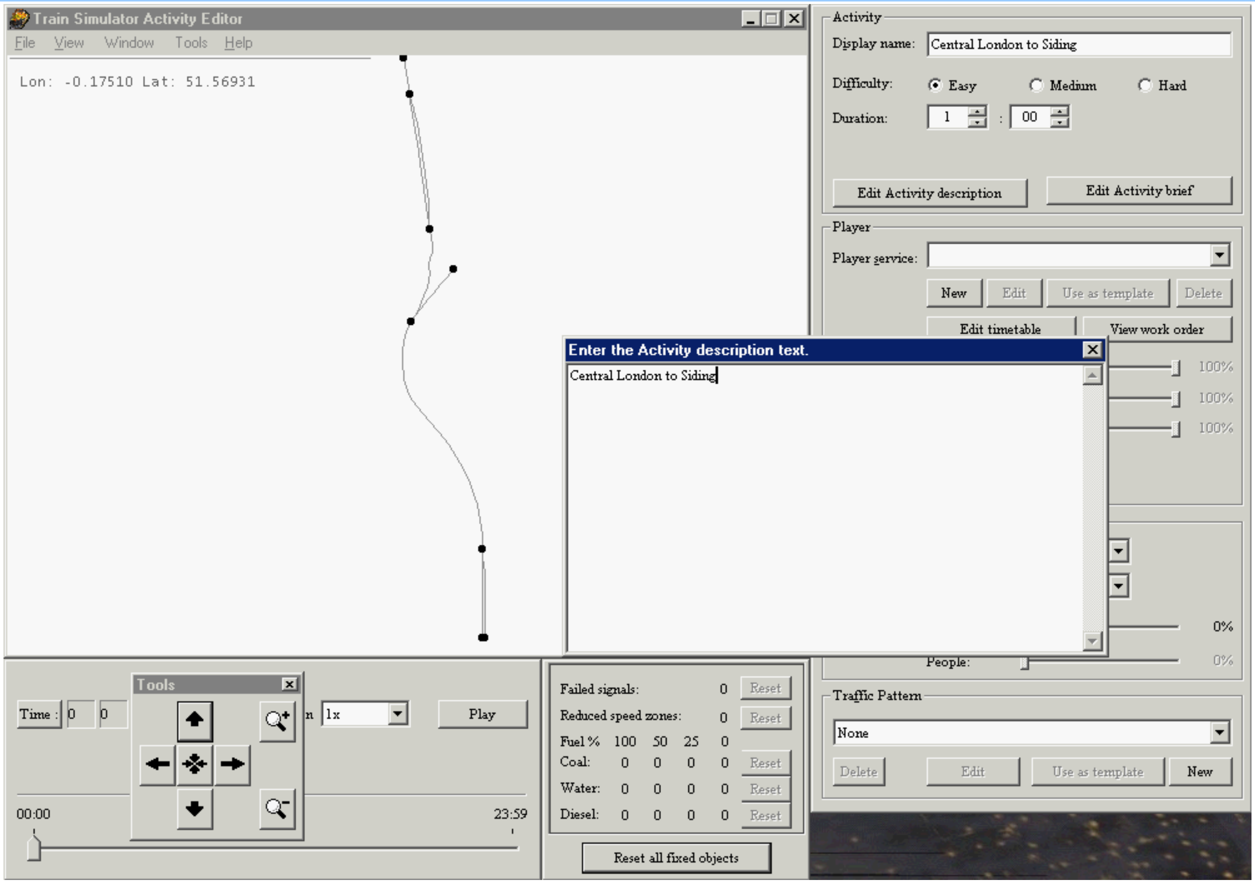
**Each time you make significant changes to the route, you should test it** by running a train over it in MSTS: try out the switches and any loops or wyes that you may have laid. If you delay testing until much later, and find your route to fail, it will be much harder to find why it fails!

**Running a train and testing a route can only be done after at least one Activity has been created for that route.**

An Activity places a train at a preselected spot on the route, ready to roll; it is created with the Activity Editor. The Activity Editor can also create a whole scenario for a trip, but for testing we only need to set up a train ready to start.

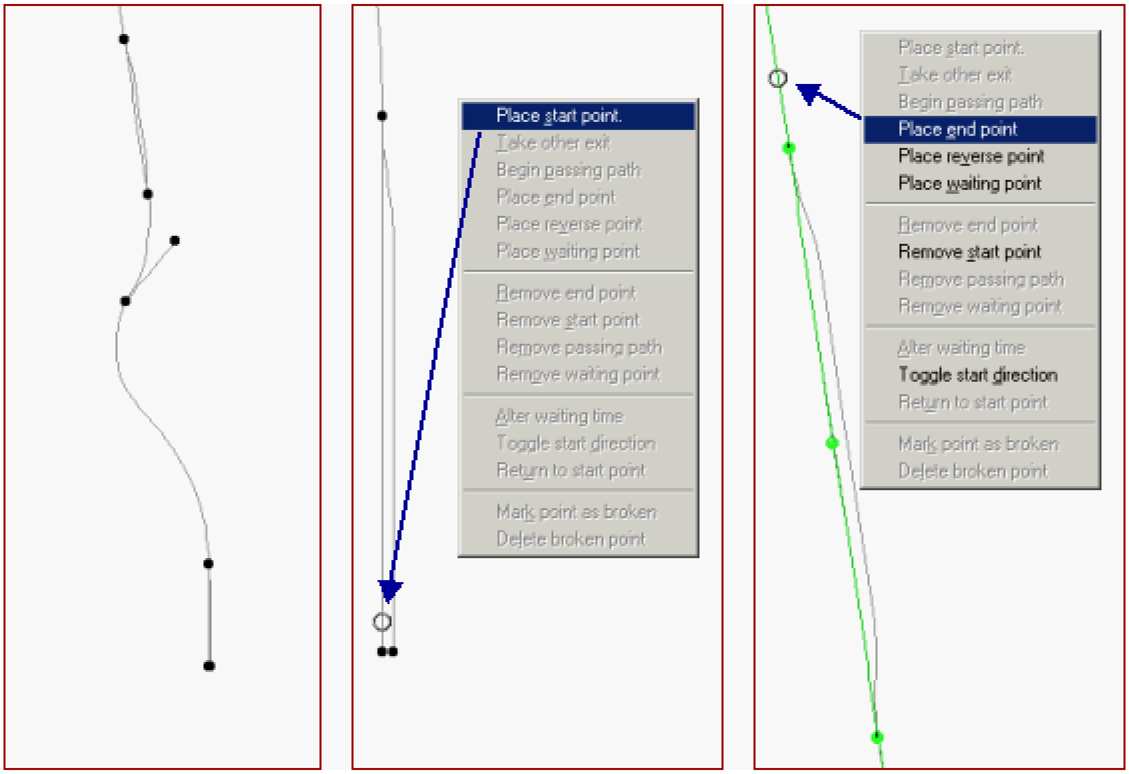
**Do the following:**

- exit from Route Editor;
- (if you already created activities earlier, delete them: delete all files in the Activities, Paths, and Services subfolders of the FirstRoute folder; you can do this with Windows Explorer or My Computer; the reason for this deletion is that existing activities become invalid whenever you add, remove or change a switch, or whenever you connect or break up tracks, including replacing tracks);
- start Editors and Tools;
- select Activity Editor: several windows open up;
- select **File** - **New**…;
- using the drop-down box, find and select your route (First Route), then click **OK**;
- type in the Activity display name "Central London to Siding" (without the quotes "");
- copy that name "Central London to Siding" to the clipboard (press Ctrl-C) for later pasting, then click **OK**: the large window now displays a map of your route (North points up), as shown in the figure below;
- to the right of Duration, increase the first zero to 1 hour (any non-zero duration is fine - it will have no effect);
- click on Edit Activity description and paste "Central London to Siding" in its box (without the quotes "", of course, using Ctrl-V), as shown here:




- do the same with Edit Activity brief;
- click on New under the blank Player service box:  this opens the Service editor;
- paste "Central London to Siding" again in the top two boxes of the Service editor;
- open the drop-down box, and choose an easy consist (train) for testing, such as "2 x GP38-2";
- click on New under the blank Path box;
- paste "Central London to Siding" for the Path name, and click OK;
- paste "Central London to Siding" for the Path display name, and click OK;
- the Path editor opens:  enter "Central London" for the Starting location, and "Siding" for the Ending location (still without the quotes "");
- shift the Path editor window away from the map, if it lies over the map;
- on the map, the mouse pointer is a circle:  you will use it to define the start point of your "path", namely where on the route the train should start in MSTS;
- zoom in to your departure station (which is at the bottom of the map):  you can drag the map around with the mouse, and enlarge or shrink it by dragging the mouse up or down while pressing its right button (you may also use the Tools window);
- point where you want the tail of the train to start, then right-click:  this opens a pop-up menu, shown in the center part of the figure below;
- click on "Place start point":  the circle becomes blue+green, while a second blue+red circle shows up, with a green connecting line;
- right-click again on the start point and click on "Toggle start direction":  the second circle disappears and the green line goes North along your route, where you want to go;

- now drag your map to follow the green line: make sure the green line does not turn into your spur (if it does, right-click on the small green circle at the spur switch and click on "Take other exit");
- continue dragging the map until you see the straight track section North of your siding;
- point and right-click on that straight section and click on Place end point, as shown in the right part of the figure below: the circle should turn blue+red, and the green line should stop there;




- click on Leave path editor;
- answer Yes to "Do you want to save path changes?";
- press OK on the Service editor;
- exit the Activity Editor: click on File - Save As…;
- paste "Central London to Siding" in the box for the File name;
- answer Yes to "Do you want to exit Train Simulator?".

Now **start MSTS, and Drive a Train**. You should see First Route in the Routes list: select it, then select Explore Route and choose any locomotive, consist, time, season and weather that you wish. (Don't use a locomotive alone: bad tracks can break a coupler or split up a consist at a switch: you would not detect these "symptoms" of bad tracks with a locomotive alone.) Then click on the Start arrow at bottom right.

NOTE: You may also select "Central London to Siding" in the Activities list, but this will allow fewer testing options, because an Activity freezes some switches.

Drive your train along your route, trying out every switch, spur, siding, and other option that you have. Look out for uneven tracks and for bad track joints. Worse problems, such as a break of couplers or a split of your consist, are more obvious. So will a freeze or crash of MSTS.

If you have any of those problems, something is wrong in your tracks. Note that a problem in one area can be due to a track error far away! For possible solutions to such problems, see section 4.6.

TIP: While driving in MSTS, you get a good close-up view of the track ahead by pressing Shift-1. And you can better control an outside view (first press 2, 3 or 4) by pressing Ctrl-Shift-9: by dragging the mouse while pressing its right button, you can scan the camera very freely.

Next we will add a high-speed dual-track mainline.

**To continue building our First Route, jump to the next blue box.**

## J. Building "First Route": switching to and from dual track, and placing a dual-track crossover

You will next split the single-track mainline into a high-speed dual-track mainline. You will then insert a crossover in it (this allows a train to switch from one track to the other).

Start from the end of the straight section that you laid after the siding. We want to **lay a left-handed switch to branch off to the left**. But we want this to be a high-speed transition, so we choose a switch **with a small turn angle**. (We choose 5°, although there is a 2.5° switch available as well; however, there is no corresponding 2.5° counter-curve for this switch!)

**Do the following:**

- select the A1tPnt5dLftMnl.s switch;
- press F5;
- place it after the straight section that you laid after the last siding switch;
- deselect the switch (press F2 and click elsewhere).

If you had not placed that straight section after the last siding switch, you would get the error message: **Invalid track placement--junction to junction placement invalid**; that tells you that you have tried to lay two switches end-to-end, without other tracks in between them! (There is one exception, which we will use soon: you may place two switches with one curved end against the other's curved end.) The **solution** is simple: **place another type of track section between the two switches**, with a length of at least 10m; make it 100m in this case. (If you followed the earlier instructions, you should already have a 100m straight track section after the last switch of the siding, so this error should not occur.

Now continue as follows:

- make sure the switch is oriented correctly (branching off to the left), so we don't need to press T to reorient it;
- add a section of A1tEndPnt5dLft.s to the curved exit of the switch, to bring the track parallel to the straight line;
- press T to flip the curve to the right;
- press Y.

**Save your work, after deselecting any selected track or objects!**

We will be laying "dual-track" sections after this: they are pairs of tracks that you lay in single pieces. So we need to first **make our two track ends line up**: that requires 70m of straight track coming from the straight exit of the switch (see section 3.3.9a for a diagram showing the 10° case):

- add a 50m straight section (A1t50mstrt.s) at the free exit of the switch;
- press Y;
- add two 10m straight sections (A1t10mstrt.s) after that;
- press Y.

Now we can **lay dual tracks**: their names are identified by A2t*.s. Start with a straight section, then lay a large-radius section curved to the right:

- select the dual-track section A2t250mstrt.s;
- place it at the end of your two tracks: you get two parallel tracks at once, and they should connect perfectly with your previous tracks (if not, try to press T);
- press Y;
- select the curved dual-track section A2t1000r10d;
- place it at the end of the existing dual tracks: it curves to the left, but we want to turn right;
- press T: this reverses the curve, but connects the two tracks incorrectly;
- press T again, and again: finally, the tracks curve to the right and connect properly!
- press Y.

What has happened here is that RE was cycling through all the different ways one could possibly orient and connect the tracks (even if some ways seem unreasonable at first sight, there could be other tracks around that would match up that way).

Next we **insert a crossover**, linking the two tracks from right to left, using 2.5° turns for high speed: see the illustration in section 3.3.9b, with the modifications indicated in the text there for the 2.5° turns. (For some reason, if you want to make a left-to-right crossover, you may not find the desired switches for the smaller turn angles.) This crossover will require 120m of straight sections on the left track:

- select switch A1tPnt2_5dLftMnl.s;
- place it at the end of the right track: it should be correctly oriented to branch to the left;

- press Y;
- place another copy of the same switch at the curved exit of the switch that you just laid;
- press T to orient it correctly (to form a parallel track);
- press Y;
- fill in the gap in the left track with 1 x 100m and 2 x 10m straight track sections (starting from either end): this should fill the gap perfectly; (press Y for each section!);
- now add 120m (1 x 100m and 2 x 10m) of straight track at the free exit of the switch on the right, to bring the dual tracks to line up again at their far end;
- at the end of the crossover, continue the curved dual-tracks with a section of A2t1000r10d.s, turned to the right (press T as needed; and press Y).

The following picture shows this crossover in a final version of the route, viewed from North to South.

We are going to **change back from a dual-track line to a single-track line**, in preparation for entering a tunnel. The dual-single track transition will be just the reverse of the single-dual track transition that we laid before (but there is no manual version of the 5° right-handed switch, so we use 10° turns):

- select A1tEndPnt10dRgt.s;
- place it at the end of the left track: it should curve to the right;
- select A1tPnt10dRgtMnl.s;
- place it at the end of the last-laid curve: press T twice to orient it correctly (with its straight part pointing back toward the right-hand mainline track), leaving a 20m gap to the right-hand mainline;
- fill the 20m gap with 2 x 10m straight sections: this should fit perfectly.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again.**

Next we will create a tunnel, and a spur that swings over that tunnel.

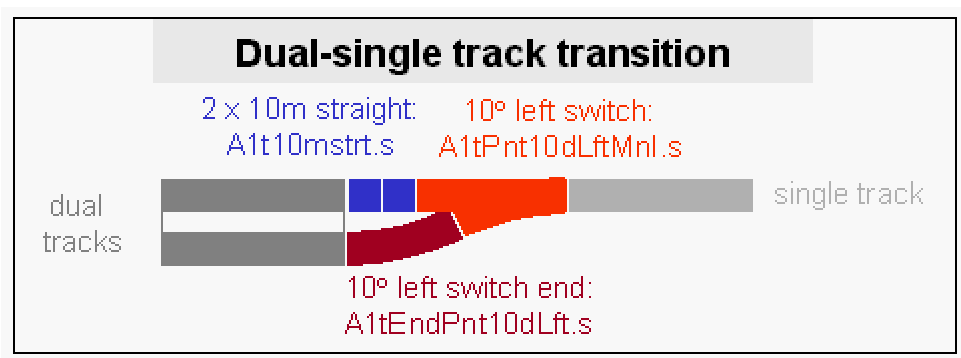
**To continue building our First Route, jump to the next blue box.**

### 3.3.9 SOME USEFUL TRACK COMBINATIONS

The following sections describe several track combinations that can be useful in a route.

#### 3.3.9a Dual-single track transition

The dual-single track transition **connects a single track to dual parallel tracks** that have the standard MSTS track spacing (5m). An example of a dual track section is A2t250mstrt.s (all names starting as A2t… are dual track sections).




This transition is illustrated here with a 10° switch. It can also be made with a 5° switch; in that case the 2 x 10m straight sections become 7 x 10m = 70m. (However, no manual 5° right-hand switch is available, so the train driver will not be able to operate it. Also, no 2.5° versions of A1tEndPnt10dLft.s and A1tEndPnt10dRgt.s are available, but you could tailor them yourself with dynamic track; again, no manual 2.5° right-hand switch is available.)

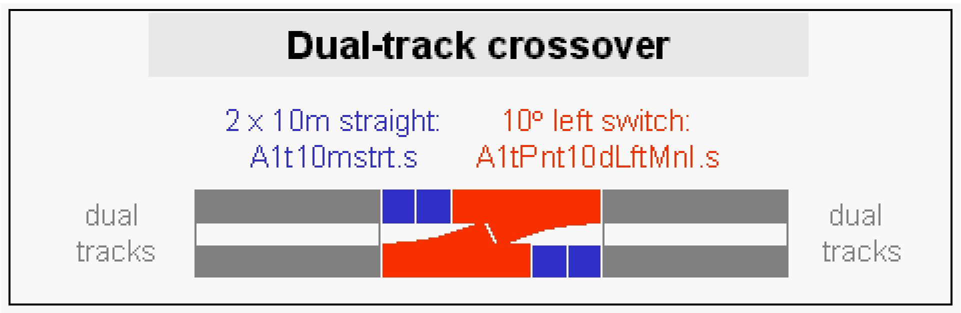
The total length of the transition (the colored part) is 60m and 150m with 10° and 5° switches, respectively.

The opposite turn direction (with a right-handed switch) is obtained with A1tPnt10dRgtMnl.s and A1tEndPnt10dRgt.s.

You can make a transition to **double tracks with a wider spacing** by including straight track sections between the switch and the curved track section (using one or more 10m straight sections, or a straight dynamic track section).

### 3.3.9b Dual-track crossover

The dual-track crossover is very similar to the dual-single track transition, using fewer types of track, but more of them. The reverse turn direction (to the right) is also similar (see section 3.3.9c). The dimensions are exactly the same as for the dual-track crossover.




This crossover can also be made with 5° and 2.5° switches: that requires 7 x 10m = 70m and 12 x 10m = 120m straight sections (blue), respectively.

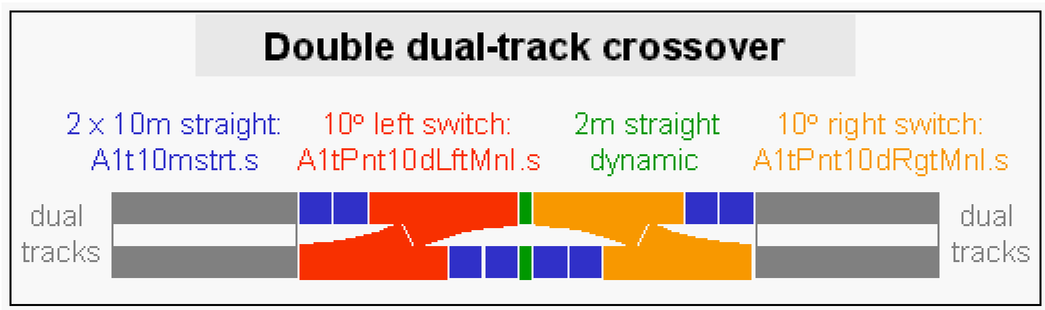
The total length of the crossover (the colored part) is 60, 150 and 250m with 10°, 5° and 2.5° switches, respectively.

Again you can accommodate double tracks with a wider spacing, by using straight sections between the two switches.

### 3.3.9c Double dual-track crossover

The double dual-track crossover puts two simpler dual-track crossovers back to back, to enable crossovers in both directions. Since the straight sections of two switches cannot be placed against each other, 2m straight dynamic track sections are inserted between them (they are the shortest available); of course, longer dual-track sections, and curves, could be inserted as well.

The dimensions are easily obtained from the simpler dual-track crossover.




NOTE: The 2m straight sections between switches can sometimes create problems; if so, increase their length to at least 10m.

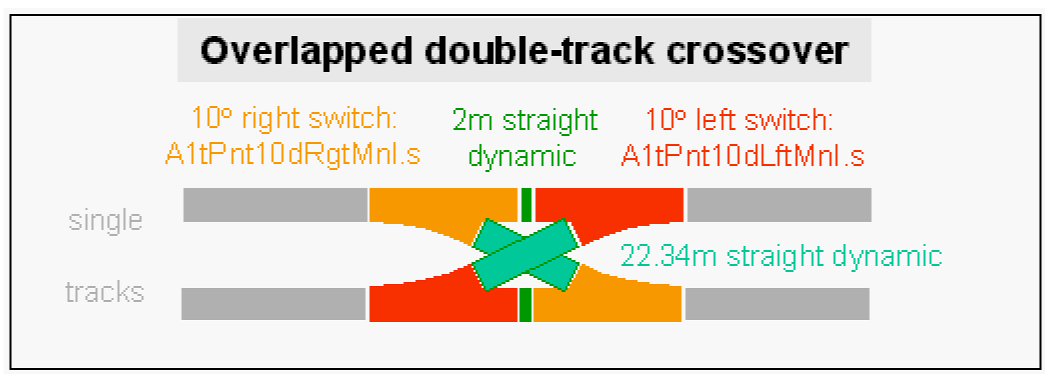
3.3.9d Overlapped double-track crossover

A double-track crossover allows two crossovers to share the same space. This is often found at the end of platform tracks, to allow a locomotive to slip out from the front of a train.

It is not possible with the standard MSTS track sections to build a cross-like crossover that fits between dual-track sections: the spacing between the parallel tracks will need to be larger than the 5m of dual tracks.

A simple way to build such a crossover is to overlap crossing track sections: we simply lay two dynamic tracks across each other, as shown in the next figure: this is not realistic, but works in MSTS.

The illustration below shows the "overlapped" crossover that gives the smallest possible track spacing with 10° switches (8.86m). The main tracks (shown in gray) have to be single tracks instead of dual tracks. Using straight connecting sections longer than 2m will only increase the track spacing (and would require more than 22.34m in the crossing).

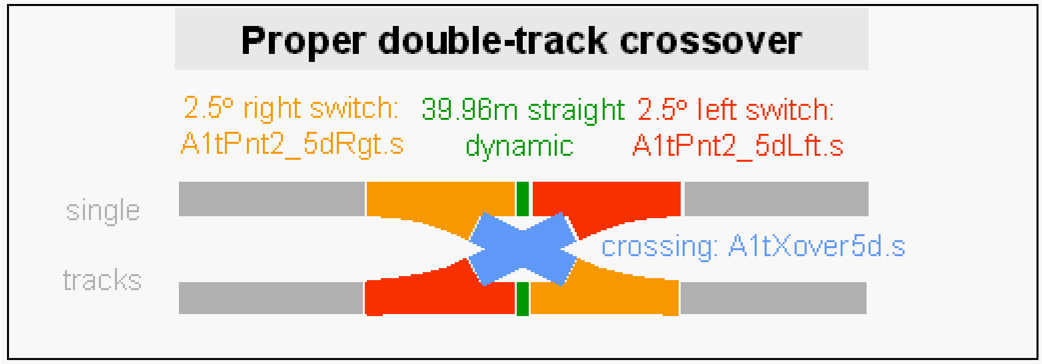



The crossed dynamic track sections must be shortened if you use 5° or 2.5° switches, to 12.04m or 12.01m, respectively (instead of 22.34m).

The track spacing (using 2m connecting sections and the indicated lengths for the crossed sections) is 8.86, 6.03 and 5.49m with 10°, 5° and 2.5° switches, respectively.

3.3.9e Proper double-track crossover

By using the crossing section A1tXover5d.s, it is possible to make a proper crossover, as illustrated in the figure below (the figure is very much shortened to better show the different parts).



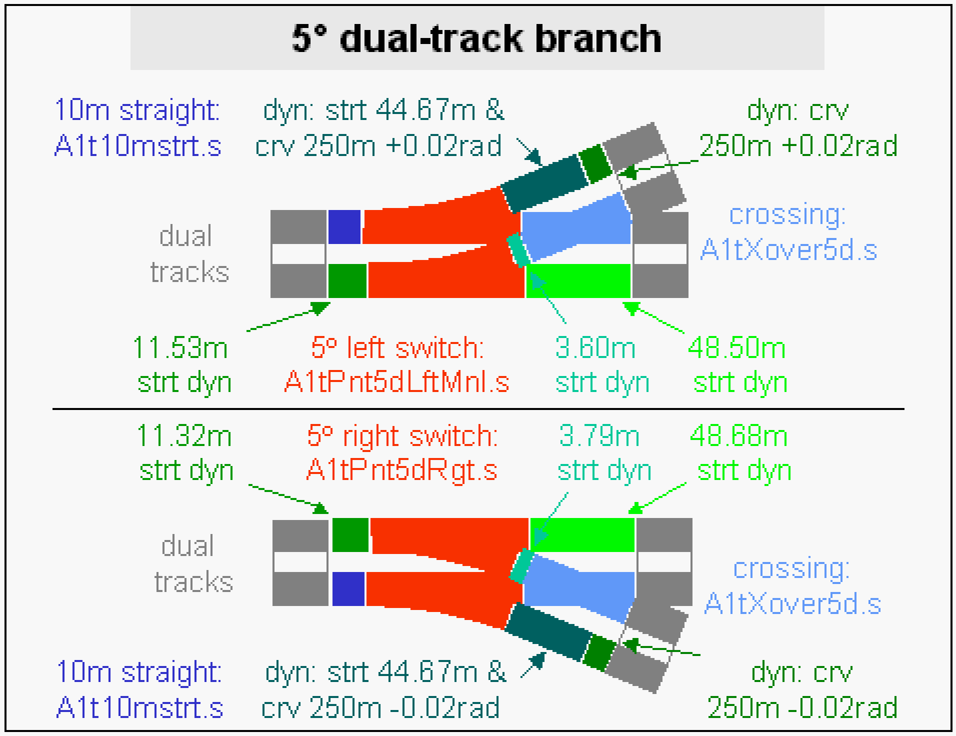

But this again cannot be done between standard dual tracks (5m apart). The two single tracks will be 7.17m apart (this will be increased if you insert equal straight sections between the switches and the crossing section, while lengthening the 39.96m straight dynamic tracks sections).

One drawback of this crossover is that no manual 2.5° right switch is available, so the train driver cannot switch from a left track to a right track.

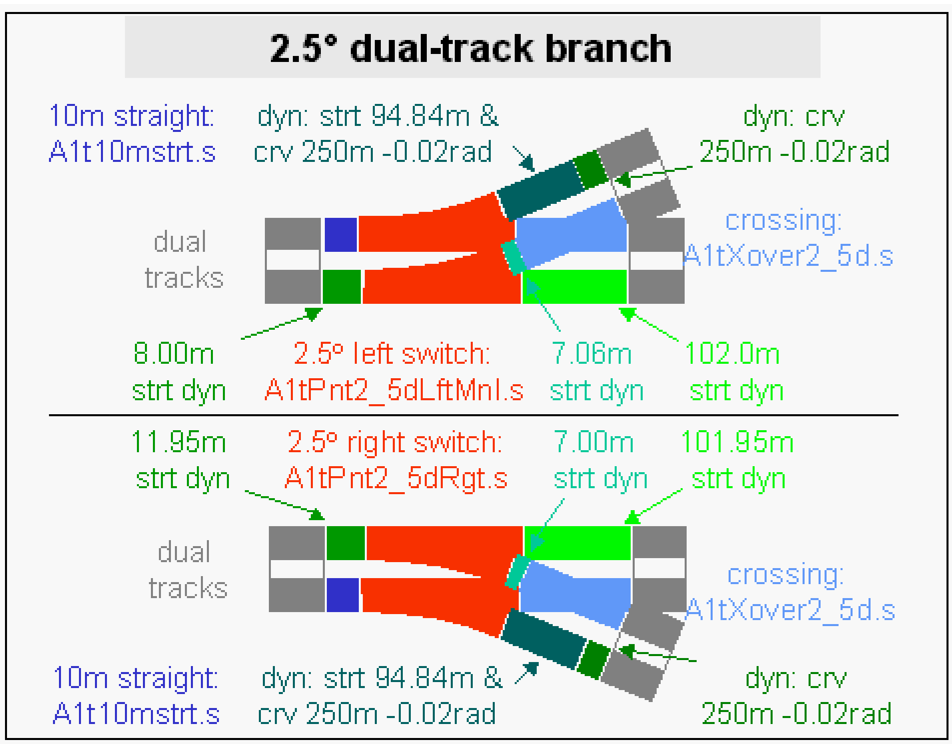
3.3.9f Dual-track branch

We can make a dual-track line branch off into another dual-track line by using a crossing section, 4 straight sections, a straight/curved dynamic track, and a curved dynamic track.

This is shown in the next two figures, based on 5° and 2.5° switches and crossings, respectively. (The figures are very much shortened to better show the different parts.)




CAUTION: The +0.02rad turn angles mean right-hand turns, obtained by pressing T!




This design is somewhat complicated by the desire to keep the standard 5m spacing in the branched dual-track line.

If you don't need the standard 5m dual-track spacing in the branched line, you can avoid some of the track sections shown in the figures. If you also don't need the 5m spacing in the main line, you can avoid most of the track sections. For example, coming from the upper left dual track (which should now be a single track section), you could lay the upper switch, then the crossing, then the second switch directly against the crossing, and finally extend the remaining 5 tracks with whatever spacings they have.

### 3.3.9g Overhead crossing

One track can pass over another rather than crossing on the same level. This has obvious advantages (two trains can cross at the same time, with no risk of collision and a lesser risk of derailment) as well as disadvantages (it takes much more space and additional structures). There are many ways to lay out an overhead crossing, which we need not discuss here.

The only question we address here is: how high should the overpass be? We can judge this from the height of the highest tunnel ceiling in MSTS, which is about 10.3m. This does not count the ceiling thickness.

So you should **allow at least 11-15m as a height difference between track levels**.

### 3.3.9h Spirals (helices)

One way to make a track climb a mountain is along a spiral (this is more properly called helix in geometry): the track makes one or more circles as it climbs on top of itself.

The main question here is: **how steep should the climb be in relation to the circle radius?**

Let's say that we want one circle to climb by 15m (as discussed in section 3.3.9g). The length of the track around the circle is 2.pi.R, where R is the radius. So the slope will be 15m / (2.pi.R), or about 2.4/R with R given in meters. The slope angle in degrees then becomes

**slope (degrees) = 143 / R (meters)**

For instance, with a 50m radius (very tight!) you can climb 15m in one circle with a slope of 143/50, which is just under 3°, the maximum slope normally available in MSTS. If you want a 1° slope, you will need a 143m radius.

So, if you use the A1tEndPnt10dLft.s or A1tEndPnt10dRgt.s track sections (which have a turn radius of about 172.7m), you will climb 15m in one circle with a slope of 0.83°.

### 3.3.9i Spreading dual tracks around an island platform

When a dual track line approaches a station with an island platform, the two tracks have to spread left and right around that platform.

Some of the platforms available in default MSTS routes have a width that allows using A1tEndPnt10dLft.s or A1tEndPnt10dRgt.s track sections:  that is by far the simplest situation. The easiest is then to look up the track layout in the default route that uses such a platform (especially the Innsbruck - St. Anton route).

But we also need a more general approach for platforms that have any width.  We discuss this next for two cases:  the symmetrical case, where both tracks spread equally to the left and right around the platform;  and the easier asymmetrical case, where one track stays straight, while the other swings around the platform.

Assumptions for the general case:

- the platform is not in a curve;
- it is approached by standard dual tracks;
- the terrain is level.

**Symmetrical solution**:  both tracks swing by the same amount to the left and right.

- extend the dual tracks beyond where platform will be, to help place the platform;

- place the platform over the tracks, orient it and center it;

- remove the dual tracks that are under and near the platform (where the tracks will have to deviate);

- place an unconnected single track section on the left side of the platform only (as seen in the direction of laying tracks), at a proper distance from the platform edge;

- connect the left dual track and the left platform track:  use a pair of curving dynamic track sections pointing at each other to fill the gap (see section 3.3.11d), so the two sections meet precisely (give both the same radius and about the same left turn angle:  try a radius of about 200m, a turn angle of about -0.15 radians;   adjust these as needed and include straight dynamic segments as needed);  make them meet roughly midway between the straight sections;  record their radii, angles and lengths;

- lay the symmetrical deviation to the right side of the platform:  place a pair of dynamic track sections leaving from the right dual track, one after the other, giving them the same dimensions as for the first two, but mirrored symmetrically (use T to reverse the turn of the first dynamic section from left to right, and remember that T exchanges the order of curved/straight segments within this dynamic track section);

- add straight track sections along the left and right platforms until the far end of the platform, making sure their total lengths are the same;

- make similar connections beyond the platform to the distant dual track, using again the same radii and turn angles, but adjusting differently the straight segments of the dynamic track (or delete the old dual tracks and place new ones);

- recenter the platform between the tracks.

**Simpler, asymmetrical solution**: one track stays straight, the other swings around the platform (easiest is the case where the left track swings to the left around the platform).

- extend one track (the right one, if possible) straight beyond where the platform will be: use single tracks first, then dual tracks well beyond the platform;

- place the platform along the single track;

- place an unconnected straight track section along the other side of the platform;

- connect the unconnected platform track to the existing dual track: use a pair of curving dynamic track sections for each connection (do this for the 2 connections, similarly to the case of the symmetrical solution).

### 3.3.10 MEASURING TRACK SPACING

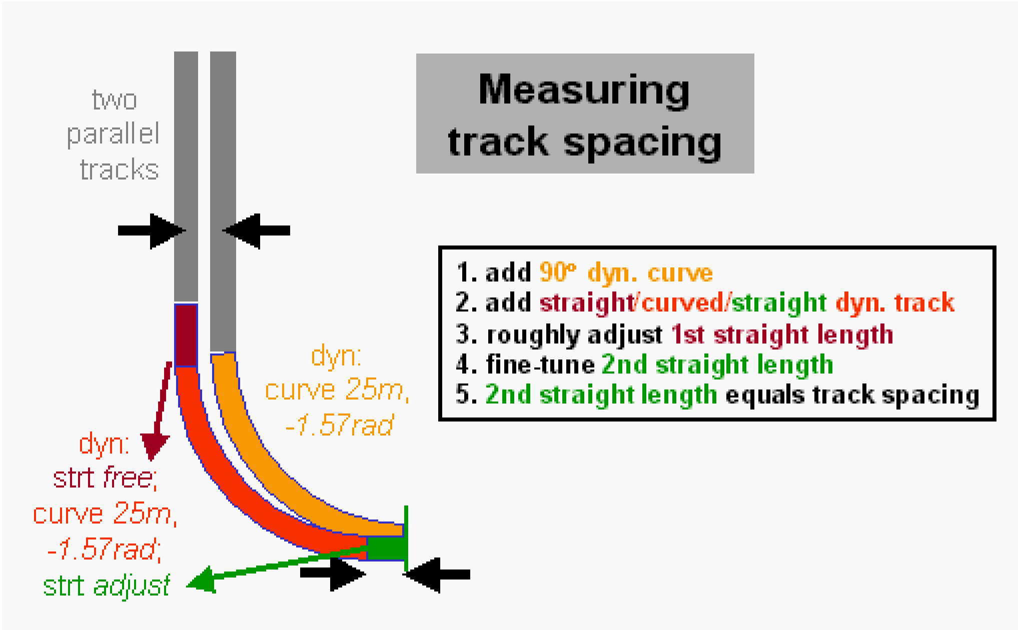
Sometimes you will want to know what is the distance between two parallel tracks.

**An important use of the track spacing is when you make two parallel tracks stay together through a curve**: to keep them equally spaced, you need to give the outer track a radius that is larger than that of the inner track by an amount equal to the track spacing (the difference in the radii will thus equal the track spacing).

In particular, you may ask what is **the spacing between the default dual tracks in MSTS**: the answer **is 4.985m**.

[NEW SINCE V1.106] One relatively simple, but less accurate, method is to lay a section of dynamic track across (at right angles to) the two tracks between which you want to measure the distance: you can then lengthen the dynamic track section until it fits the track-track distance, and read off its length in the dynamic track window.

The next figure gives you step-by-step instructions to measure a track spacing more accurately (defined as the distance between the centerlines of two parallel tracks, shown by the black arrow pair at left, also copied at bottom). The idea is that turning by 90° converts the track spacing into an offset in length, which you can measure by adjusting the length of a straight dynamic track segment (shown in green) to match up the two free track ends at the green line.




After having measured the track spacing, remove the two dynamic track sections.

NOTES:

- in step 3, you only need to bring the two dynamic track ends roughly near each other: in fact, avoid getting them too close together, so as to see the ends of both in step 4;

- the two parallel tracks need not end together: one may be longer than the other with an offset as shown in the figure; the first straight segment of the dynamic track takes care of that offset;

- the two parallel tracks need not be straight: but the offset between their ends should be straight for this method to work correctly;

- the precision of this measurement is better than 1 cm.

### 3.3.11 JOINING MEETING TRACKS

Very likely you will be faced with joining two tracks that come together. This happens for instance in a loop, where the looping track must rejoin the original straight track. It also happens in a wye, and in many other situations.

The standard track sections rarely have the right dimensions to join tracks, and MSTS offers no automatic snap-on in such cases: **you must use dynamic tracks and fine-tune them to fill the gap between tracks as well as you can**.

WARNING: Don't join tracks to form a closed circuit! And be careful with loops and wyes. (See Planning a Route.)

### 3.3.11a Using skewed track

This option is **not recommended**!

The A1t10mSkew.s track section puts a sudden small change of direction in your track by about 1° to the left. (Pressing T does not give a skew to the right, but only removes the change of direction, giving a straight section like A1t10mstrt.s.) A skew to the right is offered only by A2tskewRgt.s, but requires dual tracks.

A1t10mSkew.s offsets your track sideways by about 0.17m after 10m.

You could try to join tracks by a succession of sudden 1° left turns every 10m, which is very inelegant. However, you will still need dynamic track to fill in a final gap to join another track. You might as well use dynamic curved track instead of skewed track, as discussed next.

### 3.3.11b Joining level tracks

Joining tracks is a tricky operation. Let us first treat the case of horizontal tracks that are on the same level: since they are at the same altitude, we need not worry about bringing them to the same level.

We will assume that the tracks approach each other within about 200-400m. If the tracks approach too close to each other (how close depends on several factors), you may need to back off one or both of the end points, or use a smaller turn radius.

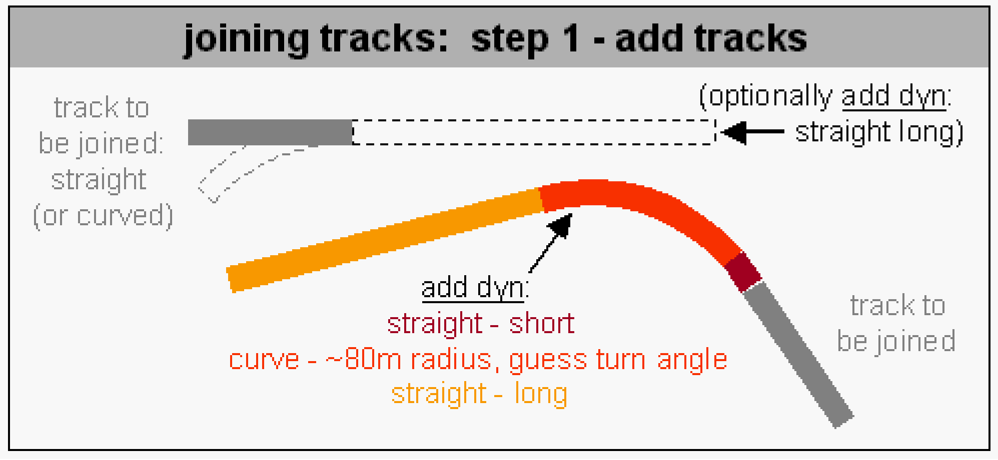
**We split the operation into the following major steps: 1) adding one or two track sections in the gap to be filled; 2) parallelizing these tracks; 3) aligning these tracks; 4) closing the gap.** But first, there is a choice to be made:

0) **Choose on which side of the gap to add a curved track**

You can add curved track from either side of the gap. From one side you will need to insert a left turn, while from the other side you will need to add a right turn. **It is much easier to add track from the side that requires a left turn.** In the example of the next figure, you would add a curved track from right to left, since that track layout then requires a left turn.

1) **Add one or two tracks in the gap**

We now add one or two track sections to produce something like this (the existing tracks to be joined are shown in gray at left and right - we will not change them):



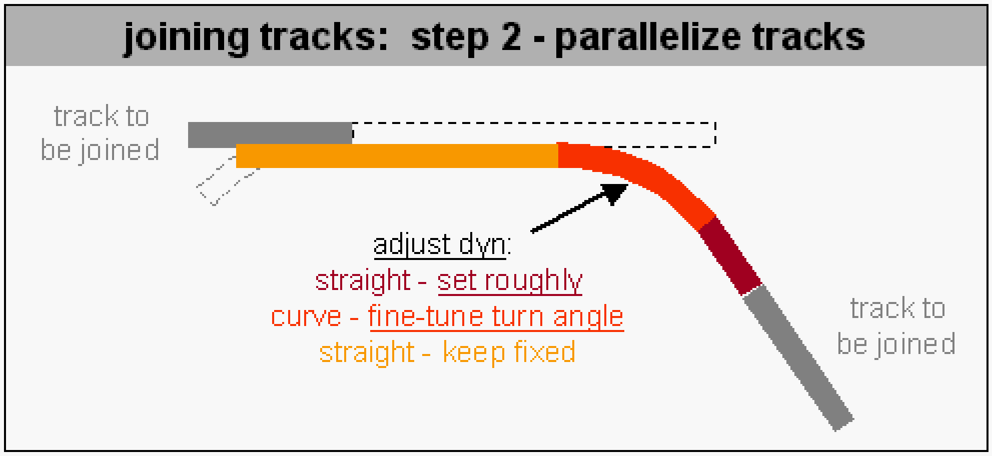

Start a curved dynamic track from one free track end (from the right in this example) and give it three segments: straight/curve/straight. Give the curved segment a radius that you like (I suggest at least 80m; you may change it later) and a turn angle that you guess roughly. If the curve should go to the left (as recommended above), give the first straight segment a 2m length, and the second straight segment an excessive length (so it forms a tail that can go well beyond the other side of the gap, as drawn in the figure). If the curve should go to the right, reverse the two lengths: long for the first segment, short for the second (then press T to reverse it; remember to press T again to reverse back for further fine-tuning).

If the required turn is more than 90°, you will need to add another curve: it could be a second curved segment in the already curved dynamic track section, or another dynamic track section, or regular curved track sections with fixed dimensions.

If the track at the other side of the gap is curved itself (shown dashed in the figure above), you can more easily visualize its orientation by adding (temporarily) a straight track segment from that side (it could be dynamical and you could actually keep it to make the final joint later): it is also shown dashed in the figure.

2) **Parallelize the tracks**

We now correctly orient the added curved track so its straight end becomes parallel with the track at the other end of the gap (they need not fit rail-to-rail yet), to achieve something like this:




What matters here is that the orange tail be parallel to the end of the gray track at upper left; the dashed straight track helps in orienting the orange tail.

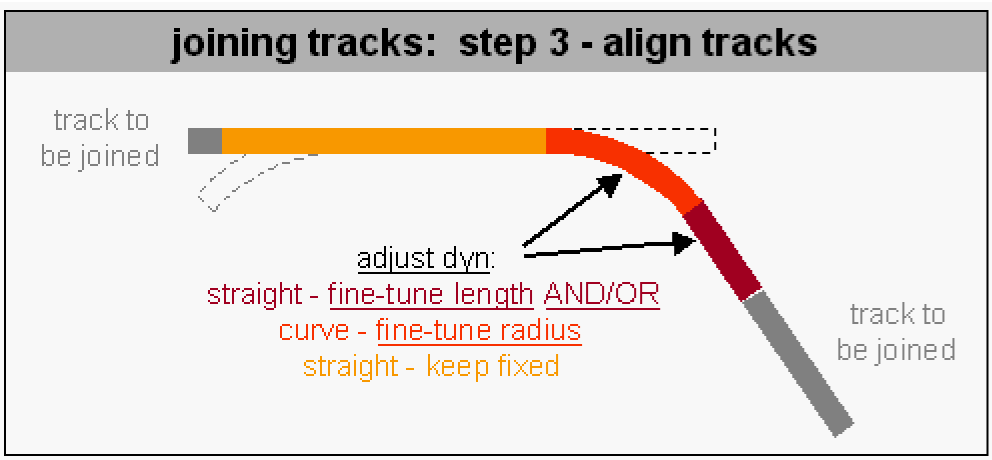
To do this, first place the RE camera at the other track's end and look toward the oncoming curved dynamic track; or take a bird's eye view from far above the tracks.

Now adjust the turn angle of the dynamic track's curved section until its long straight tail is parallel to the free track end (it will not yet align rail-to-rail): you only need to give that tail the same direction (orientation) as the free track end: you don't need great accuracy - a few degrees is good enough - just use your eyes to judge orientation!

If the tracks are too far from each other, bring them closer together by increasing the short (2m) straight section. (If that only increases the distance between them, your track end points are too close together to be joined with your chosen radius: back off one or both of the end points, or reduce the radius.)

3) **Align the tracks**

Now you can bring the tracks rail-on-rail, but without closing the gap perfectly yet, like this:



To do this, you have **three choices**:  you may increase the curved track radius;  or you may increase the length of the short (initially 2m) straight segment;  or you can do both.

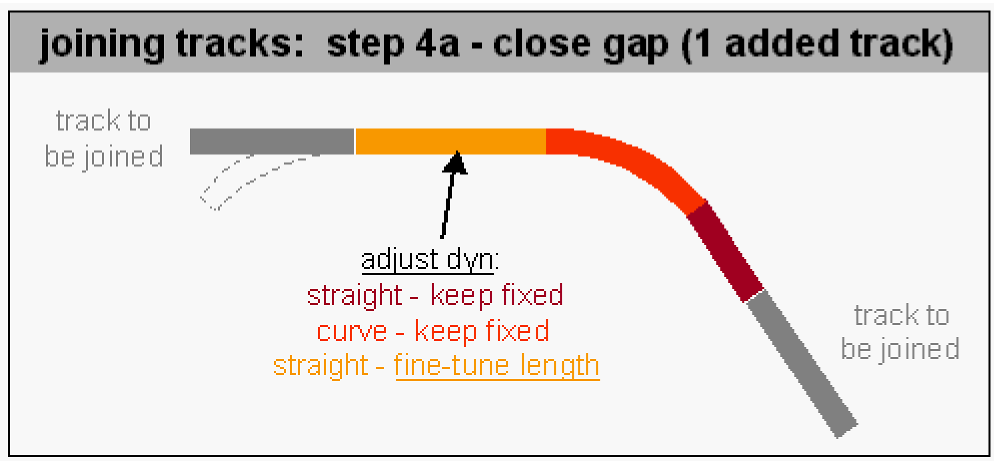
Adjust either the curve radius or the length of the short (initially 2m) straight segment to make the long tail pass right through the free track end, so the rails visually merge into each other (but let the long tail still stick out too far).  If the free end is a straight track, the long tail should now coincide almost perfectly along that whole straight track;  if the free end is a curved track, the long tail should be perfectly tangent to it and coincide with the straight (dashed) section you added to it at the beginning.

4)  **Close the gap**

All that is left to do now is to close the gap, by bringing the rails head-on.

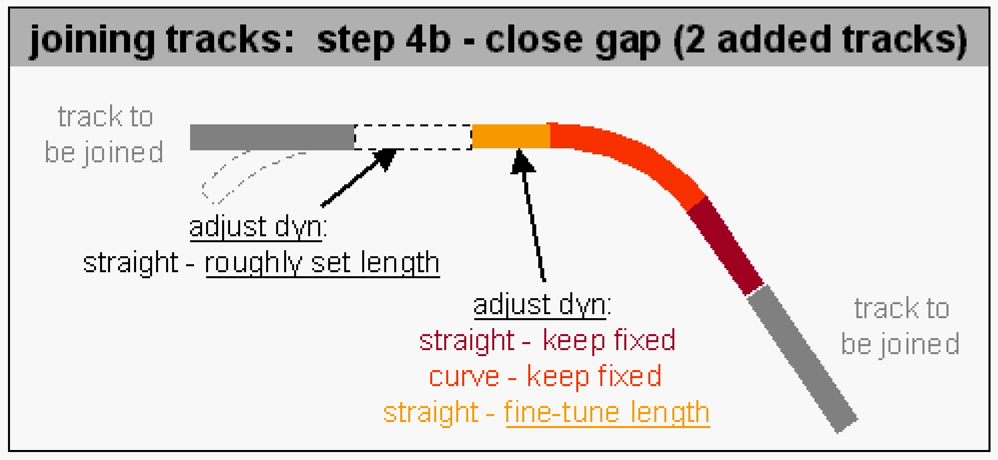
You have **two choices**.

First choice:  Delete any (temporary) straight section that you added to the far side of the gap (the track that was shown dashed in earlier figures).  That will result in something like this:



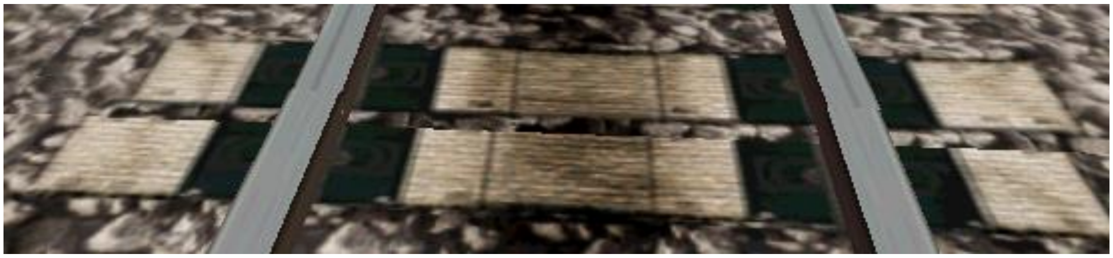

For this you simply shorten the long (orange) tail until it fits perfectly against the free track end.

Second choice:  Keep the straight (dashed) section that you added to the far side of the gap, shortening it, and using it as part of the joint (it gives you one more degree of freedom to fine-tune the closure of the gap, together with the length of the orange tail of the curved dynamic track), like so:




A mismatch of a few centimeters can be felt when driving over a joint, so be careful! For reference: a rail is roughly 10cm wide at the top.

The dynamic tracks in principle allow enough fine-tuning that you often can't see the joint even when zooming in with the MSTS camera. Here is an example of a fairly good joint, with a remaining sideways offset of less than 1cm (it closes a loop - see section 3.4.4 for making loops):



IMPORTANT: Once satisfied, deselect the dynamic track(s) and **check that the white lines above all tracks join perfectly** (the blue vertical lines at the track ends should disappear): this indicates that MSTS accepts your joint.

### 3.3.11c Joining non-level tracks

Joining non-level tracks is **very similar to joining level tracks, with the addition of one more fitting: the slopes of the tracks that meet**.

The procedure then is as follows:

0) **Choose on which side of the gap to add a curved track**

This is the same choice as with level tracks (see section 3.3.11b).

1) **Add two tracks in the gap**

Again, this is the same as for level tracks.

2) **Parallelize the tracks**

Do this also exactly as with level tracks. You can slope the tracks for better visual orientation (beware that each change can snap the track back to horizontal, even if the slope indication stays at non-zero!).

3) **Align the tracks**

Again, do this as with level tracks. But now you will probably not be able to get the rails to merge perfectly, because of the level differences. If you look straight down from above, you should be able to get a nearly perfect alignment.

4) **Adjust the slope**

Make the long curved dynamic track section slope to best approach the other side of the gap: with some luck, the tail will go right through the rail ends at the other side.

5) **Close the gap**

You have a choice (as with level joining): keep one or both added tracks.

Keeping both added sections gives an extra degree of freedom to close the gap. Make one of the two sections shorter so it leaves space for the other, then adjust the other to close the gap, while looking down from straight above (so the slopes don't matter).

If you keep only one added section, keep the curved dynamic track section and close the gap, again while looking straight down.

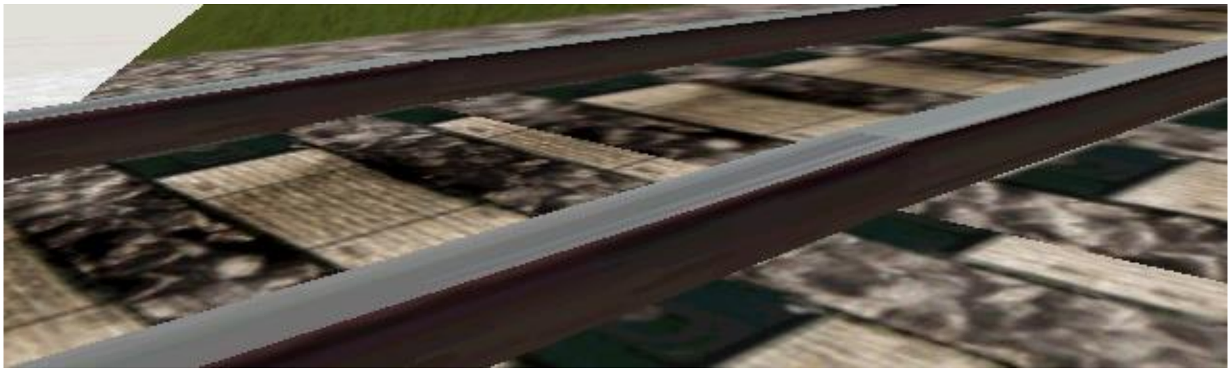
WARNING: **Adjusting the dynamic track lengths can reset their slopes to zero** (even if the slope indicator does not show zero)! So you need to check the slopes again.

6) **Check the slope(s)**

Looking from the side, again adjust the slope(s) of the one or two track sections to best join up.

You should also check that the gap is still closed correctly.

Here is an example of such a joint (it joins mountainous tracks coming from several kilometers apart):



TIP:  You get the finest vertical control by adjusting the slope of the shorter track section.

IMPORTANT:  Once satisfied, deselect the tracks and **check that the white lines above the tracks join up perfectly** (the blue vertical lines at the track ends should disappear):  that indicates that MSTS accepts your joint.

### 3.3.11d Joining offset parallel tracks

You may want to join two tracks that are already parallel to each other (they have the same orientation), but are offset sideways, so they "miss" each other.

**Using skewed track (not recommended)**:  For offsets of about 17cm (or multiples thereof) to the left, you could use the A1t10mSkew.s track section to correct the offset, if you don't mind a sudden change of direction by 1°.  But you probably will still need a dynamic track section to fill in the last gap.  For offsets that are a multiple of 17cm, use one A1t10mSkew.s track section followed by several A1t10mstrt.s sections.

**Using dynamic track (recommended)**:  You will need to **provide a mild left turn and a mild right turn, so as to make a very stretched-out "S" shape**.  This can be done with two identically curved dynamic track sections, starting from each free track end, heading toward each other and joining in the middle of the gap.

Start with a radius of perhaps 80m or more (200m or even more is better if you have the space).

To join these two identical dynamic track sections, after they are placed, use a similar procedure to that given in section 3.3.11b (the main difference is that you now make the same adjustments to both added dynamic tracks equally, jumping back and forth between the two):

1)  **Parallelize the ends**

To do this, include longer tails on both dynamic track sections, and adjust the turn angles of both sections until the two tails are oriented parallel to each other (they may still be offset sideways).

2)  **Align the tracks**

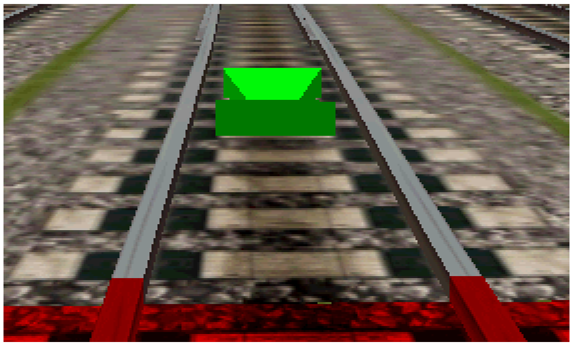
Now you can bring the two tails rail-on-rail, but without closing the gap perfectly yet.

Adjust the radii of the two dynamic track sections until the two tails fall on top of each other, rail for rail. (If this is not possible, maybe you need a longer gap between the track sections that you are trying to join. Or, if you have a long gap, it may help to restrict the "S" shape to a smaller part of the gap, using a straight segment for the rest.)

3) **Close the gap**

Close the gap by bringing the rails head-on: shorten both long tails until they meet perfectly midway.

Below is an illustration of such a joint for a very small offset of only about 10 cm (it is part of the 2.5° dual-track branch discussed in section 3.3.9f; note that the standard rail width in MSTS is about 15 cm). The two green handles of the two dynamic tracks mark the joint position (the midpoint of the "S"). To guide the eye, I have added in the background a pair of rails that are exactly lined up with the nearby red track: this shows how the track twists to the right by about a rail's width to join an offset parallel track farther away. The curve radius used here is 250m, and the turn angle 0.02 radians.



## 3.3.12 CHECKING FOR POOR JOINTS, ESPECIALLY IN GRADES

Very bad joints have a very bad effect: MSTS won't start your route, but it also won't tell you where the trouble lies! In that case, you may already be able spot trouble in the Activity Editor: the route map may show up trouble spots as dots at unexpected locations, away from switches and track ends.

Poor (as opposed to bad) joints are not directly spotted by AE or MSTS. The easiest way to check for poor joints is to **drive a train everywhere and look for the jolts that happen at poor joints**. Also, **throwing a switch** (press G or Shift-G) **can freeze MSTS as a result of a poor joint** (again not telling you where the problem lies).

**Try to fix bad and poor joints.** One way that sometimes works is to reload the route into RE while asking for the track database to be rebuilt (select the Advanced option and check the box). Otherwise, maybe adjusting a bad dynamic track section is sufficient, or simply deleting and reinstalling a poorly positioned track section will do.

One nasty joint caused me much trouble (it made MSTS freeze up when I threw a switch far away from the poor joint in question): I had a 250m-radius dual-track section turning 20° and sloping up a modest 1.05°; it was followed by another dual track section. The problem was that the slope combined with the 20° turn caused the two parallel tracks to end up at different levels (by tilting one slightly above the other), so that the next dual-track section could not match both tracks properly at the same time. The solution was to use two 10° turns instead of one 20° turn, so that the mismatch was only half as bad (and spread over two different joints).

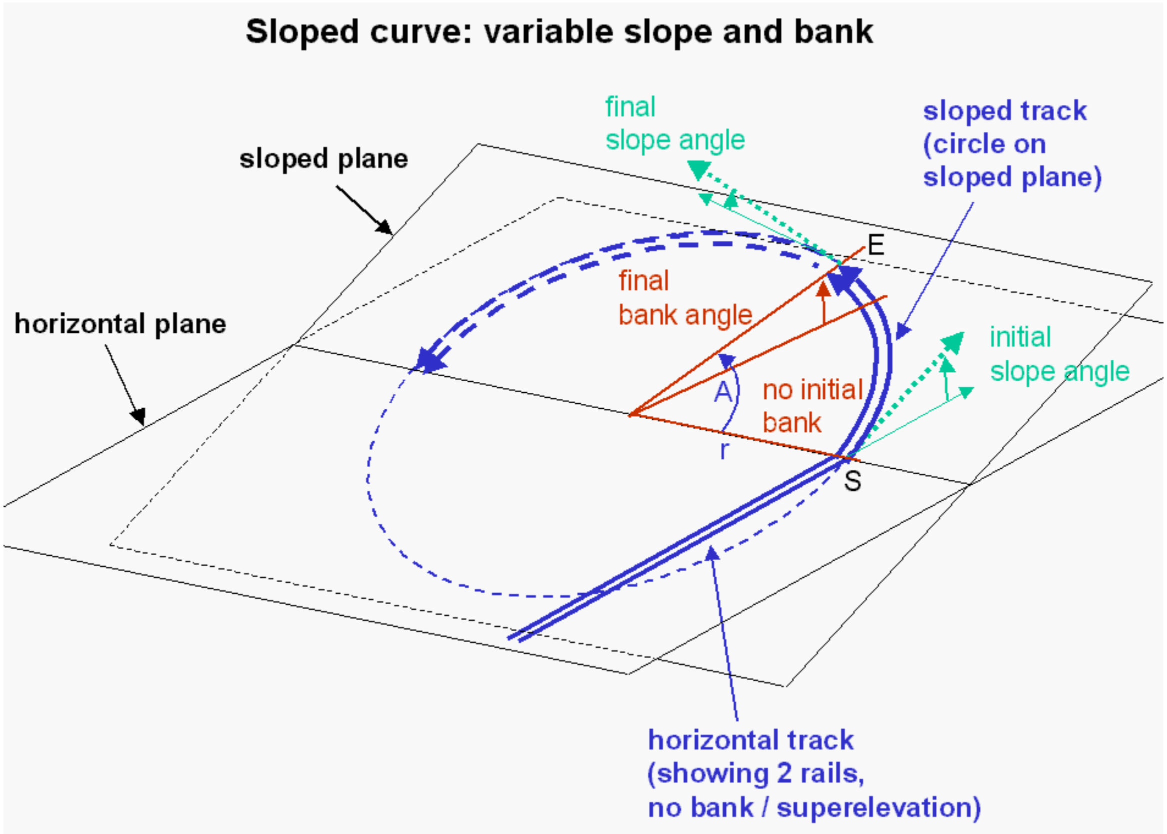
The lesson: **use <u>short</u> multi-track sections on slopes**.

TIP: While driving in MSTS, you get a good close-up view of the track ahead by pressing Shift-1. And you can better control an outside view (first press 2, 3 or 4) by pressing Ctrl-Shift-9: by dragging the mouse while pressing its right button, you can scan the camera very freely.

[NEW SINCE V2]
To better understand how tracks are joined together in sloped curves, it is important to understand how sloped curved tracks are shaped in MSTS: they are shaped as an arc of a circle on a sloped plane, as illustrated in the next figure. If you imagine the curve continuing around a complete circle, it will close the circle at the curve's starting point. A sloped curve starts with an initial slope set in the RE. However, as it curves the slope diminishes until it reaches zero after a turn of 90°; the slope then changes sign (becoming negative if it started positive, nd vice versa) and becomes steeper until a turn of 180° is completed, when it reaches the opposite of the initial slope.

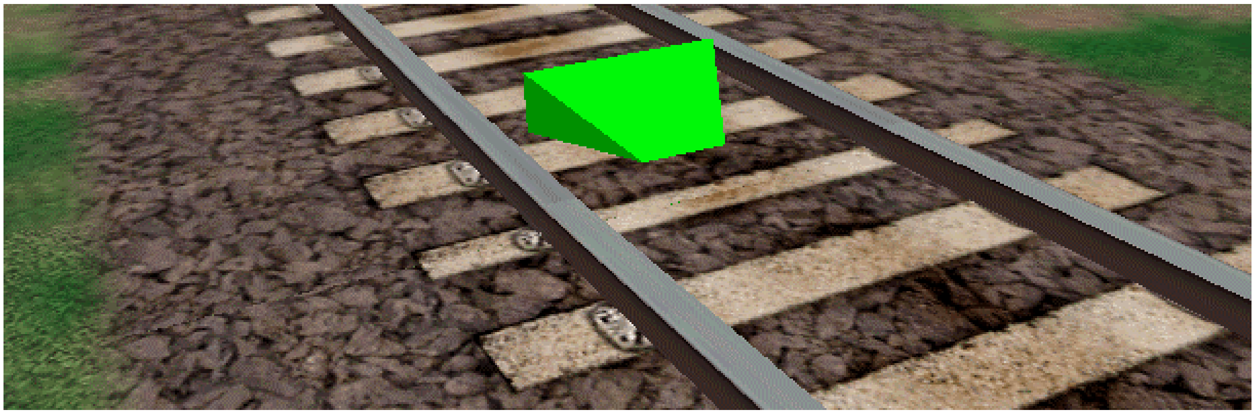
Another effect is important also: sloped regular (non-dynamic) curves also have a changing sideways "bank" (superelevation), as shown in the figure below. At its starting point the curve has zero bank by default (no superelevation); but as the track curves it starts to bank, reaching maximum bank after a 90° turn (at that point, where the slope itself has dropped to zero, the bank angle has grown to be equal to the initial slope); then the bank diminishes again, and reaches zero after a 180° turn. However this bank effect is different with dynamic track: dynamic curves keep a zero bank (superelevation) everywhere along the curve. For either kind of track, even when there is a bank (as on non-dynamic sloped curves), MSTS makes trains ride as if there were no bank: the track bank is ignored!

In fact, dynamic track has the special property that its bank is always zero at all points along the track (this suits the non-banking trains perfectly). This is true for any degree of slope, and, more surprisingly, even if you artificially apply a "bank" through its QDirection (using Object Rotator)!

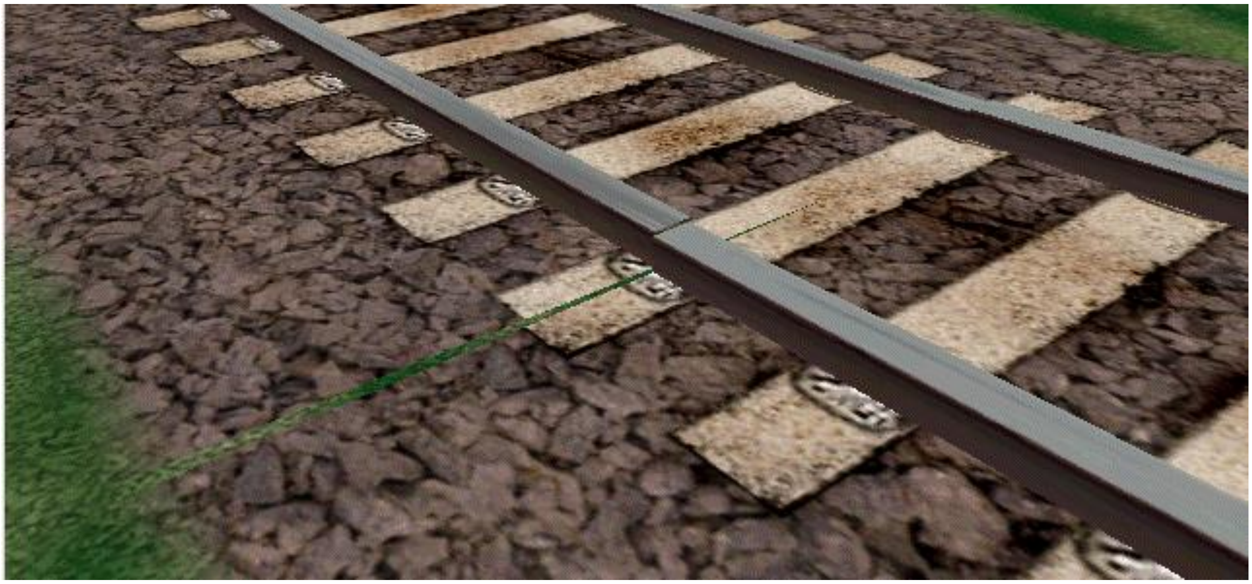



The importance of this becomes clear when you attach another track section at the end of a sloped curving track section, as illustrated in the following figures. By default, the new track section will start with zero bank, and whatever slope you give it: suppose you give it the same initial slope as the preceding track section.

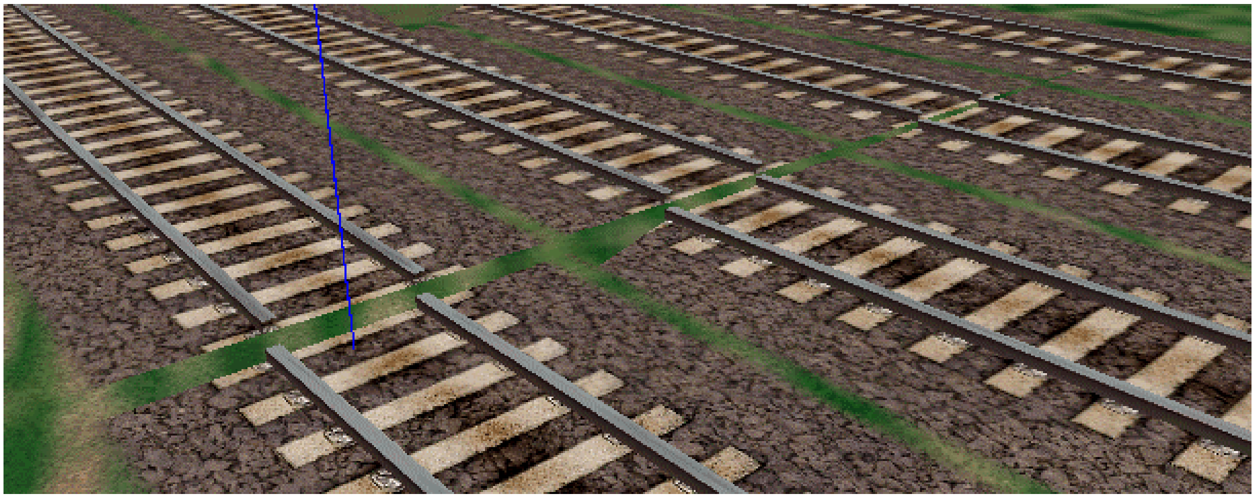
With dynamic sloped curves (next figure), the bank is always zero, so the track joint looks very good. Nevertheless, there is a slight change in slope, but it is quite small for small turn angles (the next three figures use 20° turns): in the illustrated case, it is in fact invisible to the eye from all viewing directions. However, for larger turn angles the mismatch grows: for a 90° turn, the slope mismatch would be 3° for 3°-sloped curve sections!



With non-dynamic sloped curve sections, the end point of the first section has a smaller slope and a non-zero bank at the joint, creating a double mismatch: there will be sudden changes in both slope and bank. This is illustrated in the next figure (made with A1t500r20d curves). Again, the slope change is invisible for small turn angles, but the bank change is clearly seen. This visual mismatch is not a serious problem for a train, because it follows the smooth centerline of the track, while ignoring the bank change.



The effect is greatly amplified if you use multiple-track sections, as shown in the next figure (made with A4t500r20d curves): the effect is so large there that one of the tracks does not connect across the joint (as shown by the presence of the blue pole, which indicates that the route's database does not recognize a joint there). Trains will likely derail at the middle two of these 4 track joints, and will not even be allowed across the unconnected "blue-pole" joint!



This kind of bank mismatch is unavoidable with non-dynamic tracks: it is best not t o use multiple-tracks in sloped curves. To minimize the mismatch in both slope and bank for single non-dynamic track, you should use short curved sections (small turn angles) and small grades. With dynamic track, also use small turn angles and small grades to avoid slope mismatches (there will be bank mismatches).

[NEW SINCE V1.106]

### 3.3.13 ADD-ON TRACKS AND ROADS, INCLUDING XTRACKS

**Adding non-default track and road sections to MSTS is not handled the same way as standard objects. Also, this addition has deeper implications, which can be viewed as "changing" MSTS, since it can affect other routes besides the one you are working on. Before doing this, you should understand the possible consequences.**

The default MSTS track and road sections offer many ways to construct tracks and roads, but still are limited compared to what is possible in reality. For instance, in the case of tracks, "slip" switches are missing (these allow switching between crossing tracks, which is very useful and space-saving in stations); so are switches in tunnels and crossings at large angles; also some standard switches do not include both right-handed and left-handed versions. Furthermore, there are no narrow-gauge tracks.

Therefore, some independent designers have produced a number of add-on track and road sections. For example, the set called XTracks (by Okrasa Ghia, and available for free download as xtracks.zip at Train-Sim.com) provides a large selection of new sections. Types of add-on tracks in that and other packages include:

- various straight and curved track sections with different lengths and radii, including high-speed large-radius curves, and short-radius tram curves;
- track sections with level crossings and pedestrian crossings;
- various crossings with different angles;
- various switches (points), including short-radius switches for trams, single- and double-slip switches, a Y switch with tunnel, and a short-radius Y switch for trams;
- many track sections with tunnels, including switches;
- many track sections with built-in bridges, bridge tracks alone (without trackbed but with guard rails, and a bridge end;
- tracks on a floating pontoon;
- the (non-functioning) default MSTS turntable converted to a track section, and a roundhouse;
- many track sections in narrow gauge.

However, **tracks and roads are not normal objects** (because they need to be connected). So they cannot be added simply like other objects. This leads to complications, which this section tries to discuss.

The track and road sections in MSTS are defined as "global" objects that affect all routes, not just the route you are building (although their textures can be made route-dependent). They are defined in the file tsection.dat present in the MSTS GLOBAL folder. If you change or add track sections in the global tsection.dat file, you will likely impact all routes.

(Caution:  There is also a different file with the same name tsection.dat in each route:  these files are not equivalent and should not be confused!).

Since each type of track and road section receives a unique number in the global tsection.dat file, it has been necessary to allocate new numbers to designers of new track and road sections, in such a way that different designers use different numbers.  This has led to a "standardized global tsection.dat" file (by David Beach, available for download as tsection.zip at Train-Sim.com): this file knows those unique numbers and avoids conflicts.  This file will be updated as new add-on tracks and roads are produced by the different designers.

Thus, **if you wish to use add-on tracks or roads (for example those in XTracks), you must also install the standardized global tsection.dat**, so as to maximize compatibility with other routes.

Replacing the global tsection.dat file of MSTS by the standardized version is not simply an "addition":  it is a modification of a core element of MSTS.  **You need to be well aware of the consequences of this modification of MSTS before deciding to follow this approach.**

**I will call the modified version of MSTS "extended MSTS".  And I will call routes made with "extended MSTS" "extended routes", in contrast with "classical routes" developed for the default MSTS.**

(1)  You should be aware that Microsoft and Kuju (the designers of MSTS) have not been consulted about this modification.  So there is no guarantee that an update (patch) or new version of MSTS will respect this approach, although we very much hope that they will.  An update of MSTS would not likely cause problems, but a new version of MSTS may well take an incompatible approach and make "extended routes" unusable until suitably converted.  It must be added that we don't even know whether it will be possible to run "classical routes" in a next version of MSTS: they may need conversion as well.

(2)  The "extended MSTS" approach has been developed by trial and error, so it remains possible that problems or improvements will arise that require future changes to this approach, resulting in different versions of "extended MSTS".  Therefore, there may appear routes built with different versions of "extended MSTS":  however, these are planned to be upwardly compatible (so that a later version of "extended MSTS" can run routes produced with earlier versions of "extended MSTS").

(3)  Existing "classical routes" will run correctly in "extended MSTS".  This includes the six default MSTS routes, and any routes built within the original MSTS.  However, if you want to edit such "classical routes" in "extended MSTS" with the Route Editor, you will first have to convert them for compatibility with "extended MSTS":  this is done with a separate program called "Horace" (by Okrasa Ghia, available for download at Train-Sim.com).  Horace updates the track numbers to reflect the new numbers present in the standardized tsection.dat, thereby converting a "classical route" into an "extended route".  Note that you only need to apply Horace to the route you want to edit:  other routes need not be converted with Horace.

(4) A route developed in "extended MSTS" without add-on tracks or roads would need to be converted using Horace to run in the original MSTS, using the original global tsection.dat.

**There are different ways that you can handle the coexistence of "classical routes" and "extended routes" on your computer:**

(1) In "extended MSTS", you can run both "classical routes" and "extended routes". You may have to update the standardized global tsection.dat file when you install a new "extended route", to reflect new track or road sections installed with this new "extended route"; older "extended routes" should still run after that update.

(2) If you have the original MSTS, but want to run an "extended route", you should first change MSTS to "extended MSTS", by replacing the original global tsection.dat with the standardized global tsection.dat and by installing any needed track or road shape files (*.s, *.sd), for example from XTracks (do this while Train Simulator does not run).

(3) If you want to change "extended MSTS" back to MSTS, you must disable any "extended routes", or else they will cause the original MSTS to fail. You disable "extended routes" by removing their folders from the ROUTES folder (move their folders to another folder on your hard drive, such as into c:\backup).

(4) You can have two versions of MSTS installed on your computer (by copying the entire "Train Simulator" folder to a new folder with a new name, such as "XTrain Simulator" or "extended MSTS"): the first is kept as the default MSTS, while you change the second to "extended MSTS". The default MSTS can be used to run all "classical routes", while the second version can be used to run all "extended routes". You may have to update the standardized global tsection.dat file and install new track or road shapes when you install a new "extended route": older "extended routes" should still run after that update.

**If you distribute a route, you should keep in mind that different users will have either MSTS or "extended MSTS" (or more rarely both) on their computers. Remember this:**

(1) You must make clear to future users (before they download your route) whether your route is an "extended route": you could write in its "advertisement" that it uses XTracks or other add-on tracks, and requires the standardized global tsection.dat. In the installation instructions, you must specify which version of tsection.dat is needed, as well as which track or road add-ons must be installed.

(2) The user can change MSTS to "extended MSTS", or vice versa, as described above (the user should carefully follow the original instructions of the authors). However, not all users will be comfortable doing so, or even willing to do so.

(3) If you develop a "classical route" with the original MSTS, this route will work under MSTS as well as "extended MSTS" (but it would need to be converted with Horace to become editable under "extended MSTS").

(4)  If you develop an "extended route" (with "extended MSTS"), the user will need to run it under "extended MSTS" (with a suitable version of the standardized global tsection.dat, since this file may be updated occasionally).

The following list gives an idea of **add-on track and road sections that are currently (May 2002) recognized** through the standardized global tsection.dat file.  They come from XTracks by Okrasa Ghia and from other packages (you will have to download the packages individually):

- Microsoft/Kuju sections (initially inactive, activated by Okrasa Ghia), as well as tramway sections, standard gauge sections and narrow gauge sections (all in XTracks.zip);
- Tim Bridge's UK Narrow Gauge sections (available from http://www.uktrainsim.com/);
- Teemu Saukkonen's Bridge and Carfloat sections (available as Enocell's Bridges from Train-Sim.com).

[NEW SINCE V1.106, RENUMBERED SINCE V2]
### 3.3.14 CHANGING THE TRACK'S APPEARANCE

The track appearance is given by several graphics files used to draw the track bed and rails.  The graphics differ among the default routes, giving four styles:  one for EUROPE1 and EUROPE2, a second for JAPAN1 and JAPAN2, a third for USA1, and a fourth for USA2.

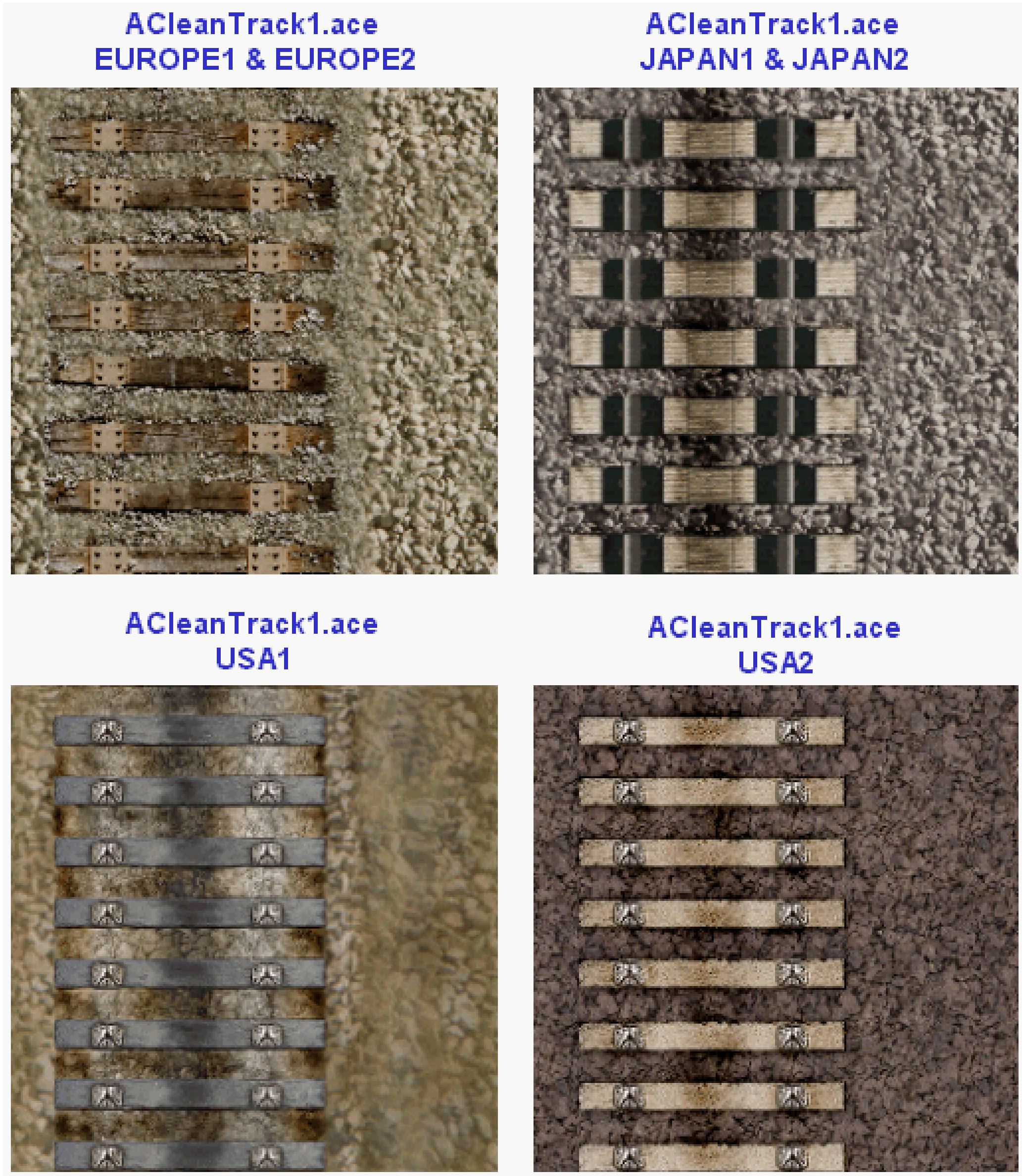
The next figure shows the four default styles of track bed (these are contained in the file ACleanTrack1.ace), that appear when there is no snow.  Three other files are also needed to draw tracks:  ACleanTrack2.ace contains the rails and other details, while ACleanTrackBase.ace contains soil texture below the rocky track bed (I am not sure when this is actually used, if it used at all), and ATrackTunFloor.ace draws the track bed in a tunnel;  there is also ATrackTunWall.ace to draw the tunnel walls and ceiling, which are identical in all default routes.  There are also snow-covered versions of the ACleanTrack*.ace files, which show a light snow covering.

If you want to change the appearance of tracks in your route to that of one of the default styles, you thus only need to copy the corresponding style of track texture files.  For example, if you want to copy the track style of the USA1 default route, you should:

- copy the 5 abovementioned files from the USA1 route's TEXTURES folder to your route's TEXTURES folder;

- copy the 5 abovementioned files from the USA1 route's TEXTURES\SNOW folder to your route's TEXTURES\SNOW folder.

If you want to create your own track (and/or tunnel) appearance, you will have to modify or replace the abovementioned files (including the snow versions).




If you want your tracks to appear as if **covered by deep snow**, you can download snowpack.zip by Ruben Geerling from Train-Sim.com. By running "Snow track.exe" and unzipping to your route's TEXTURES\SNOW folder, both ACleanTrack1.ace and ACleanTrack2.ace will be overwritten: now the trackbed and ties will be invisible and replaced by white snow. You should combine this with "deep-snow" terrain textures, as discussed in section 3.9.1d.

[NEW SINCE V2]
3.3.15 CHANGING THE TRACK'S APPEARANCE

The track appearance is given by several graphics files used to draw the track bed and rails. The graphics differ among the default routes, giving four styles: one for EUROPE1 and EUROPE2, a second for JAPAN1 and JAPAN2, a third for USA1, and a fourth for USA2.

## 3.4 Special track layouts

### 3.4.1 BRIDGES

**By default, MSTS offers only one type of bridge** in a new route (it is called jp2bluebrg in the Object selector, and resides in the file jp2bluebrg.s in the Shapes subfolder): that bridge is meant to carry a road rather than tracks, but you can use it to carry tracks as well.

**To place tracks on another type of bridge, you need to first import a bridge from a default MSTS route**. That unfortunately involves several steps: finding a suitable bridge, copying it into the Shapes subfolder of your route, and listing it in the *.ref file of your route. To do this, **see section 3.10**.

**Let's assume that you either use the default bridge mentioned above, or that you have imported the bridge you want** from a default route.

The normal way to lay track over a bridge is to **first lay track over flat terrain**, **then place a bridge object under the track**, and **then depress the terrain** to form a valley.

If the terrain is not flat to start with, you can lay track and adjust the terrain to fit it by pressing Y: then you can add track sections until you reach the other side (if you use a long piece of track which goes beyond the end of the bridge, that may not even be necessary).

For very long bridges these methods of laying track may be inconvenient, however.

TIP: Sometimes, I found it possible to **add tracks in mid-air** to form a long bridge without adjusting terrain. The trick is to look vertically down onto the track, and then to position the next track as usual, sometimes by zooming in, sometimes by zooming out: often the new track will be placed on the ground below (delete it right away), but at other times, it will connect properly to the "flying" track. (Move the camera a bit if you fail.)

TIP: If adding tracks in mid-air also does not work, the next solution is to pull a bit of terrain up to the end of the existing "flying" track (raising just a couple of grid points to form a tower is enough): then you can add another track section as usual, starting from the top of that tower (which you can destroy later).

**Once the track is laid, you can place a bridge object under the track**: you need to carefully move and rotate it until it fits properly. Because this involves many degrees a freedom and requires a lot of accuracy, I recommend the following procedure to do this.

1) **Select and place a bridge section:**

- select the bridge section you want;
- do not use wire-frame mode for the terrain (if necessary, toggle by pressing W);
- press F5;

- place the bridge section near its desired position:  this will normally place its pillar(s), if it has any, on the ground, and orient the track bed to the North;
- press F6 or right-click to open a pop-up menu:  select "General (cont.)" and put a check mark before "Terrain object";  if you don't, the track ties and ballast will not be visible after the bridge is placed.

2)  **Orient the bridge section** so it becomes parallel to the track:

- move the camera so it looks straight down on the track and bridge section;
- press F4;
- turn the bridge:  first move roughly (using the keypad's left- and right-arrows), then fine-tune until it is parallel to the track (dragging the mouse left and right while pressing Ctrl);  fine-tuning the orientation is quite important so that other bridge sections fit the first one correctly (this fine-tuning can tilt and slope the bridge slightly:  we will correct for that in step 4).

3)  **Roughly center the bridge section over the track:**

- keep the vertical camera view;
- press F3;
- while pressing Ctrl (to freeze the bridge's altitude), left-click to roughly place the bridge section with its center over the track.

4)  **Adjust the bridge section's slope:**

- press F4;
- press N:  this gives the bridge section the same slope as the terrain under the track.

5)  **Lower the bridge section** so it fits right under the track:

- keep the vertical camera view;
- press F3;
- push the bridge down to just under track level:  first push it down fast (using the keypad's down- and up-arrow keys while pressing Shift), then slowly (using just the down- and up-arrow keys):  when it reaches track level, the wires that run across the track bed will become invisible under the ties and ballast, while any "handrails" on both edges of the bridge remain visible above ground.

6)  **Move the bridge section horizontally** to exactly where you want it to be:

- keep the vertical camera view;
- press F3;
- while pressing Ctrl (to freeze the bridge's altitude), left-click to roughly place the bridge section;  you can repeat this to try to position the bridge better;
- fine-tune the horizontal position of the bridge:  while pressing Ctrl, use the left-, right-, up- and down-arrow keys to move the bridge to its final position.

IMPORTANT: **After laying a bridge object, right-click on it, and make sure it is enabled as a "Terrain object"**. Otherwise the ties and ballast will not be visible.

[NEW SINCE V1.106] TIP: **Long bridges on a grid**. If you are building a longer straight bridge made of several identical bridge sections, one approach is to place those sections using the grid option of the Placement Tool (see section 3.10.2): the grid option allows you to place the same bridge section at constant and precise intervals, so the sections align and join perfectly. Since bridge sections are initially placed with a north/south orientation, and since the grid is also oriented north/south, you should first build a long bridge in that orientation; you should also do this on a flat piece of terrain. You can guess the grid size by trial and error, or you could "measure" the bridge length with dynamic track and use that length as the grid size. After placing the desired bridge sections, you may move and rotate the entire bridge after selecting all the sections (if the bridge must be sloped, try to give it the right slope before rotating it away from the north/south orientation).

TIP: **Double and redouble.** For a long bridge made of several identical bridge sections, you may also use an alternative approach (this works best if you don't have flat terrain for the using the grid approach, and if the bridge is curved). Position and orient the first bridge section correctly, then **copy and paste** it to make the next section (that way the orientation is maintained; see section 3.10.6 for copying and pasting objects); after moving the second section into place, you can select the first two sections (as a pair) by pressing Ctrl while selecting each, and then **copy and paste the first pair** together to form a second pair that you can move into place; for very long bridges, you can **double the length at each step**, copying 4 sections at once, then 8, etc. (at the other end of the bridge, copy a smaller number of sections to fill the remaining gap).

After you have placed the bridge, **shape the valley** below it (see section 3.5).

The final step is to **fine-tune the terrain around the bridge ends**. It is often convenient to use a steep "stamp" to shape steep slopes there (see section 3.5.4).

NOTE: Some bridges are accompanied by a road piece (for example, the default JP2bluebrg is accompanied by JP2bluebrg_RoadPiece). Such a road piece can be placed in the bed of the bridge to better simulate a road passing over that bridge. However, you may instead place the road type of your choice on the bridge.

### K. Building "First Route": digging a tunnel, and laying track over it

You will next make the single-track mainline go through a hill that requires a tunnel. After that, you will create a spur that loops back over that hill and passes over the tunnel. See section 3.4.2 for more details.

You may have noticed in the Object selector that many track sections have a version with "Tun" or "RndTun" in their name: those are all track sections that have "built-in" tunnel walls and ceiling. We must choose these for making tunnels.

The "RndTun" variety (for "round tunnel") has a circular cross-section like a tube: but it is too low for larger trains, such as the American trains, or for electrified trains that require a high overhead wire. Nonetheless, the American trains will work in the low tunnels: but they will be sticking out through the ceiling!

It is recommended to **use only tunnels labeled "Tun"**, since all trains can pass through them properly.

NOTE: **Switches and dynamic tracks are not available as tunnels**.

Laying tunnel tracks can proceed as with non-tunnel tracks: we will shape the terrain later into a hill that covers the tunnel. So we will be laying tunnel-covered tracks on top of relatively flat ground.

Start from the end of the switch that you laid last (it joins the dual tracks into a single track). We will enter the tunnel with a left curve that slopes up a bit, then go straight and level, and finally slope down while turning right toward the tunnel exit.

**Do the following:**

- select track section A1t500r10dTun.s;
- place it at the end of your last switch: it should curve to the left, showing wire-frame walls (depending on the camera's position, you may not see all wall and ceiling parts);
- press F4;
- slope it up 0.9° (you could make gentler slope changes with 10m straight tunnel sections);
- press Y: the terrain adjusts as usual to the bottom of your track;
- select track section A1t100mstrtTun.s;
- place it after the previous tunnel section: leave it horizontal, and press Y;
- add the same curved section A1t500r10dTun.s: press T to turn it to the left;
- slope it down -0.9°, and press Y.

This tunnel looks very strange on its little dike!

You can inspect the inside of the tunnel with the camera. You may find some terrain leaking into it. But don't worry: we will later lift that terrain up above the tunnel to form the hill.

**Save your work, after deselecting any selected track or objects!**

Next we **create a spur** (as we did once before), but it will climb and turn back over the tunnel, as shown in the next figure.

Before we do this, let's choose a slope for the hillsides that will be formed on both sides of the track.  That slope is called "embankment" in RE and is shown in the Terrain window:  it has a default value of 45°.  That is rather steep for a hill, so we want to reduce that slope to 20°.  Do this:

- **in the Terrain window, enter 20 for the "embankment"**.

Such a terrain slope will look something like this, after you have laid the curving spur:

**To lay the spur, do the following:**

- select the switch A1tPnt10dRgtMnl.s;
- place it at the exit of the tunnel (without slope):  it should branch to the right;
- select track section A1tEndPnt10dRgt.s;
- place it at the curving exit of the switch (not sloping): it should turn to the right;
- extend this spur by adding the same curving track section many times over, sloping it upward at 2.1° (but use a smaller slope at first to give smaller slope changes);  press Y after each section is laid;  stop climbing when you reach about 30m of altitude (watch the "y" in the Object window);  and stop turning when the track points at the middle of the tunnel;
- then add several non-sloping 100m straight sections (A1t100mstrt.s) to pass over the tunnel, until you are well beyond it.

NOTE:  If you change your mind about the terrain shape you have created, **press ` (the back-quote) to cancel your last terrain changes** (those made after your last save).

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again.**

This exercise has shown **one method of creating a hill**, or at least the beginnings of a hill: lay track and press Y to pull the terrain up to it. There is no need to make the track connect to existing track: just lay it anywhere on the terrain, slope it, rotate it (after pressing F4 to slope it, drag the mouse left or right), press Y, and then delete it.

Have a closer look inside your tunnel now: enter through either end. You should see the terrain blocking it! The good news, however, is that a train can pass through the tunnel even if blocked by terrain. The bad news is that to unblock the tunnel entrances is a complicated process: it requires first making a hole in the terrain (easy, but the hole will be too big), then importing a tunnel entrance object and a terrain texture (more complicated), and finally placing the tunnel entrance and texture to hide the big hole (an exacting exercise). We will do this later. And all of this is explained in detail in section 3.11.

NOTE: **You can move the camera underground after pressing /** (a second press will bring the camera aboveground again).

**I strongly recommend that you test your route at this stage, as explained earlier: delete old activities, create a new activity, and then drive along your track.**

Next we will create an underwater tunnel.

**To continue building our First Route, jump to the next blue box.**

## 3.4.2 UNDERGROUND TUNNELS - THE EASY PART

Laying tracks in tunnels underground is easy, **almost as easy as laying tracks outside tunnels** (see section K). **However, you will need to add tunnel entrances at both ends of the tunnel: that is much more complicated**, unfortunately: it uses a trick (making a hole in the terrain) and requires importing both a tunnel entrance object and a "transfer" texture to cover up parts of that hole. See section 3.11 for adding tunnel entrances.

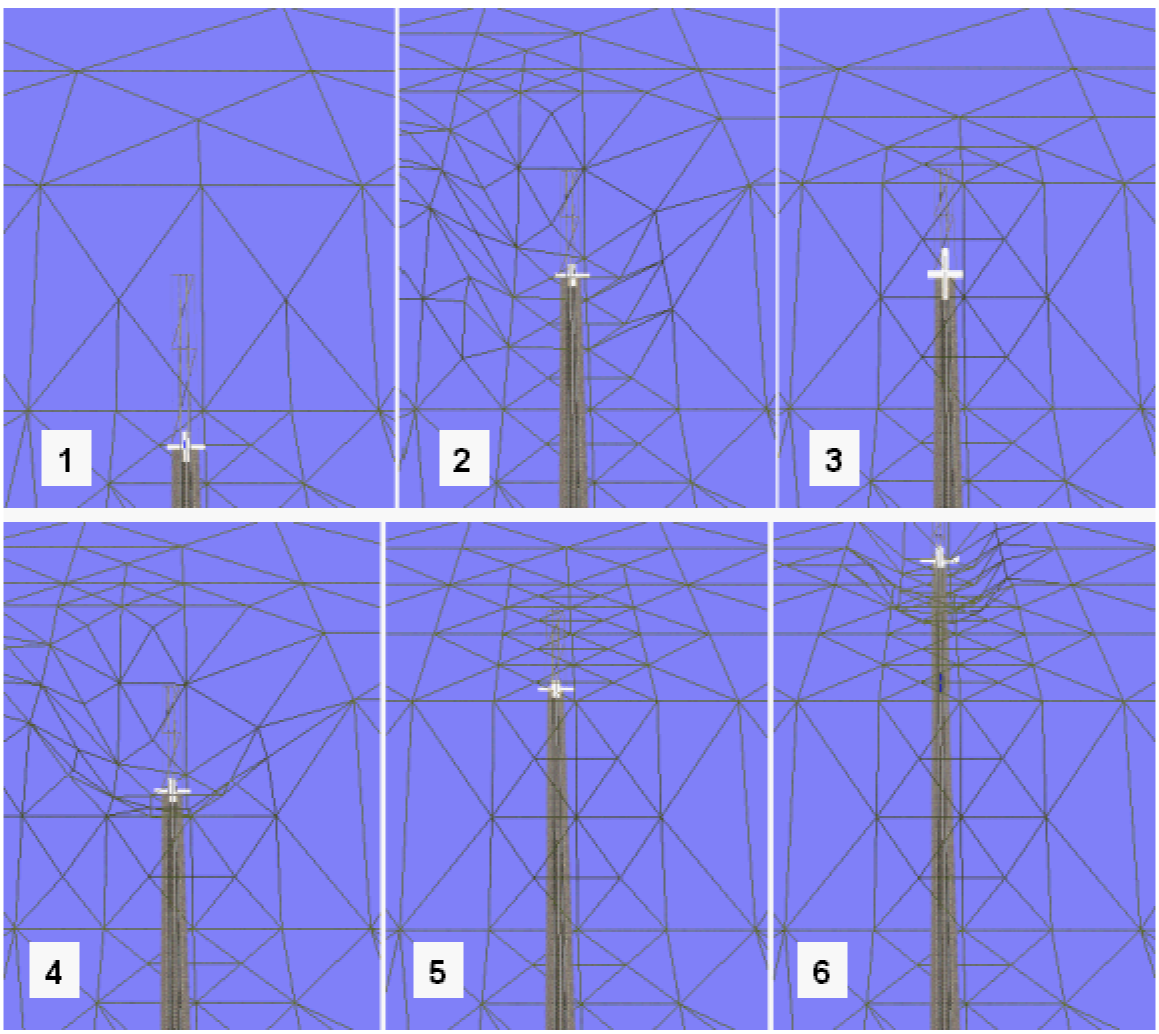
**In this section, we only discuss the easy part: laying tracks underground. The hard part is dealt with in section 3.11**: continue there after you have learned to import and handle textures, transfers and objects.

The normal way to produce underground tunnels is to **first lay tracks (with tunnel walls) as usual**, and to **adjust the terrain by pressing Y so the ground stays below the track**: this creates a deep cut in a mountain, if you already have a mountain (use steep walls, as explained in section 3.5.4). You can see an illustration of such a cut in section 3.4.3.

**Later, you should go back and pull the terrain above the tunnel to the desired height of the hill or mountain**, and fine-tune the terrain near the tunnel entrances.

**Simpler is to create the tunnel before you build the mountain!** Build the mountain over the tunnel later on.

**Another approach can be called "dig while covering up your tracks" through an existing mountain, illustrated in the next figure.** It can be relatively efficient and quick. Its principle is to lay a track section into the mountain (see panel 1 of the figure) and adjust the terrain to it (by pressing Y, see panel 2), thus digging a trench (don't save it!); then lay the next track section at the end of the trench (panel 2) and RESTORE the terrain (by pressing `) to recover the mountain(panel 3); next adjust the terrain to this latest track section (press Y, panel 4), lay the next track section in the resulting trench (panel 4), and restore the mountain (press `, panel 5); continue like this (panel 6) until you emerge from the other side of the mountain. Remember to NOT save the result until you have completed the tunnel.



As with bridge building, sometimes it is possible to **not** adjust the terrain while laying track underground. With a bit of luck, you can point at the desired location through the terrain wire mesh and get RE to place the next tunnel section underground where you want.

NOTE: You can **move the camera underground** after pressing /, but you can't place tracks while the camera is underground; pressing / again will make the camera jump above the surface.

WARNING: **If you want to add objects** (such as mileposts or signals) **inside a tunnel, do so before pulling the terrain up above the tunnel.**

To add tunnel entrances, see section 3.11.

**L. Building "First Route": showing water, and making an underwater tunnel**

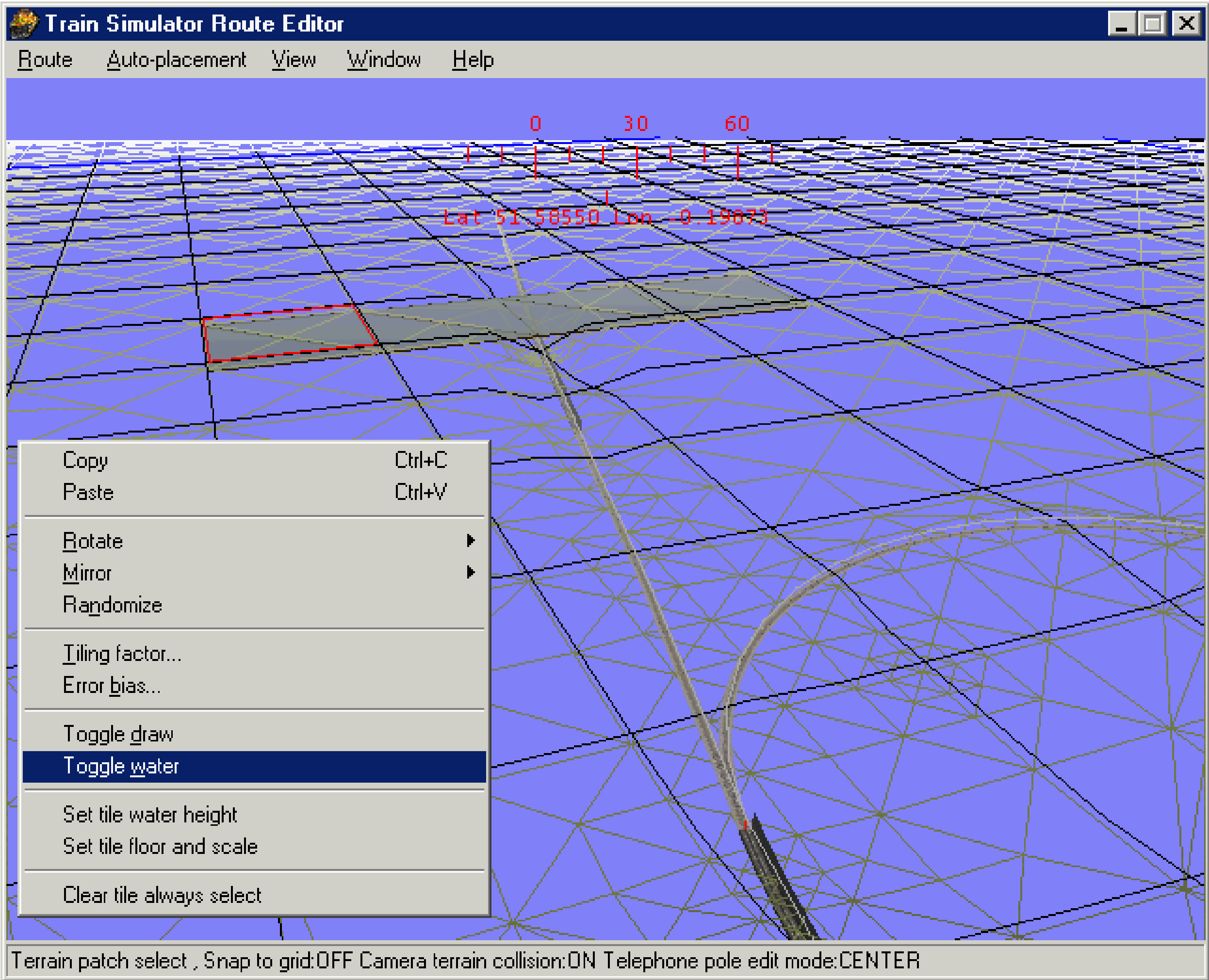
You will next make the single-track mainline go under a river, which needs water, under which you will lay an underwater tunnel. See section 3.4.3 for more details.

**To show water** in MSTS (and RE), you need to do two things: **toggle water on**, and **push the terrain below the water level**. **Water is toggled on in small patches** that you select: let's call those "wet patches". Each such patch is 128m x 128m in size. Within those wet patches, water becomes visible wherever terrain is depressed below an altitude of 0m. (The 0m water level can be changed for mountain lakes, etc.)

**To lay an underwater tunnel**, you need two things: the tunnel of course has **to slope down to go under the water level**; and the **tunnel should not pierce the water level in a wet patch**; otherwise, the tunnel will look flooded. So, **an underwater tunnel must pass completely underneath any wet patch**.

This takes a bit of planning, because we have to start sloping the track down early enough. Since the steepest allowed track slope is 3° (which is about 5m up or down for every 100m forward), and since an underwater track needs to be at least 15m below the water level, we need at least 300m for the down-slope. Measured in terms of 128m patches, we thus need about 3 patches to slope down, 1 or more patches to pass under the water, and another 3 patches to slope back up.

For our case, look at the next figure. It looks beyond our tunnel and our curving spur, in the direction where our mainline will continue and pass under a river: the underwater tunnel is actually already visible there. I have pressed F7 to show **the patches**: they **are outlined in black**. **The 4 darkened patches are "wet": they have water toggled on**; the dark color is simply the blue water surface (at 0m altitude). These wet patches will later contain our river.




Notice that there are at least three non-wet patches between our last switch (at the tunnel exit) and the wet patches: those are needed for sloping down to get under the water of the wet patches.

The figure above shows our resulting plan of construction.

NOTE: Your track positions may not be exactly those shown in the figure above. If you followed the entire First Route construction to the letter, you should have the situation depicted in the figure above. But if you varied your route somewhat, you may have a different situation and you should adapt your plan to that situation.

**First do the following**, to get away from the hill and the spur:

- reset "embankment" in the Terrain window to 45, if it is not now set to 45;
- place a 100m non-sloping straight section (A1t100mstrt.s) after your last switch (the one that shows up near the bottom of the figure above).

Next **toggle the water on in the patches that you want to make "wet"**:

- move the camera to give you a bird's eye view like that in the screenshot above;
- select a patch, so its border turns red as shown;
- right-click to open the pop-up menu;
- click Toggle water;
- repeat the last three steps for other patches that will define your river.

**Save your work, after deselecting any selected track or objects!**

NOTE: In the following, your exact choice of track lengths and number of sections may depend on where your First Route has led you so far. The only important point is to make your tunnel pass completely under the "wet" patches.

Now **start diving into the ground**:

- use several 10m straight sections to gradually slope to -3° (I use slopes of -0.6, -1.2, -1.8, -2.4°), pressing Y at each step;
- select track section A1t100mstrt.s;
- place 2 of these 100m sections after the last 10m section, sloping them down to -3°;
- press Y to depress the terrain: it will form a descending trench;
- select track A1t100mstrtTun.s;
- place two of these tunnel sections after each other, both sloping down -3°, press Y for each.

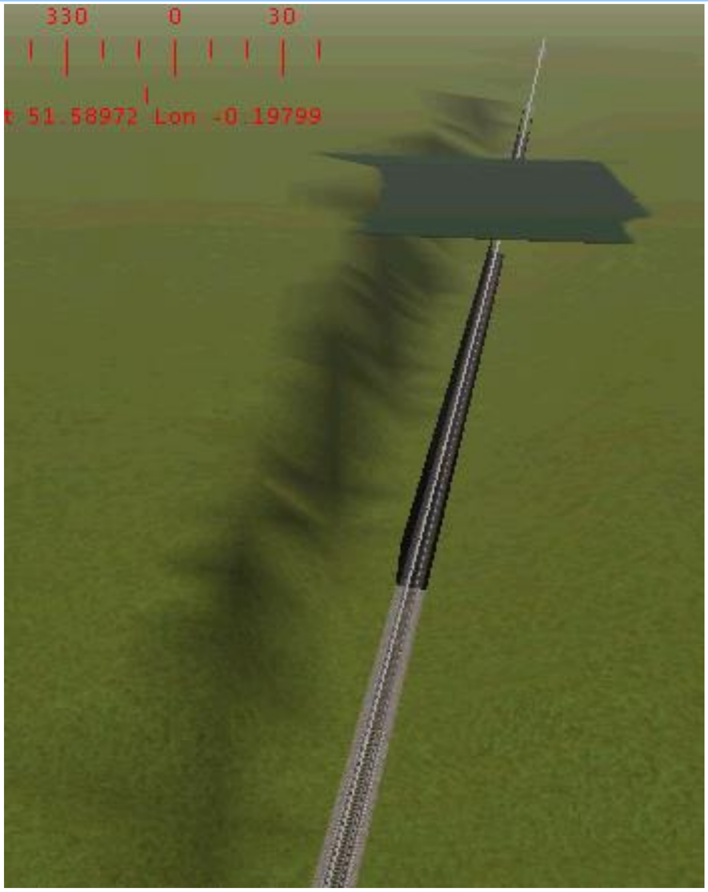
NOTE: You can decide when to change to tunnel sections: already above the water level or only below the water level; that is an engineering choice!

**This should bring your track into the wet patch and below its blue water surface**: the water should become visible as a triple layer at altitude 0m. If your particular tracks don't reach there yet, add some shorter tunnel sections.

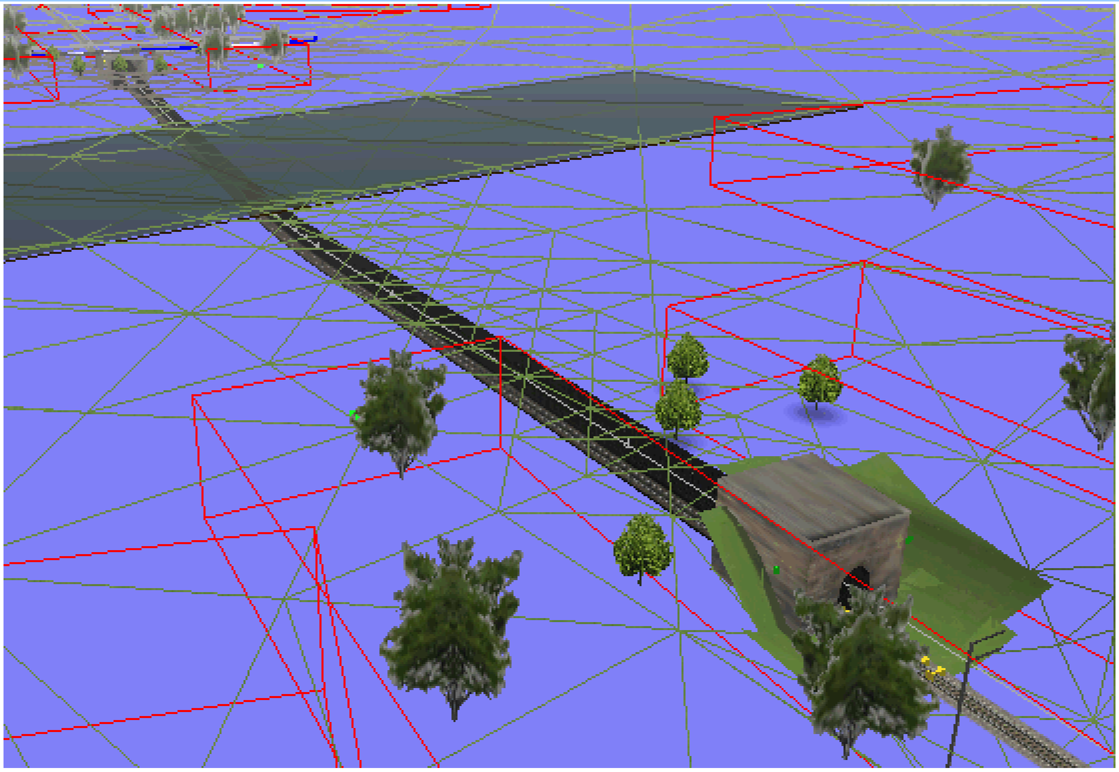
Now **gradually change the slope of your tunnel from -3° to +3°**, adding several 10m straight tunnel sections (A1t10mstrtTun.s). Press Y each time.

Then use two 100m straight tunnel sections to **climb up** at +3°, followed by two 100m straight non-tunnel sections, also at +3°, to get back above ground. Press Y each time.

**Finish off** with 10m sections **to bring the slope back to zero**. The result should look somewhat like the following figure. Here you see the tunnel lying on the bottom of the trench. The trench has made a piece of water visible. You will later raise the terrain above the covered part of the tunnel, while keeping it below the water surface where the river should be.




The next figure gives a preview of what the tunnel may look like in RE after shaping the river, adding tunnel entrances, and planting trees.



**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will terminate our mainline by making a simple station with yard, before adding a couple of more complex track layouts: a wye and a reversal loop.

**To continue building our First Route, jump to the next blue box.**

### 3.4.3 UNDERWATER TUNNELS - THE EASY PART

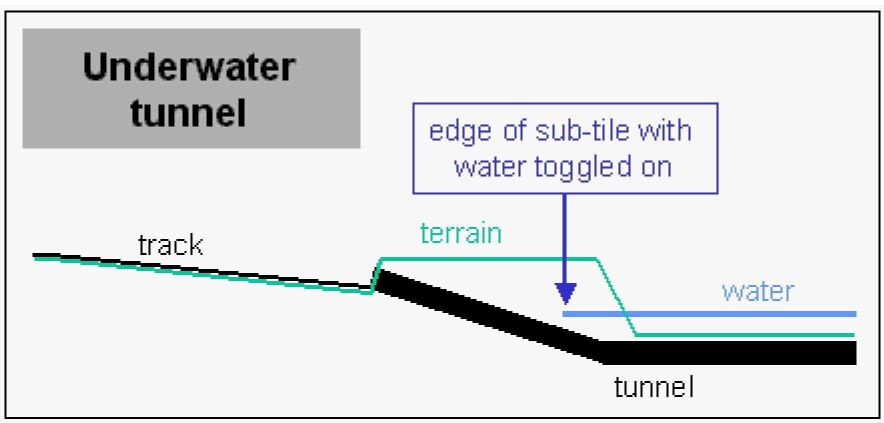
Underwater tunnels are **basically the same as underground tunnels**, with three simple twists (see section L for more illustrations):

- not surprisingly, the tracks should be laid sloping down to below the water level (the terrain can be made to initially follow the track down, forming a trench, as discussed for underground tunnels, or, again with some luck, you can lay tunnel tracks below ground without depressing the terrain); remember that by default the ground level is initially 1m above the water level, and that the water surface is actually composed of three surfaces to give some depth impression.

- avoid letting the tunnel penetrate the water level within a patch that has water toggled on: otherwise the train driver will see a flooded tunnel and the train will appear to sink into the water, or to emerge from water at the other end; the tunnel can pass without problems under a patch that has water toggled on;

- it is best to make the terrain pass above the tunnel, after laying the underwater tracks (thereby burying the tunnel); but of course keep the terrain below the water level (so the water shows up above); the terrain should only intersect the tunnel near its two entrances (alternatively you can make pieces of terrain invisible where they intersect a tunnel, by pressing V after selecting a nearby grid point); otherwise the driver will see terrain within the tunnel.

Here is a **diagram showing how the track, tunnel, terrain and water should ideally be layered**:



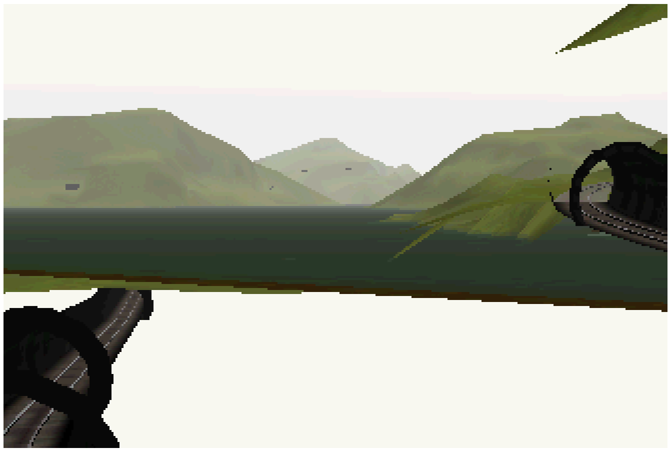

NOTE: You need a length of between 200 and 300m in a water-free patch to make a tunnel dive from above sea level to below sea level, with a 3° slope. So you will have to plan such a slope rather carefully, given that patches are 128m on a side! Normally you will need 2 to 3 water-free patches to manage this.

The next figure shows a **tunnel under construction**: it lies in a trench in a hill (produced by pressing Y after each tunnel section is laid with a downward slope); it dives under the water of the next patch (the water is toggled on in that patch, but toggled off in the patch near the camera). The tunnel continues underwater until it emerges on the distant island (the emergence is not seen in this view).

The next step in the construction is to raise the terrain to cover the tunnel, both under water and on land.

The next figure shows two separate **completed tunnels, viewed outward from under a transparent hill** near the coast (it is as if you were under the hill below the word "terrain" in the above diagram). The terrain has been raised to bury the underwater tunnels, and to bury the tunnels under the hill on land.

The left tunnel (the same one shown in the last figure) now dives under the green sea bottom, which in turn lies below the blue water surface. The right tunnel has just climbed above the water level and is emerging from the ground, as it turns right onto land. The water surface starts at the edge of a patch that has water toggled on. (The sea bottom, in this underground view, only seems to start under the sea: that is because terrain is sometimes transparent from below, where the camera is, so the nearby hill is invisible.)



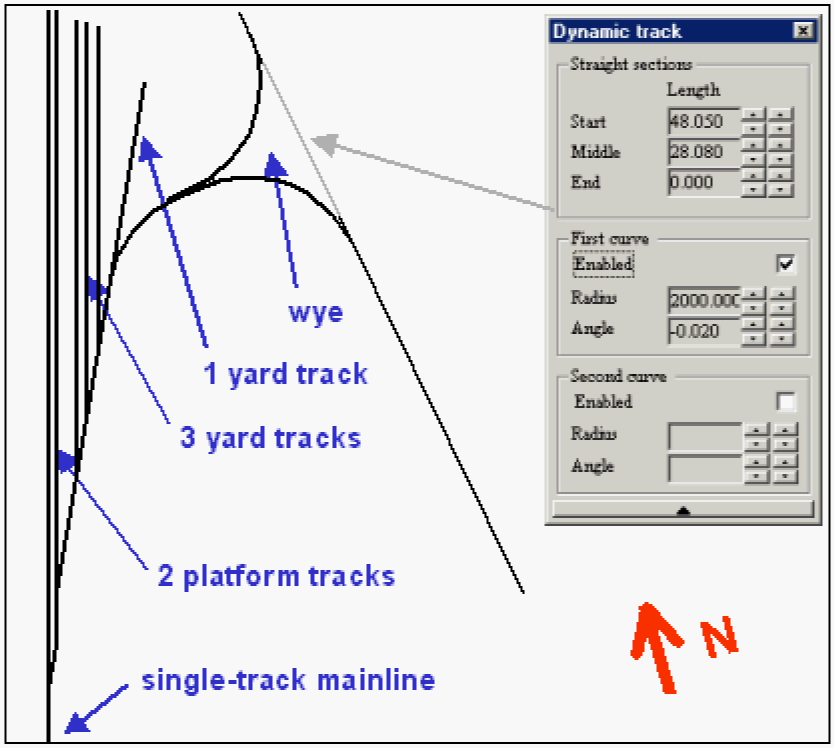
NOTE: The camera can go underwater at any time; in addition, you can move the camera underground after pressing /; and pressing / again will make the camera jump above the surface.

To add tunnel entrances, see section 3.11.

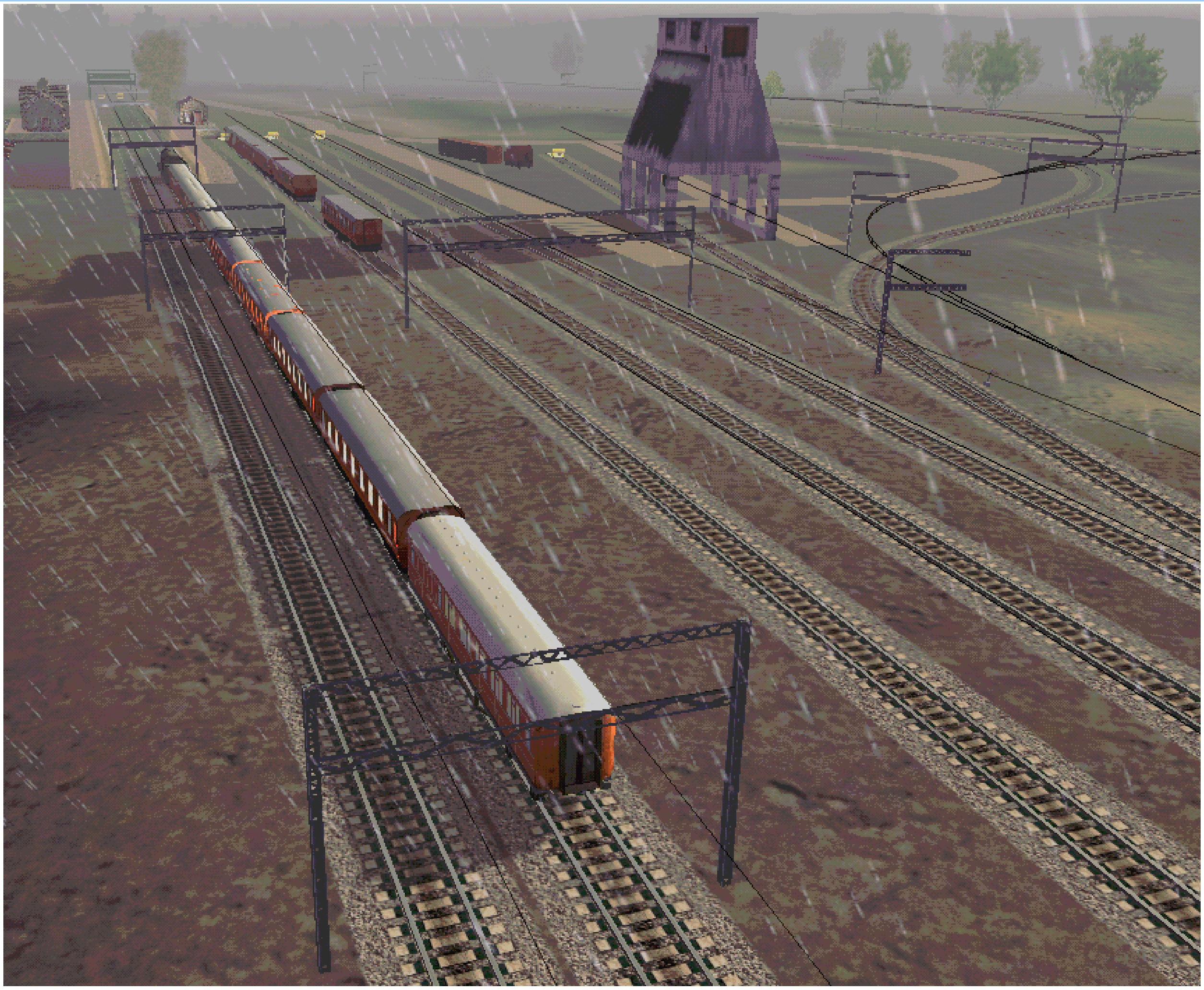
## M. Building "First Route": making a simple station with yard; adding a wye and a reversal loop

You will next create a simple yard. You will then add a "wye" to it, which allows a train to reverse its direction. As the final laying of tracks, you will also place a reversal loop on the last spur. See section 3.4.4 for more details.

For your guidance, the following diagram shows the layout of the destination station, with its two platform lines at left, the yard with 3 + 1 tracks to its right and the wye further to the right. You are laying track coming from the bottom left.




For further guidance, see the following view of a final version of the North London station and yard, looking to the Northeast, with the 2 platform tracks at left and the wye at top right (you can guess yellow buffers at the ends of the tracks):



First **make the station**.  The station will be a pair of platform tracks similar to our departure station, except that we will insert an extra switch that will lead to the yard.  These will use level tracks (no slopes).  Do the following (remember to press Y after each track is laid!):

- at the end of the track coming out of the underwater tunnel, add a 100m straight section;
- add a right-handed switch (A1tPnt10dRgtMnl.s):  it should branch off to the right to create the second platform line;
- to the curved exit of this switch, add a curve to the left (A1tEndPnt10dRgt.s) to start the second platform track:  press T to make it turn left;
- to the straight exit of the switch, add 60m of straight track (50m + 10m);
- after the last curve that you laid, add one more right-handed switch (A1tPnt10dRgtMnl.s):  this will lead to the yard;
- now your mainline and the track parallel to it end together, and you can add some dual track to make the two platform lines:  add two sections of type A2t250mstrt.s;
- at the ends of both platform tracks, place buffers (A1tUSBuffer.s).

**Save your work, after deselecting any selected track or objects!**

Now **build the yard**:  it will branch off to the right and split up into three yard tracks to the left that are parallel to your platform tracks, and it will have another short yard track lined up with the switches.  Do this as follows:

- at the curved exit of your last-laid switch, place a 50m straight section;
- to its end, add a left-handed switch (A1tPnt10dLftMnl.s), to create a yard track to the left;
- to the straight exit of this switch, add a 10m straight section;
- repeat the last two steps twice more, so as to place a total of three left-handed switches in succession:  each creates a yard track;
- at the curved exit of your last-laid switch, place two 100m straight sections, forming a yard line;
- do the same with the two other yard lines, but add 100m or 50m track sections to bring the ends of the three lines close together (they will not match perfectly, so you could add straight dynamic sections if you wish, but there is no real need);
- add three buffers to terminate the three yard lines;
- at the straight exit of the last-laid switch, add a 10m straight section;
- add another right-handed switch there, to make a branch to the right:  it will go to the wye;
- to the straight exit of this switch, add a 100m straight track, terminated by a buffer:  it is just another yard track.

**Save your work, after deselecting any selected track or objects!**

**I strongly recommend backing up your FirstRoute folder again now, because you are going to add more "dangerous" tracks next.**

Now **add a wye**, which will branch off to the right of the yard.  Do this:

- at the curved exit of the last-laid switch, add a 45° right-handed yard curve (A1t45dYardCrvRgt.s);
- add to this curve a wye switch (A1tYPnt10dMnl.s):  make sure it is oriented so that it fans out both left and right.

We will **complete the wye as illustrated in the diagram in section 3.4.4c**:  consult that diagram to follow the instructions given here.  You will be using three curved dynamic track sections, **as follows**:

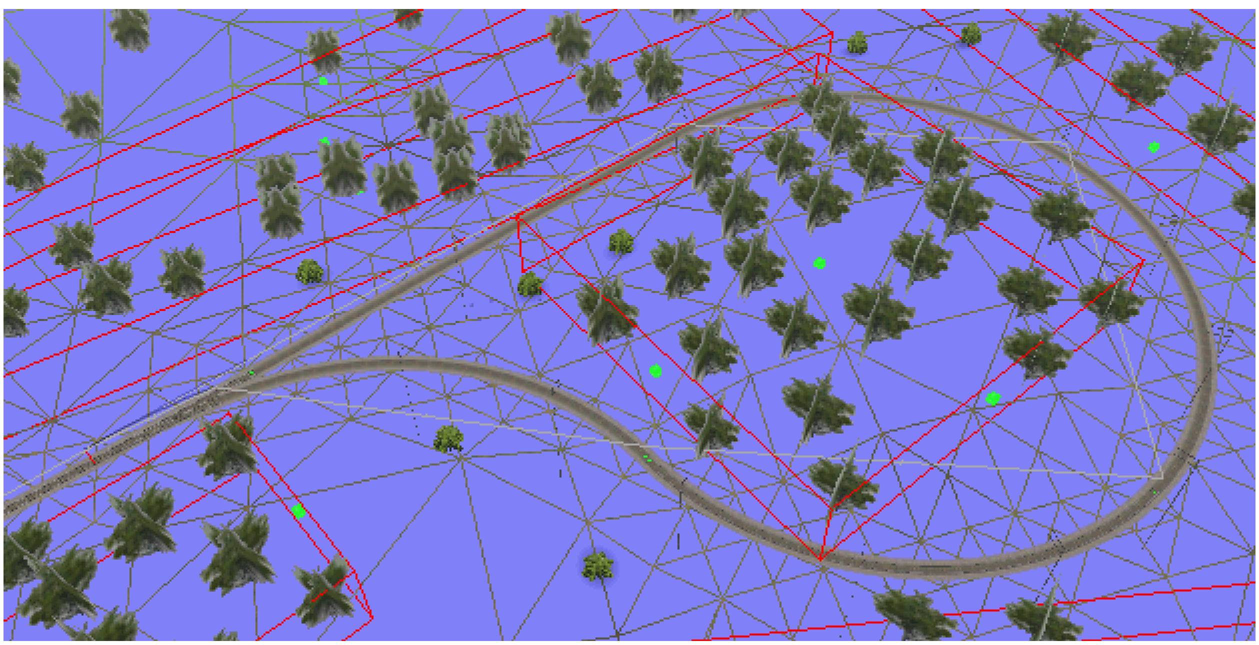
- place a curved dynamic track at the left exit of the wye (as shown in green in the diagram), with a radius of 80m and a turn angle of -1.23 radians (make sure you set the length of the Start segment of the dynamic track to zero);
- place a right-handed switch (A1tPnt10dRgtMnl.s) at the end of that dynamic track, oriented as shown in the diagram, so it can later connect with the red track segment;
- place a curved dynamic track at the right exit of the wye (as shown in blue in the diagram), with a radius of 80m and a turn angle of +1.23 radians:  the +1.23 means that you first set -1.23 radians and then press T to reverse to a right turn (make sure you set the length of the Start segment of the dynamic track to zero);

- place a left-handed switch (A1tPnt10dLftMnl.s) at the end of this dynamic track, oriented as shown in the diagram, so it can later connect with the red track segment;
- at the straight exit of this last-laid switch, add another dynamic track section (shown in red in the diagram): it is more complex than the dynamic tracks we have created so far; it has a Start Length of 48.05m, a First Curve Radius of 2000m, a First Curve Angle of -0.02 and a Middle Length of 28.08m (you should set these in this order; these values are also shown in the diagram above); this should make a perfect joint with the other switch: after you deselect the dynamic track, check that the blue poles disappear and a gray/white line forms along all the tracks;
- finally, to each free exit of the two last-laid switches, add a 250m straight section, and a buffer.

**Save your work, after deselecting any selected track or objects!**

Next **add a reversal loop**: it will be attached at the end of the spur that swings up over the hill above the underground tunnel that you built.

This loop will look as shown next (after the hill is completed and forests and trees are added later on).



**Consult the layout shown in the diagram in section 3.4.4b, and proceed as follows**:

- check that "embankment" is set to 20 in the Terrain window;
- place a right-handed switch (A1tPnt10dRgtMnl.s) at the end of the existing track, so it branches off to the right;
- press Y: do this after each track section is laid, to build the hill as you move forward;
- at the curved exit of the switch, add a dynamic track section with radius 80m and turn angle 1.39 radians, reversed by pressing T, so it turns to the right (see the blue track in the diagram);
- at the end of that dynamic track, place two others with the same radius of 80m and a turn angle of 1.57 radians, turning left (see the two green tracks in the diagram);

- add one more similar dynamic track, but now with radius 80.28m and turn angle 1.566 radians to the left (see the dark red track in the diagram);
- now add your last track of this project, at the free end of the switch: a straight dynamic track of length 135.86m.

NOTE: This loop may give you some trouble while running a train (it has for me in various ways: train breakup, sudden reversal, uncontrolled acceleration, the problem depending on the train). This is an illustration of the unpredictability of loops (and to a lesser extent of wyes), and supports the warnings that I give in this guide about loops (and wyes). See section 3.4.4 for more details.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again.**

**This is also a good time to test your tracks, since you have installed both a wye and a loop.** If you have any trouble during this test, it probably is due to either the wye or the loop (more likely the loop). You can determine which of the two (the wye or the loop) causes the problem, by removing one track from the loop and running a new test with a new activity: if that solves the problem, the loop was its cause; if it does not, probably the wye is the cause; then remove one track from the wye and run a new test with a new activity. If the loop or the wye was indeed the problem, you may have made a mistake in laying it, so rebuild it (delete all its tracks and lay them again). By this approach, you can step-wise narrow down the source of the problem and try to fix it. This illustrates that if you don't test frequently, it may become very difficult and time-consuming to trace the cause of a problem.

Next we will start shaping the terrain, beginning with the river and the underwater tunnel.

**To continue building our First Route, jump to the next blue box.**

### 3.4.4 LOOPS AND WYES

Loops (sometimes called balloons) and wyes are **very useful for reversing the direction of travel of trains**. However, **MSTS is very sensitive to both: if placed incorrectly, loops and wyes cause MSTS to freeze in various ways**.

I will only describe making **level** loops and wyes. Sloping loops and wyes can also be built: the only difference is that the joining track will then need to also fit the altitudes.

#### 3.4.4a Positioning loops and wyes

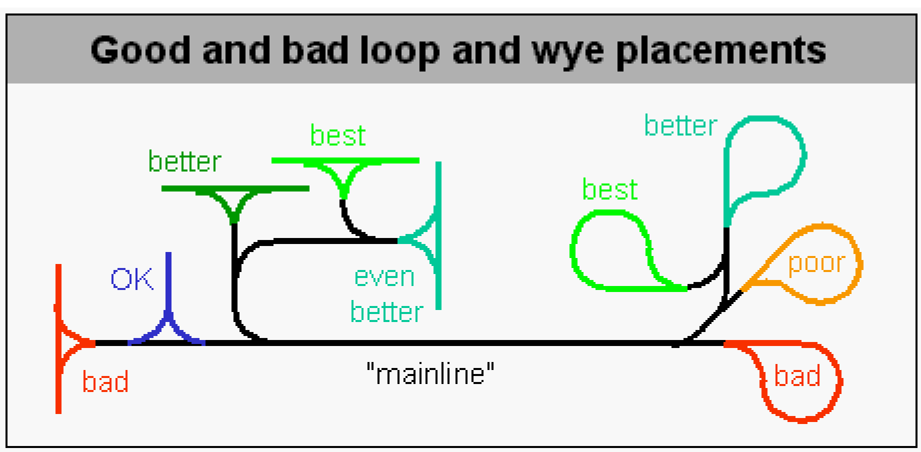
Unfortunately, I am not aware of a general rule that tells you how to correctly place loops and wyes in a way acceptable to MSTS.

Therefore, you have to **build a route by trial and error**: lay out a track design; try to run a train (using an activity that you make for that purpose); if this fails, change the layout; etc. That is why it is recommended to first build a simplified trial route and run frequent tests as the route develops.

Here are hints that seem to help (but not always!): **avoid a track layout that allows driving round and round in a circuit (in other words: avoid a track layout on which a train can continue driving forever without stopping); do not put loops and wyes "face to face" at opposite ends of your mainline (that creates a circuit); put loops and wyes on branch lines, spur lines or sidings rather than on a mainline**.

The next diagram suggests good and bad positions for loops and wyes, relative to a "mainline". But I offer no guarantees! It is not even clear to me what exactly is a "mainline", but I suspect it has something to do with going "straight" through switches, instead of turning at switches. Remember to test any placement you choose!

Letting a mainline branch into a loop or wye (as shown in red in the diagram) is usually fatal; it seems to be OK to place a wye "sideways" within a mainline (as shown in blue in the diagram).




TIP: **put as many turnoffs as possible between your "mainline" and any loops or wyes**.

TIP: **Replacing a fatal loop by a wye will often solve your problems.**

TIP: **Replacing the wye switch (A1tYPnt10d.s or A1tYPnt10dMnl.s) by another (asymmetrical) switch will sometimes solve your problems.**

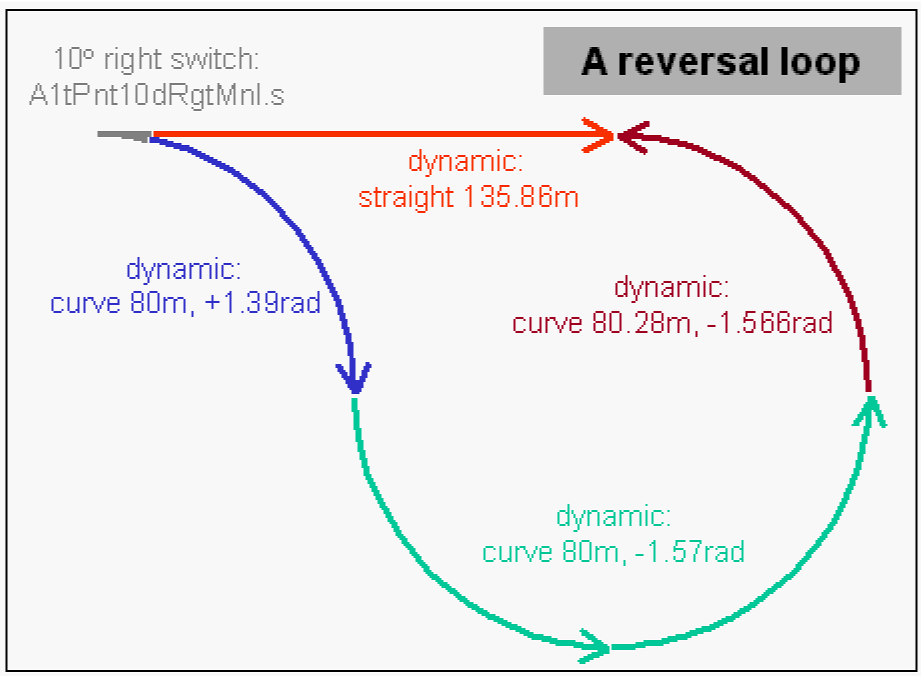
WARNING: It is easy to create a loop without noticing it, because a loop can be very large and may include within itself any number of switches, spurs, sidings and other loops or wyes. Basically, if there is a way to drive out of the curved exit of a switch and somehow come back to the straight exit of that same switch without stopping (thus reversing the train's direction), you have a loop.

## 3.4.4b Laying a reversal loop (balloon)

No matter how you design a loop, you will need to make a joint with **dynamic track** (because the standard track sections will not close a loop, except in very special circumstances).

The simplest way I found to make a simple and "clean" loop uses 4 curved and one or more straight dynamic track sections, as described in this section. Many variations in shape are possible, but will generally end up joining tracks as discussed here.

The result will look schematically like the following diagram. This diagram assumes a basic 80m radius, which is rather tight for long locomotives and cars. The values shown for the red dynamic track sections will vary if you choose a basic radius different from 80m; the arrows indicate the direction of laying track sections; the positive +1.39rad implies a right turn obtained by pressing T).




To achieve this in general (for any radius), do the following:

1) **Lay switch**

At the end of existing track, place a switch with a 10° turn (such as A1tPnt10dRgtMnl.s). Let's say the turn in the switch goes to the right. (If it goes to the left, you will have to reverse all the turns of the later tracks sections, by pressing T.)

2) **Extend the straight track**

Add a 200m long dynamic straight track section to the straight exit of the switch (shown in red in the diagram).

3) **Extend the switch curve**

Continue the curve out of the switch: add one curved dynamic track section of any desired radius (such as 80m) with turn angle of 1.39 radians, and reverse it to turn to the right by pressing T (shown in blue in the diagram).

This turn angle achieves (together with the 10° in the switch) a turn of close to 90°. Avoid a combined turn angle larger than 90°, as it would force you to use an extra dynamic track later on.

4) **Add similar curves turning the other way**

Now add two more dynamic track sections with the same radius, but with turn angles of 1.57 radians, turning the other way (to the left).

5) **Complete the turn**

Next we will add another dynamic track section to complete the turn (shown in dark red). And we will give it a long straight tail after the curve (not shown), pointing straight toward the switch: that tail will only be used to choose the proper radius and turn angle, and will then be removed.

To achieve that, do the following (this procedure is the same as we discussed for joining tracks in section 3.3.11b): add a dynamic track section to the last one; its first segment is curved with the same radius and 1.57 radian turn angle as before; give it a straight second segment of similar length (200m or so) as the straight section you added earlier at the straight exit of the switch.

Then, adjust the turn angle to make the two straight sections parallel to each other (they will still be offset sideways).

Next, adjust the radius to make the two straight sections fall on top of each other (removing the offset).

6) **Close the gap**

You now have two straight sections running over each other: reduce the length of the straight segment of the curved/straight dynamic track section to zero (or a non-zero length if you need extra length to fill the gap).

Adjust the length of the straight dynamic track section that you attached to the straight exit of the switch until it fits the free end of the curved dynamic track.

If the straight dynamic track section is not long enough (its length is limited to 200m), you can add another straight dynamic track section to bridge the gap, or you can again extend that tail from the last curved dynamic track section.

## 3.4.4c Laying a wye

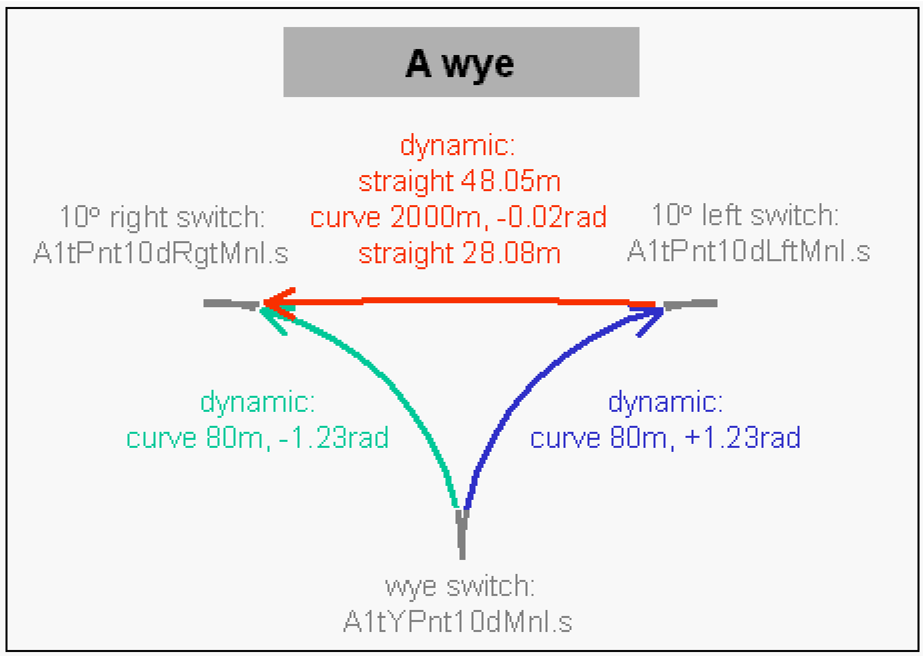
A typical wye (named after its "Y" shape, as illustrated in the diagram below) starts from a straight section that splits through a symmetrical switch (A1tYPnt10d.s or A1tYPnt10dMnl.s) into two branches with switches, which can be linked to each other with a straight section like the top of a "T". But you can freely vary these shapes: the straight section at the top of the "T" may also be curved, the wye need not be symmetrical, you may use normal (asymmetrical) switches, etc. In fact, it seems that the dangers of wyes in MSTS routes are due to misbehavior of the Y-switch itself, rather than to the other parts of a wye: so, using normal (asymmetrical) switches may resolve problems.

(It is actually quite difficult to build a symmetrical wye in MSTS with a straight "T" top: most attempts will end up with a slightly curved "T" top, as illustrated below.)

The way I build wyes is to **start forming the "Y" shape and then add the top of the "T"**. The top of the "T" will not be quite straight in practice, and can be built with dynamic track sections as in section 3.3.8.

So **first create the two curved sections of a wye** (you can start from a top corner or the bottom of the "Y"), using switches and curve radii that you wish. **Then fill in the missing top of the "T" with one or more dynamic track segments**, by joining the tracks coming from the two switches.

Here is one example of such a wye, built by starting at the wye switch; it is based on 80m radius curves (2000m for the "T" top) and 10° switches (the arrows indicate the direction of laying track sections):



NOTE:  You can also make a slightly different kind of wye at the end of dual tracks (instead of the end of a single track as illustrated above):  let the two dual tracks split apart (no switch is needed), one turning left and the other right, by about 80°;  add a switch to each (similar to those in the above diagram);  and connect them as shown above (you will need different values for the red connecting dynamic track).  You may use 45° yard curves (like a A1t45dYardCrvLft.s curve with a A1t45dYardRgt.s switch, and the corresponding pair for the other side;  these have a very small 60m radius):  that way you can make exact 90° turns, so that the red connecting dynamic track can stay straight (with a length of 115.28m).

### 3.4.4d Loops and wyes in tunnels

The standard track sections do not include switches in a tunnel, so **switches should remain outside any tunnel you build**.  If you really want to put a switch inside a tunnel, you need to find a way to assemble a tunnel yourself with imported objects that look like floor, walls and ceiling.

**Dynamic tracks also should remain outside tunnels**:  they do not have the option of tunnel walls.  Again, you may be able to construct a tunnel yourself around dynamic tracks, from imported objects.

## 3.5 Terrain shaping

This section assumes that you shape your terrain yourself, rather than importing it from external sources.

If you import terrain (such as from DEM data), you obviously don't need to shape its large features.  However, you may still have to shape its finer features along your tracks, for example by manipulating individual wire grid points (see section 3.5.1).

Terrain shaping allows you to build hills, mountains, valleys, rivers, lakes, seas, and to make realistic terrain around your structures (tunnels, bridges, buildings, etc).

**Large features like mountains and deep valleys are a serious challenge for terrain shaping**, because they involve very large changes in the default flat terrain that RE starts with.  This can also be dangerous work, since a single mouse click or key press can change vast amounts of terrain (check each time what you have done!).  You can approach this challenge in two ways: bottom-up or top-down.

**Bottom-up** is the more obvious choice, given that you start from flat terrain at sea level:  this approach raises terrain toward the mountain tops.  To do this, you can use tracks or the "stamp", as described below.

The **top-down** approach, on the other hand, first creates flat terrain at the level of the mountain tops, and then digs valleys into it (just like rivers eroding their way down).  This approach has some advantages:  it pays more attention to the shaping of the valley bottoms than the mountain tops;  since tracks normally lie closer to valley bottoms, those are rendered more directly, while the mountain tops are rarely even visible from a train, so they can remain rough.  How to raise the terrain to be flat above the mountain tops?  Set the altitude limits as described in section 3.1.5 (the terrain will jump to the upper limit if the lower limit is above 1m).  How to then dig valleys?  Place your first track where desired on the flat terrain, then lower it to its proper height and press Y to carve out a part of the valley;  or use the "stamp" with Set Height (see section 3.5.2) to first carve a deep hole and then lay your first track section within it.

**I recommend the following sequence for shaping terrain:**

- lay tracks, adjusting terrain to their height by pressing Y after each section is laid;

- roughly shape large features (hills, mountains, valleys, lakes and seas) by using a wide and shallow "stamp" (as I call the "Alter Terrain Height tool");

- refine terrain shapes with narrower and steeper "stamps";

- flatten rough terrain;

- fine-tune the terrain point-by-point along all tracks and other structures.

The fineness of detail is in principle set by the Terrain Detail Scaling Factor (in RGE, see section 2.3):  however, I always get an 8m resolution no matter what value I use for the Terrain Detail Scaling Factor.  That resolution is the distance between the closest grid points seen on the wire mesh (viewed by pressing W).

More important:  **for fine resolution, set the "error bias" to zero** for each tile where you need that resolution:  see section 3.1.5.

The **two basic terrain shaping methods** are:

- **use track sections to shape any terrain** (also away from tracks), by pressing Y after each section is laid;  you can delete them afterward;  this is convenient for making valleys and ridges, but especially for shaping terrain close to your real tracks in areas of high track density (such as yards);  the width and slope of the terrain shaping is controlled by the "stamp", the properties of which are adjustable in the Terrain window (embankment, width, cutting):   see section 3.5.4;  you can do the same with road sections, which allow steeper slopes than track sections, but be careful:  adding and removing road sections can cause the route to fail.

- **use the Alter Terrain Height tool** (press F9):  the Alter Terrain Height tool allows you to manipulate individual wire grid points, but also small or large collections of grid points together at once;  its use is described in more detail in section 3.5.2.

A useful command is to **press J:  it refreshes the terrain**, including the sunlight shadows.  You can also **turn the sun with + and -** to vary those shadows.

WARNING:  You may have noticed "dynamic" pop-up slopes in the MSTS default routes or in your own routes - terrain details can change as you approach (with the camera or in a train). Particularly with steep terrain near your tracks, ground may cover your track as seen from some distance, then disappear (or reappear) as you approach.  This is unavoidable:  MSTS seems to change its mind about which grid points to use when imaging terrain, depending on angle and distance of view.  The only way to minimize this disturbing effect is to adjust the terrain so it does not matter which grid points are used for imaging.  In RE, move your mouse around to see the choice of grid points change, and try to adjust individual grid points so MSTS' different choices never cover your tracks.

[NEW SINCE V2] TIP:  There is an easy way to **quickly measure the local terrain altitude**: press F5 (as if to place an object), so the cursor turns to a + sign, and watch the y value shown in the Placement window;  this y value gives the terrain altitude at the position of the cursor (+ sign).  As you move the mouse, this y value will continuously show the altitude at the location of the cursor.

[NEW SINCE V1.106]

### 3.4.5 LINKING DISTANT PLACES WITH STRAIGHT TRACKS

You may be faced with the task of **laying straight track between places that are so far apart that you cannot see one from the other** in RE:  how do you then orient your straight track to join those points?

For example, you may want to link two distant railway stations or two tunnel ends with straight track.  Or you may know the coordinates of successive turns in a railway line, but the distance between them is so large that you cannot see that far in the RE.

We present two approaches to do this:  the first method uses trial and error within the RE, while the second uses an add-on tool to calculate the required track orientation.

One way to lay straight track between distant places is to **use the camera to try out track orientations until you know the desired compass direction;  then lay track in that compass direction**.

You may generate markers along the way as a visual guide.  This can be done by linear interpolation of the coordinates to intermediate positions, such as at the midpoint or the quarter-points, etc.).  Note, however, that RE does not always line up the markers perfectly.

Do this **as follows**:

- place visible objects at both places, such as trees or markers (which you can place at intermediate positions also);  if you have laid track at one place, better not lay track at the other place yet;
- position the camera at one of the two places to be linked (if you have laid track at one place, position the camera there);

- aim the camera toward the other place:  estimate the needed compass direction;
- fly the camera straight toward the distant place;
- if you hit the distant place, you have guessed the right compass direction;
- if you miss the other place, try again with another camera orientation;
- once you have found the correct compass direction, lay your track in that direction.

The camera can be oriented with a precision of about 1°, so you can aim the camera in principle within 0.5° of the desired direction.  If the camera direction is off by 0.5°, you will miss the distant place by about 9 meters for every kilometer of distance (about 48 feet for every mile).  So if you need to cover 10 km, the camera may not be able to hit the distant place closer than 90 meters or so (almost the size of one patch, which is 128 m on a side).

On the other hand, you can orient track with a precision of about 0.06° (that is the finest angle increment available in a curved dynamic track section, 0.001 radians), which is 8 times more accurate than the camera orientation:  so you should be able to lay track that hits the distant place within about 1 meter after 1 km (or 11 meters after 10 km).  To guess the required orientation with that precision, you need to measure by how much your camera missed the target and convert that to a radian angle to adjust the curve of a dynamic track section.

However, whichever way you do this within the RE, you also need to be able to determine the orientation of the track sections that you are laying:  that is unfortunately more difficult, since RE does not tell you the exact track orientation.  One way to determine the orientation of a track section with an accuracy better than 0.5° (which is the accuracy you can achieve by looking at it with the camera) is to attach a dynamic track section to your existing track and make it turn so its end is visually oriented parallel to the tile and patch lines (which are oriented N/S and E/W):  the turn angle needed in the dynamic track tells you the orientation of your existing track end with a precision of about 0.06° (0.001 radians), relative to those tile and patch lines.  A similar method relates a given track section's direction to that of a freshly laid independent section:  laying an independent new track section makes it point exactly to the north, so that joining your existing track to it with dynamic track can tell you their relative orientation.  Either of these ways is certainly labor-intensive, until a better way to do this is discovered in MSTS.

**A more precise way to do this is to use the Object Rotator add-on to this Guide**:  it includes the ability to find the precise orientation needed to lay straight track (or road) between any two points.

## N. Building "First Route":  shaping a tunnel entrance, one grid point at a time

You will next turn your attention to the terrain itself.  You will first learn to **shape the terrain by manipulating individual grid points**, and thereby shape the ground around a tunnel entrance.

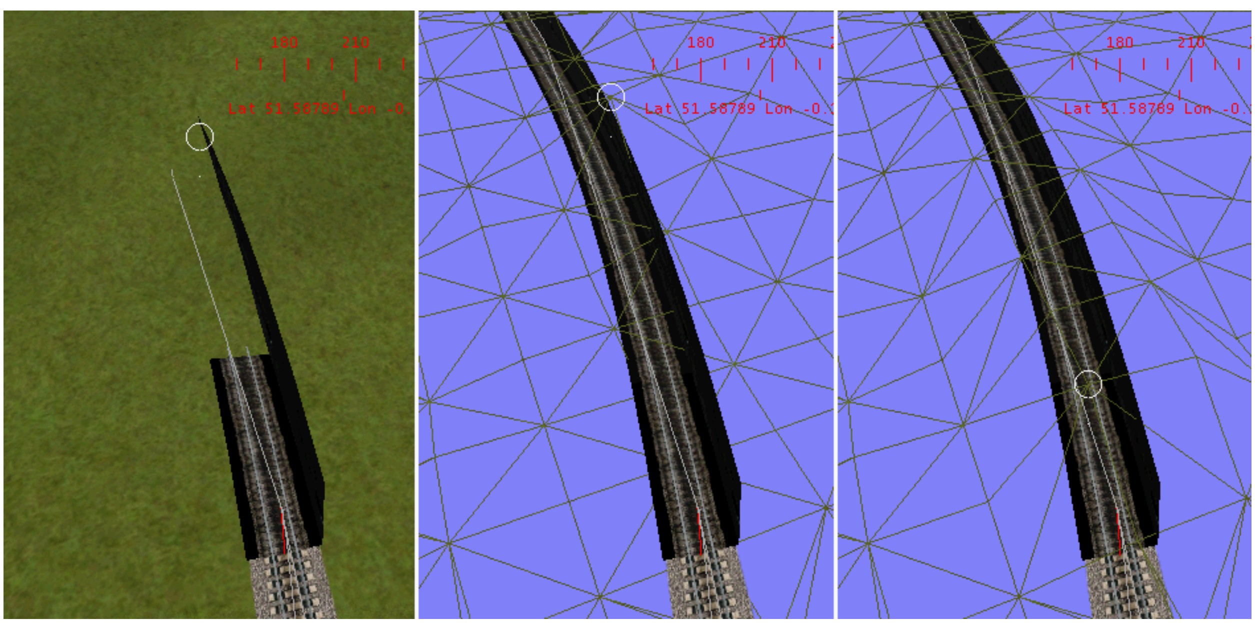
As you will find, this is slow work, and best suited for the final fine-tuning of terrain shapes around tracks, structures like tunnel entrances and other objects.

For more details, see section 3.5.1.

Let's work on the entrances of your underground tunnel.  Move your camera to one of the two tunnel entrances which lead into the hill that you created by laying an overhead spur.  That hill extends roughly to both tunnel entrances, but most likely does not reach the flat ground either within the tunnel or outside the tunnel.  This is illustrated in the left part of the figure below: there the tunnel is not properly buried.  To fix this, we will pull the terrain up, by raising individual grid points of the terrain mesh.  Note that you may not see exactly the same details in your First Route, because they depend on exactly where you laid each track section.

**Do this as follows:**

- press F9:  this puts RE in its Alter Terrain mode;  the mouse pointer becomes a white circle that jumps from one grid point to another as you move the mouse;  in the left part of the figure below, the circle marks a grid point where the tunnel should be buried;
- by tapping on the up-arrow key (on the NumPad), raise that grid point a bit, and repeat this until that part of the tunnel disappears;
- press W to see what happened to the wire grid:  that is shown in the middle part of the figure: you have created a little "pyramid" in the terrain;
- you can lower a grid point by tapping on the down-arrow key (on the NumPad);
- now repeat this for other grid points around the tunnel, working your way to the tunnel entrance;
- the right part of the figure shows one result of such terrain shaping:  this "dike" covering the tunnel is clearly not the final shape you want;  so:
- raise grid points farther away from the tunnel, to form a more natural hill shape.




NOTE:  You can undo the last terrain changes by pressing ` (the back-quote).

TIP: Frequently switch wire-frame mode on and off, to visualize both the grid shape and the terrain appearance (press W).

TIP: Press J every now and then, to recalculate the sun's shadows.

TIP: Vary the sun's shadows by moving the sun East or West: press + or -.

CAUTION: When you repeatedly tap on the up- or down-arrow, or when you keep pressing on those keys, the white circle may jump to another grid point and raise or lower that grid point instead of the one you want! To prevent that, try to follow the grid point with the mouse as you raise or lower it.

NOTE: Save your changes often, so you can easily recover from bad terrain shaping.

**Save your work frequently as you work with terrain, after deselecting any selected track or objects.**

It is obvious that this point-by-point terrain shaping is a slow and painstaking process. We need faster methods to shape entire hills, mountains, valleys, etc.

However, the point-by-point method will still be necessary to fine-tune terrain shapes around tunnel entrances and other structures (objects like bridges, etc.): so this method remains the best for the final fine-tuning of the terrain.

Next we will learn methods that accelerate terrain shaping by adjusting larger groups of grid points together. However, those methods are rougher and thus not suitable for fine detail work.

**To continue building our First Route, jump to the next blue box.**

## 3.5 Terrain shaping

[NEW SINCE V2]
The purpose here is to change the initially flat terrain into hills, valleys, rivers and mountains, as well as to shape the terrain near the tracks to form embankments, cuts, tunnel entrances, etc.

One aspect of terrain shaping is defining the lower (floor) and upper (ceiling) limits of the terrain heights: this needs to be done individually in each tile, as described in section 3.1.5. In relatively flat terrain, you can define a floor and a ceiling that are close together, while in mountainous terrain you will need to define widely separated floor and ceiling.

If you wish to import real-world terrain elevations (DEM data), see section 2.6.

### 3.5.1 MANIPULATING INDIVIDUAL WIRE GRID POINTS

You get the finest control over terrain shape by manipulating individual wire grid points: use this for your final fine-tuning of terrain shapes, especially around your tracks, tunnel entrances and other structures.

**After pressing F9**, the mouse moves a little circle around from grid point to grid point (see the grid points by pressing W).

**While the circle is located at a grid point, you can press the up- and down-arrow keys (of the NumPad) to lift or depress that grid point alone** (be careful: you may lose the grid point and automatically switch to another grid point, unless you follow the grid point with the mouse).

Simultaneously pressing **Shift accelerates the motion**: but that will often move grid points too fast!

You may have to adjust the camera's viewing direction and distance to "grab" the grid point you want. In particular, it may be hard to grab a grid point beyond which are other grid points (as when looking through a hill to other terrain behind it).

**You can also set the height of an individual grid point numerically**: point at it, right-click, select Set height…, then enter a height in meters.

Obviously, this point-by-point approach is too time-consuming for shaping larger areas.

**O. Building "First Route": shaping the river, and covering the underwater tunnel**

You will next learn techniques of terrain shaping that manipulate larger groups of grid points at once: this will accelerate the process. You will apply these techniques to produce a river, and also to cover the tunnel that runs underneath it.

You will here learn three techniques: **using tracks to shape terrain**; **setting the height of a rectangle of grid points**, and **"flattening" or smoothing terrain**.

For more details, see sections 3.5.2-4.

One of these techniques you already know: **use tracks to shape terrain** around them, by making the terrain adjust to them (pressing Y). With this technique we made the trench for the underwater tunnel, and in fact exposed water in that process, as if making a riverbed: that is what we will do! It turns out that it is quicker to **simply throw unconnected tracks** along the riverbed, rather than to make a connected railway line (which we would remove anyway). The illustration below shows how to do that: you can guess the drowned track sections by their white overhead lines, and you can guess the location of the river by the darkening of the grid lines. Later you can delete those track sections.

NOTE:  You can use the same technique to create a valley in a hill or mountain.

[NEW AFTER V2] CAUTION:  You may also use road sections to shape terrain, but do this only if you already have placed some permanent roads in your route.  It is reported that the route will fail if you delete ALL the road sections in your route (requiring a track database rebuild that can be difficult in this case).

**Do this as follows:**

- put your camera over the river area shown at bottom left of the figure below (this is East of the underwater tunnel, looking West, but you can also start on the West side, looking East);
- in the Terrain window, set embankment 45, width 25, cutting 35;  the most important choice here is the width:  it gives the approximate width of the riverbed - choose what you need;
- place a track section (50m straight is fine) near a distant corner of the river, over a "wet" patch (where water is toggled on):  see the track section at bottom left of the figure below;  since the terrain there is at 1m of altitude, this track section starts above the water level;
- press F4;
- rotate the track section left or right (by dragging the mouse left or right) in the direction you want the river to go;
- dip the track down (by dragging the mouse down, opposite to what you are used to!) to make it slope underwater;
- press Y:  this forms a lake at the deep end of the track section;
- delete the track piece (it has not been removed in the figure, so as to more clearly show where to place the next tracks);
- now place the next track section in the water on the riverbed;  turn it to where the river should go;  then press Y;  (unfortunately, you can't copy and paste a track section);
- repeat the last two steps to trace out the river;
- if you wish, delete the track sections.

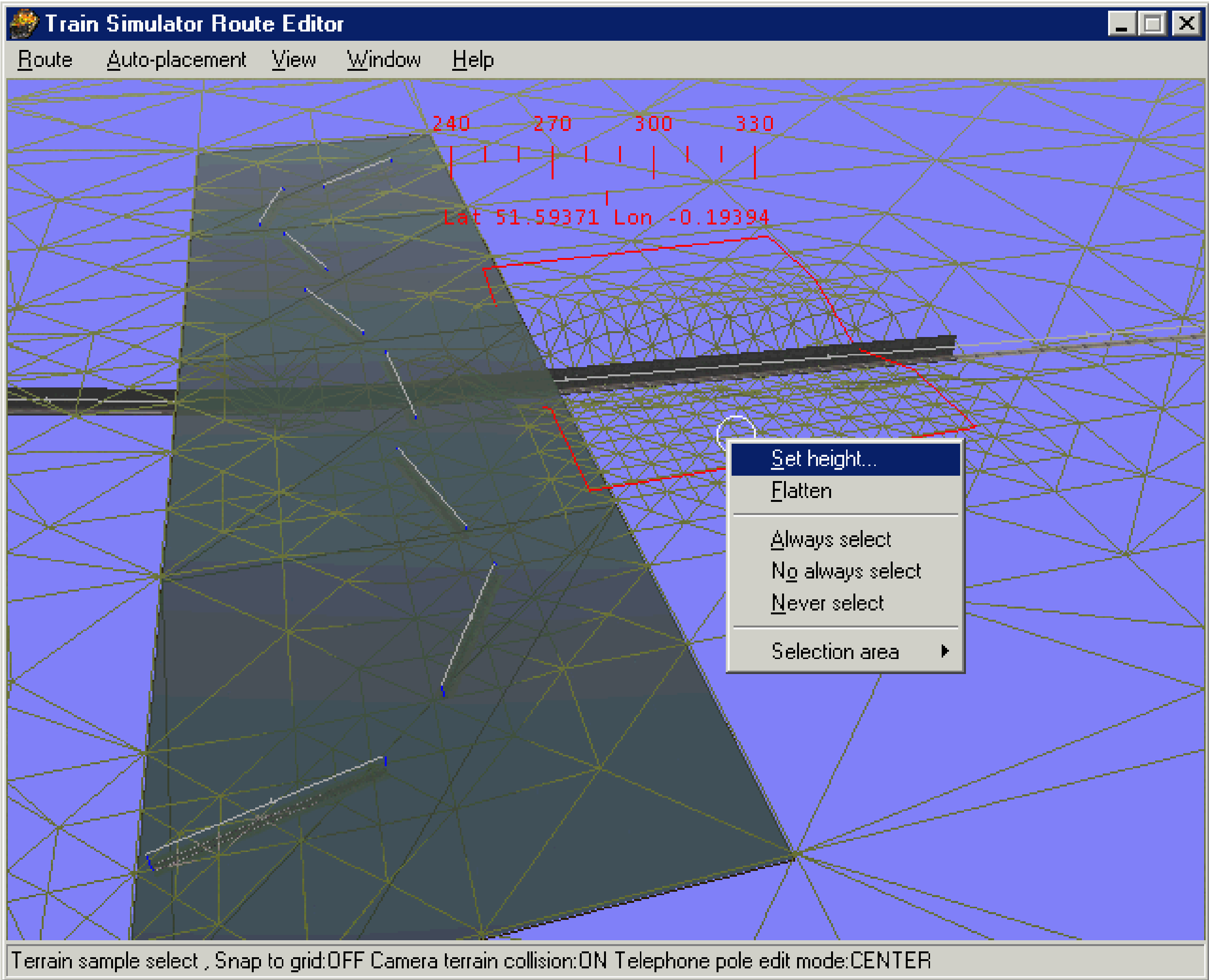
CAUTION:  The terrain shape that the camera sees can vary with the distance of view:  if you don't see what you expect, zoom in.  This is true in wire-frame mode (press W) as well as normal mode.

NOTE:  If you don't like what you get, press ` (the back-quote) to undo the last terrain change; then reorient your track section, or delete it and place another one elsewhere.

NOTE:  If you want to raise terrain again out of the water, just place a track section on the banks of the river and make it hang over the water to pull up terrain.

IMPORTANT:  **Save your route frequently, after deselecting any selected track or objects (after a few track sections have been placed and removed).**  When shaping large pieces of terrain, a lot of terrain elevation data are changed, so you need to save them frequently. Otherwise, the RE will slow down, and may even get confused, leading to big trouble.

Note that this process automatically covers up part of the underwater tunnel (if you keep the riverbed above the tunnel ceiling). You could in fact cover up the entire tunnel this way, using track sections placed on the banks of the river. But, in this case, we will use a different method, as follows.




Another terrain shaping technique is illustrated in the same figure (above): **set the height of a rectangle of grid points, by typing in an altitude (positive only!)**. This is of course most convenient in flat terrain. That is the case here: we want to cover up the tunnel away from the river, restoring the flat terrain.

**To do this:**

- press F9: this puts RE in its Alter Terrain mode; the mouse pointer becomes a white circle;
- drag the white circle to define a rectangle: it will be outlined in red, as shown in the figure above;
- right-click to open a pop-up menu: click Set height…;

- an altitude will be offered (it seems to be the average over the altitudes at the 4 corners of the rectangle): accept it or type in another; in this example, accept a 1 or type a 1 (for an altitude of 1m).

The result will be as shown at the left in the figure: the terrain becomes flat again, and covers the tunnel.

TIP: Use a big rectangle at first, then smaller rectangles to take care of smaller areas (such as near the tunnel entrances).

CAUTION: Sometimes, the height gets reset well outside your rectangle. If this reaches too far, press ` (back-quote), save the previous terrain for safety, and try again with a smaller rectangle.

TIP: If you don't know the local altitudes to type in, you can use this technique to find them: define a small rectangle, right-click, select Set height…, and read off the average local altitude; then cancel.

TIP: **If you press D** after defining a rectangle, **that red rectangle will stick to your mouse pointer, so you can move it around**.

**Save your work, after deselecting any selected track or objects!**

A third terrain shaping technique is also easily illustrated with the above figure: **"flatten" or smoothen terrain, which is much like "ironing"**. This removes all sharp corners and rounds off all curves in the terrain. It is very useful for ironing out rough spots, and for removing the appearance of "blocky" terrain (for example, you can make river banks less straight this way).

**To do this:**

- press F9;
- drag the white circle to define a rectangle (as shown in the figure);
- press F: this smoothens the terrain inside the rectangle;
- press F several times to smoothen further.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will learn to use some effective ways to make larger terrain features like hills, mountains and valleys.

**To continue building our First Route, jump to the next blue box.**

## P. Building "First Route": making hills, mountains, and valleys with the "stamp"

Next, you'll learn other techniques of terrain shaping that help in making larger shapes, like hills, mountains and valleys.

You will here learn to **use the "stamp" to shape terrain**.

For more details, see sections 3.5.2-4.

You have already used a rectangle to change large collections of grid points at once. This same rectangle can be very effectively used as a "stamp" to shape the terrain: it can push pieces of terrain up as well as down. You can make the stamp wide to shape gentle hills or sharp for mountain peaks, and anything in between.

We will use this technique to build up the hill through which we have placed an underground tunnel.

CAUTION: The first time you try out this method of shaping terrain, you may want to stay far away from your tracks! That way you do not risk damaging the terrain shapes that you already have. But here I will illustrate using this technique near your tracks to form the hill over your underground tunnel.

**Do this as follows:**

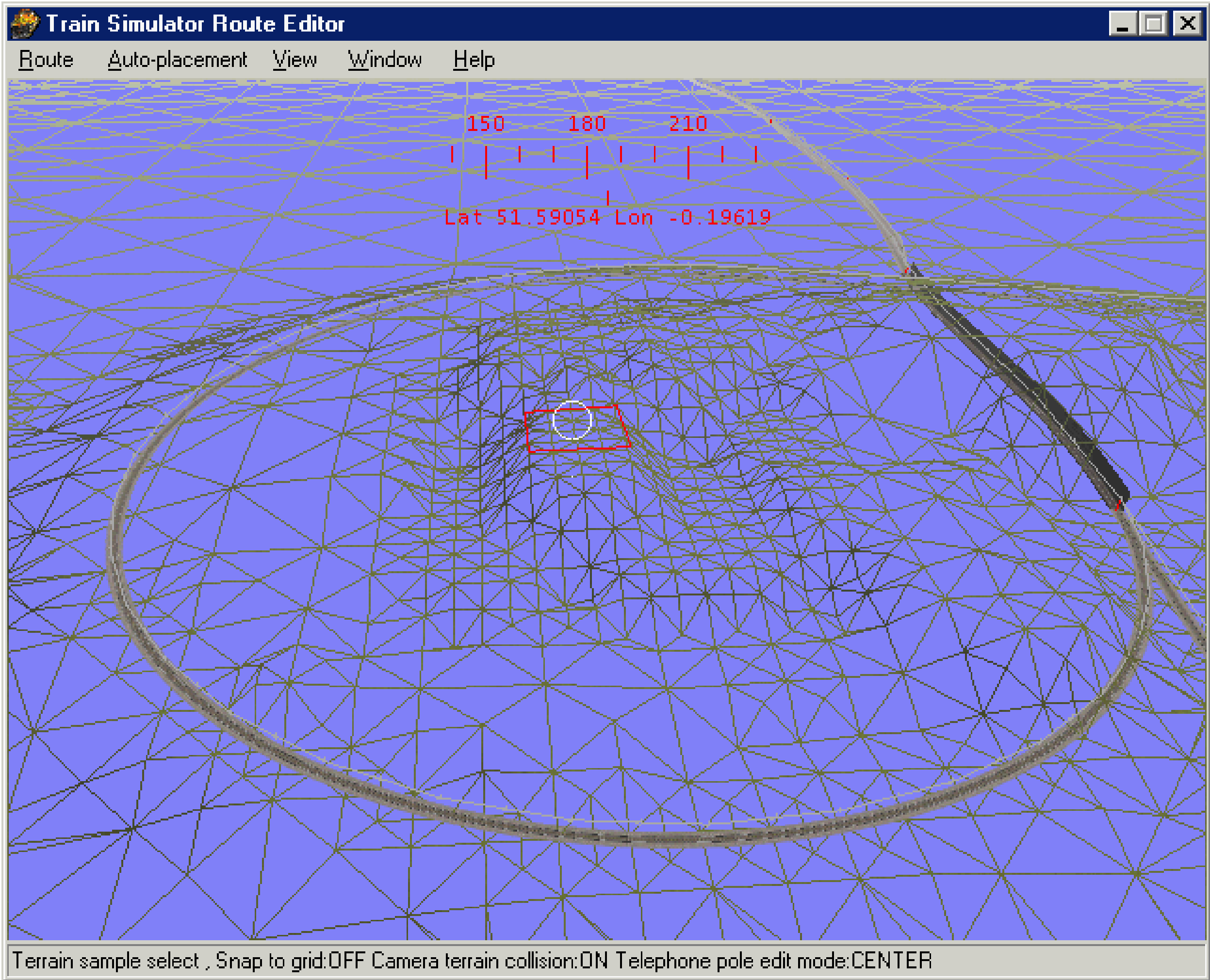
- put your camera so it overlooks the looping spur from the North, as shown in the next figure;
- in the Terrain window, set Embankment 45, Width 18, Cutting 45 (if not already set);
- press F9: this puts RE in its Alter Terrain mode; the mouse pointer becomes a white circle;
- drag a rectangle around terrain that you want to raise: it will be outlined in red;
- press the up-arrow key (on the keypad): all the grid points within this block will adjust to form a flat "table top";
- press the up-arrow key again and again until you get the desired altitude; note the sloping hillsides;
- repeat this process for another rectangle, to create another table-top hill; and repeat again.

Don't worry about raising terrain slightly above tracks, or pushing terrain below tracks: we will fix that later. But do bury your tunnel!

This way, you create a landscape like the one shown in the figure below: it is a hill, but it is terribly "blocky".

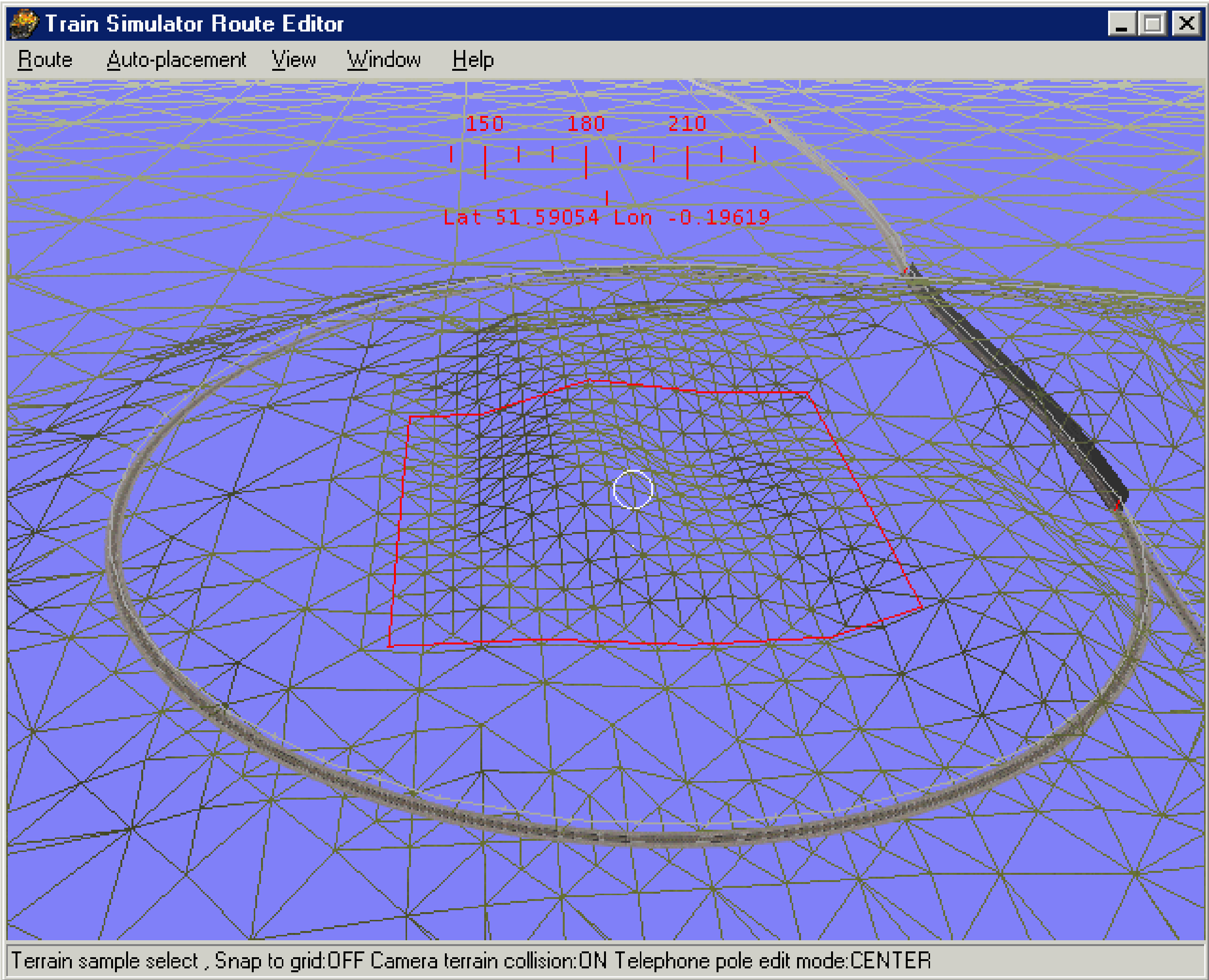

**Save your work, after deselecting any selected track or objects!**

To reduce the "blockiness" of this hill, continue raising terrain with smaller rectangles (you can also depress terrain with the down-arrow key). You may get a hill that looks like this (in wire-frame mode):




**Save your work, after deselecting any selected track or objects!**

Now apply the miracle of **smoothing or "ironing" terrain** (RE calls it "flattening"): as before, drag the mouse to form a large rectangle, then press F. This rounds off all corners. You can smooth the same area again and again to make it smoother and smoother. The result after smoothing is much more realistic, as shown in the next figure.




TIP: If you press D after forming a rectangle, you can then move the rectangle around with the mouse and press F as it passes over different parts of your hill. This allows great speed in smoothing large areas. (Press D again to release the rectangle from the mouse.)

NOTE: After you get out of wire-frame mode, press J to update the shadows; and press + or - to change the sun's direction.

**Save your work frequently when adjusting large numbers of grid points, after deselecting any selected track or objects!**

A closely related technique works much like **"spreading whipped cream onto cake"**, because it allows you to raise terrain almost continuously as you move the mouse across it. **It works as follows:**

- press F9;
- drag the mouse to define a relatively small rectangle: it is outlined in red;
- press D to glue the rectangle to the mouse;

- move the mouse (and the rectangle) slowly over the terrain that you want to raise, and click the up-arrow key repeatedly; you may even keep the up-arrow key depressed continuously (as if your "iron" were blowing steam continuously); you can do this with the down-arrow key as well, to depress terrain.

This "whipped cream" technique will quickly (sometimes even too quickly!) raise pieces of terrain over a large area. Also, if you use a very small stamp, it can roughen up the surface quite fast! This actually helps to randomize the shape of a hill. By "ironing" (pressing F) you can later smoothen it again.

It can be very useful to **vary the stamp dimensions and slopes (embankment, width, and cutting)**: use a shallow embankment and cutting for low, wide hills, and a steep embankment and cutting for high, sharp mountain tops (the width is not used).

**Save your work, after deselecting any selected track or objects!**

After this type of terrain shaping, you still need to **fix up the tracks**, since you probably inadvertently changed terrain along tracks (that is unavoidable with these methods): all you need to do is select each track section in turn and press Y to adjust the terrain to fit under each track. You may want to adjust the embankment, width, and cutting before you do that, choosing a relatively steeper embankment and cutting.

Now you have all the techniques you need to make hills under and around your tracks. It is a good idea to add hills around the area of your route, so that the view from the train is not simply a flat plain extending empty to the horizon. Simple hills look much more natural.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will add a few useful objects along the tracks, such as names, signals and mileposts.

**To continue building our First Route, jump to the next blue box.**

### 3.5.2 MANIPULATING COLLECTIONS OF GRID POINTS WITH THE "STAMP" (ALTER TERRAIN HEIGHT TOOL)

The Alter Terrain Height tool allows you to select a rectangle of grid points and move them up or down together. At the same time, it will drag along neighboring grid points to avoid steep slopes. **It acts like a stamp: it will imprint a rectangle** (upward as a hill, or downward as a depression), **with sloping sides**. The slopes are controlled by the values entered into the Terrain window (embankment, width, cutting): see section 3.5.4.

To apply this tool, you first **press F9, and then select a rectangle of grid points by dragging the mouse** (or by right-clicking and selecting a desired Selection area). **Then you can raise or depress the terrain by pressing the up- or down-arrow key, perhaps repeatedly.**

NOTE: The first arrow-key press seems to produce not a higher or lower terrain, but an average horizontal version of the previous terrain. Press again to raise or lower the terrain from there.

Simultaneously pressing **Shift accelerates the motion**: but that will often move grid points too fast!

**You can also set the height of a collection of grid points numerically**: select a rectangle, right-click, select Set height…, then enter a height in meters.

TIP: **You can move the selected rectangle around with the mouse after pressing D** (this glues the rectangle to the mouse pointer); pressing D again detaches the rectangle from the mouse.

**A convenient way to quickly and roughly make large shapes** (such as hills, mountains, valleys, lakes and seas) is to first repeatedly apply a wide stamp with shallow slopes, as you move it around with the mouse (after pressing D). This will produce unnatural blocky terrain, with flat tops. To make this look more natural, apply a narrower stamp with steeper slopes: this allows you to roughen up the surface. You can repeat this with an even narrower stamp for sharp mountainous features. For speed, glue the rectangle to the mouse (by pressing D), and sweep the rectangle while pressing the up- or down-arrow (as if ironing clothes).

This will probably result in too much roughness: you can "flatten" (smoothen) the terrain as described in section 3.5.3.

CAUTION: This "roughening" process can "tear up" the terrain at the borders of tiles, resulting in gaps through which you see the white underworld. If that happens, flatten (smooth) the tear.

### 3.5.3 FLATTENING (SMOOTHING) TERRAIN

The flattening option would better be called smoothing option (it does not produce horizontal terrain, but rounds off sharp corners).

You can **smoothen terrain by selecting a rectangle, and pressing F**. You can quickly smoothen large areas by choosing a large rectangle and pressing D to move it around with the mouse, repeatedly pressing F as you go.

WARNING: If you smoothen terrain near tracks, you will need to fine-tune the terrain there; for instance, you could then select each track section and press Y to adjust the terrain to it (set the width and slopes of the "stamp" as needed); then you can fine-tune further point by point if needed.

### 3.5.4 CONTROLLING THE WIDTH AND SLOPES OF THE "STAMP" (ALTER TERRAIN HEIGHT TOOL)

You control the properties of the "stamp" with the Terrain window: you can enter values for the "embankment", "width" and "cutting":

- the "**embankment**" is the slope (in degrees) of raised terrain (such as the "dike" upon which track rests after the Y key is pressed);
- the "**cutting**" is the slope (in degrees) of depressed terrain (such as the "trench" within which track rests after the Y key is pressed);
- the "**width**" is the width (in meters) of the flat horizontal part of the stamp (such as the flat terrain on which rests the track after the Y key is pressed).

[NEW SINCE V2]

### 3.5.5 MAKING TERRAIN COVER THE TRACKBED

By default, the MSTS tracks always show the underlying trackbed, including ties. You may, however, want to make **the trackbed and ties invisible**, for example where dirt has covered them up, or where the space between the rails has been filled in stations or streets, so people and vehicles can more easily cross them.

This can be easily done for horizontal tracks by lifting the terrain so it lies just under the top of the rails. In the next illustration, the darker terrain in the background has been raised to almost the level of the top of the rails. This is done with the stamp, as discussed in section 3.5.2, by entering a desired height with the Set height option. You will have to choose a height by trial and error, especially as the Route Editor may change your choice slightly. Once you have found a good height, save its value (with Ctrl-C after selecting it in the Set height window) and paste it into the Set height window for every desired piece of terrain.

Another method is more effective for sloped tracks (and works also on flat terrain): first lay track and adjust the terrain to it (by pressing Y). Then lower the track into the terrain until only the top of the rails remains visible. But make sure all track sections still connect properly!

NOTE:  The above picture also shows an alternative approach at bottom left:  covering the track with road sections, in this case a dirt road section.  To make roads cover the trackbed, see section 3.8.4.

(In the above picture, the dark ground texture is obtained by placing a transfer, OETranstexMM.ace, along the length of the tracks.  The platforms were made with JP2Wall50m.s sections, lowered into the ground until only their tops remain visible.)

[NEW SINCE V2]
### 3.5.6 RECOVERING BACKED-UP TERRAIN SHAPES

You may want to **cancel already-stored changes of the terrain shape** (for instance if you change your mind about terrain shaping), **or recover from accidental damage to your terrain** (unasked, the RE may occasionally make large changes to your terrain, such as creating a mountain or depression where there was none).

To fix this, you can recover earlier terrain from a previous backup.  Since the terrain shape is stored in the TILES folder for each tile separately, you can recover all old TILES files, thus restoring your terrain to what it was when it was backed up.

But you may want to restore only one or a few tiles to their older terrain shape, for example if you want to keep more recent changes on undamaged tiles.  The trick then is to find which files correspond to which tiles.  One way to do that is to find the tile labels with the RGE and thereby identify the tile names in the TILES folder:  see section 2.7.  If you don't know the tile labels, you can use the RE to make a simple change in the one or more tiles that were damaged:  it is sufficient, for example, to raise one terrain point by a few centimeters in a tile - any change will do.  Then save the route and go look in the TILES folder which files have been updated during this save (by checking the time attached to them):  that tells you which files are damaged, and therefore which files you need to restore from your backup.  You should also check for similar effects in LO-TILES (since I never used the LO-TILES, I am not sure about this).

## 3.6 Water

Water can be made visible whenever the terrain dips below the local water level:  see the illustration in section O, which depicts a river extending over four "wet" patches.

**To see water when you run MSTS** (as opposed to RE), **make sure its option to display water is checked on** (as it is by default).

In RE, **to make water visible, you first must toggle water on in individual patches** of a tile:

- **press F7** to see the tile and patch grid;

- **left-click within a patch** (with black borders);  if you want to select a rectangle of patches, press Shift while selecting the patches at two opposite corners of the rectangle:  the patch border(s) will turn red;

- **right-click** to open a pop-up menu;

- **toggle water on (or off)**.

Next **you may reset the water level**, which is done for a whole tile at once:

- **press F7** to see the tile and patch grid;

- **left-click within a tile** (with blue borders):  the borders of one patch in that tile will turn red;

- **right-click** to open a pop-up menu;

- **select Tile water height**;

- **enter the desired water heights** at the four corners of the tile (SW, NW, SE, SW), in meters above/below sea level;  the option to have varying water levels allows you to make sloping rivers and streams.

Now water will become visible where the terrain dips below the water level.

NOTE:  The water surface in MSTS is actually composed of 3 levels, which give the impression of seeing a little bit below the surface.

NOTE:  A train can drive in and under water.

[NEW SINCE V2] TIP: To help you decide at what altitude to set the water height, you can **quickly measure the local terrain altitude** in the following way: press F5 (as if to place an object), so the cursor turns to a + sign, and watch the y value shown in the Placement window; this y value gives the terrain altitude at the position of the cursor (+ sign). As you move the mouse, this y value will continuously show the altitude at the location of the cursor.

## Q. Building "First Route": adding objects along the tracks - names, signals, speed limits and mileposts

You will next learn to add some useful objects along your tracks. These will be helpful to the train driver. Note that it is best to finish shaping your terrain before you add objects!

First **add platform names and siding names** (see section 3.7 for more details). These will be visible to the train driver by pressing F6 in MSTS: they will show up as blue poles flagged with the names you set. Nothing else will be visible! A platform is a separate object that must be imported (because none is available by default in a new route) and placed (see section 3.10).

### To place platform names, do this:

- move the camera to the tracks of your Central London departure station;
- press F5;
- select Interactive object "Platform" in the Placement tool;
- point in the middle of the western platform track (point exactly between the pair of rails), and left-click: this places a pair of green "handles" on the track (similar to the handle of dynamic tracks);
- while one of the green handles is selected, press F6 or right-click to open a pop-up menu, and enter the station name (Central London), as well as a platform name (Central London 1); ignore the other options;
- do the same for the other platform track, but call this platform Central London 2.

You can repeat this at the North London station.

WARNING: If you try to place the platform name off the track, RE will complain, and, when you try again, it will force you to reselect the Platform in the Object selector.

### To place a siding name, do this:

- move the camera to the siding (just North of the first hill in your First Route);
- press F5;
- select Interactive object "Siding" in the Placement tool;
- point in the middle of the siding (point exactly between the pair of rails), and left-click: this places a pair of yellow "handles" on the track;

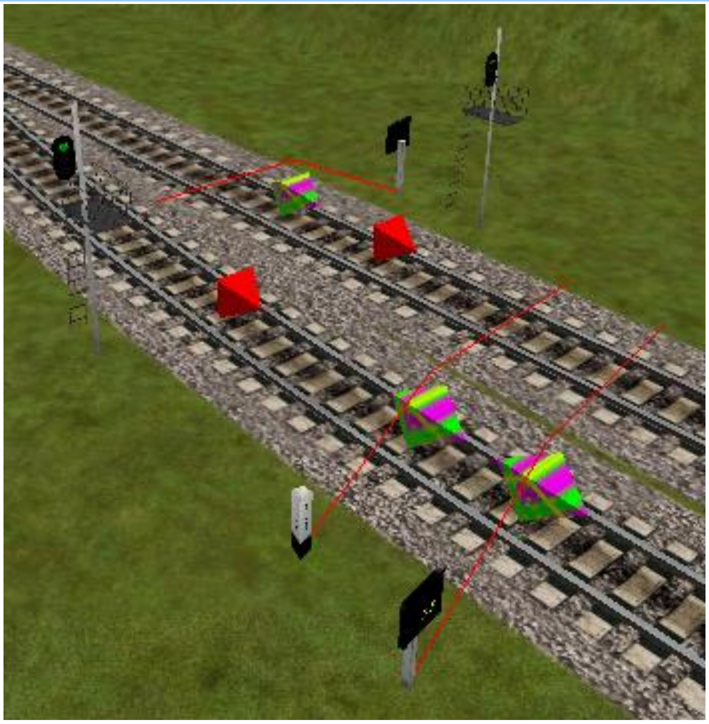
- while one of the yellow handles is selected, press F6 or right-click to open a pop-up menu, and enter the siding name (let's say "River siding", because we will later add a river near here); ignore the other options.

**Save your work, after deselecting any selected track or objects!**

Next **add signals**. By default RE includes one type of signal, which you can place on the left side of the track. Now put a signal at each of the exits of the platform tracks of the Central London station, and another before the entrance to the North London station, as follows.

- move the camera to near the switch at the end of the Central London departure station;
- select the Track object "Japan 3 Light Signal (Left)";
- point before the switch at one of the track sections (point exactly between the pair of rails), and left-click: this places a red "handle" on the track (see the two examples in the figure below on another piece of track; ignore the two speed limit signs and milepost, with their more colorful handles); this also places the signal pole itself right between the rails;
- press F3;
- point at the desired position of the signal itself (on either side of the track), and left-click;
- if needed, rotate the signal itself (press F4, then, while pressing Ctrl, drag the mouse left or right);
- press H so the signal rests on the ground: the result looks like that of the next figure (again ignoring the other signs);
- repeat this for the other platform track;
- repeat this before the switch that leads into the North London station.

NOTE: Don't place more signals at this stage, because we will need to first place mileposts using just one or two signals - see below.



**Save your work, after deselecting any selected track or objects!**

Next **add speed limits**. By default, RE provides 3 speed limit signs, including one speed warning sign and a resume speed sign (the warning speed sign does not work). Let's say that we want to limit the speed upon exiting the Central London station to 35 mph until the level crossing, and then increase it to 45 mph for the rise up the hill (you can add more along the route). **Do this as follows:**

- select the "Japanese - Speed limit sign" in the Speed limits class;
- press F5;
- point and left-click between the rails just beyond the switch that terminates the Central London station: a multicolored handle shows up (two such black speed limit signs are shown in the figure above, with their multi-colored handles, on another track); also a small sign is placed right between the rails;
- press F6 or right-click, then specify the desired speed limit (such as 35 or 45) in miles or kilometers per hour (uncheck the MPH box for km/h), and click OK;
- press F3;
- left-click where you want the sign itself placed off the track;
- press H to rest it on the ground;
- press F4;
- rotate the sign to face oncoming traffic: the result looks like that of the figure above;
- repeat this to place a speed limit sign (for 45 mph) soon after the level crossing.

**Save your work, after deselecting any selected track or objects!**

Next **add mileposts**. There is one difficulty in doing this: **Where to put the mileposts?** How will you know the distance from the reference point ("milepost 0")?

The easy way out is to put no distance information on the mileposts (leaving them blank), and to consider them purely decorative (they are hard to read anyway!). Then do this:

- select the Track object "Japanese - Milepost";
- point at a spot exactly between the rails, near where you want to place a milepost;
- left-click to place it: you will see a multicolored handle (see one in the figure above) and, if you look close, a small white post under it - that is the milepost itself;
- press F3;
- point and left-click away from the track where you want to place the milepost itself;
- press H, to make it rest on the ground: the result should look like that in the figure above;
- repeat for other mileposts.

**One general way to determine where to put mileposts** is to first place a signal (if none is placed yet) at "milepost 0" (don't include other signals yet). Then you can drive a train along the route in MSTS toward that signal while watching the Train Monitor: it will tell you the distance to "milepost 0". Press 0 (zero) to show the "head-up display" (the compass with latitude and longitude). Now, when the Train Monitor indicates a distance where you want a milepost, note the latitude and longitude (the quickest way to note these is to do a screen grab while you drive, by pressing PrintScreen, which records all the information you want; you will find the various screen grabs as PCX files in the main Train Simulator folder, with names scrngrb*.pcx, and you can open them with most graphics viewers). Next, you can place the mileposts in RE at those positions.

Let's put mileposts along the main tracks from Central London to North London. "Milepost 0" will be the exit of the Central London station. So we need to place a signal there that a train approaching from North London can use to measure the distance to "milepost 0" with the Track Monitor of MSTS.

**The first step is to place a signal at "milepost 0"**, at the North end of the Central London station, facing North: do that now in RE, and exit from RE. Make sure you have an activity that starts a train at North London.

**Then do as follows:**

- start MSTS and (if you have an activity) run the route from North London to Central London while watching the Track Monitor for the distance to the next signal;
- make sure the Track Monitor is active (press F4), as well as the "head-up display" (press 0);
- let’s say we want to place mileposts at fifths of miles (0.2, 0.4, …), starting with 0 at Central London;
- each time the distance shown by the Track Monitor changes to the next two tenths of a mile, press PrintScreen, or stop and record the longitude and latitude shown by the "head-up display";
- now return to RE;

- place mileposts at the positions that you noted or recorded with PrintScreen (and set the proper numbers 0.0, 0.2, 0.4, …); for this, select "Japanese - Milepost" in the Track Objects; press F5; point between the rails near where you want the milepost to be; press F6 or right-click, and enter the Milepost number (0.0, 0.2, 0.4, etc.); uncheck "Show dot" for whole miles like 1 and 2, so they will show as 1 and 2 instead of 1.00 and 2.00; 0 will show as blank); press F3; point at the desired off-track location, and left-click; then press H so the milepost rests on the ground; if needed, press F4 and rotate the milepost so it faces oncoming traffic.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will add other objects, such as telephone poles, forests and bridges.

**To continue building our First Route, jump to the next blue box.**

## 3.7 Interactive track objects

### 3.7.1 PLATFORM AND SIDING NAMES

**Names placed at specific locations like platforms, sidings, etc., are convenient guides for the train driver**: by pressing F6 during driving, these names will be visible in MSTS. Sometimes, this is the only way a driver has for telling where he/she is. These names can also be used in Activities.

A platform name includes a station name. Station names are used in Activities to define stops along a path, while platform names are also displayed by the Activity Editor.

You place such a name just like you do a track section or another object (it is part of the Interactive Objects class), but you place it in the middle of existing track.

[NEW SINCE V2] Platform and siding names are limited in length to about 2 km, meaning that you may not make the length spanned by such names (the length of the green line) stretch to more than 2 km, a length which you should never need. The length of a platform name determines where a train should stop to have a successful station stop in an MSTS activity. A siding name has a similar purpose for dropping off or picking up wagons in an activity.

WARNING: **Do not let a platform name or siding name cover a switch.**

WARNING: **A Platform Interactive object is not a platform!** A platform will have to be separately imported (because none is provided by default in a new route) and placed as an object (see section 3.10).

**To place a platform name**, you select the Interactive object "Platform", and place it between the rails of a track section.

When a Platform Interactive object is placed, you see two green "handles" with a green connecting line that is kinked to one side. You can select either handle, then press F6 or right-click, and specify a station name (for example: Central London) and a platform name (for example: Central London 1, or Central London 2). You may also specify on which side an actual platform object will be located (that is the side on which passengers will get on and off; both sides are allowed simultaneously): the kinked green line segment should point to that side (or both sides). You can drag the two green handles to mark the platform ends (if they are known yet): press F2, select one of the two handles, press F3, and drag the handle to a platform end; repeat for the other platform end.

**To place a siding name**, you select the Interactive object "Siding", and position it between the rails of a track section. Press F6 or right-click, then enter a siding name.

A siding name can of course be placed on a siding, but you may also place it on any existing track section: thereby you can label any interesting spot, such as a bridge, a tunnel, a loop, a wye, a pass over a mountain, etc. It will only be seen by the train driver after pressing F6 in MSTS. The Siding name Interactive object shows up as two yellow "handles" with a yellow connecting line; the name itself is typed into its properties (opened by right-clicking after selecting one of the handles). After selecting a handle and pressing F3, you may drag the two handles to the ends of the siding (this is useful in Activities).

CAUTION: If you try to place a platform or siding name too far off the track, RE will tell you so: try again, and reselect it again from the Object selector.

### 3.7.2 YARD DEFINITIONS

**When a train is in a yard to which a "yard definition" is applied, the driver can take a bird's eye view of the entire yard**: by pressing 7 in MSTS and using the arrow keys, the driver can "fly" over such a yard to view it better.

A yard definition is applied within RE, and is then usable in MSTS. To apply a yard definition, do this: select the "Yard definition" in the Track Objects class; place it (like any object) in the center of a yard: a red box shows up (similar to that of a forest), with three green handles; select one of the handles on a side of the red box, press F3 and drag it to stretch and turn the red box so as to cover as much as possible of the yard; select the other handle on the other side of the red box, and drag it to further adjust the red box size and orientation; if necessary, select the central handle (in a similar way), and move the red box horizontally.

[NEW SINCE V1.106] The orientation of the bird's eye view is determined as follows. Suppose you look vertically down on the yard definition box in RE, so that one of its green handles points to the left of your screen and the other to the bottom of your screen: that is the view orientation that will be shown in MSTS.

[NEW SINCE V1.106] You may change the "mobility" of the yard camera by changing an MSTS file (see Appendix J): this allows the user to move the yard camera anywhere in the entire route (outside yard definition areas), and to orient the camera in any desired direction. This way, you can watch one train while driving another, or you can watch distant signals as your train moves.

[NEW SINCE V1.106] NOTE: You may place a yard definition anywhere you like. For example, you could place a series of overlapping yard definitions to cover a longer interesting portion of your route: then the player can view the train from above along that entire portion. But be very careful: accidents have happened during route building after placing continuous yard definitions, so be prepared to return to an earlier backed-up version. When you overlapping two yard definitions, it is helpful to orient them roughly the same way (with their green handles in roughly the same orientation), so that the bird's eye view does not change its orientation too abruptly, causing disorientation. (It is useful to tell the user if you have provided yard definitions outside yards, because most users will not suspect that this option is available outside yards.)

### 3.7.3 SIGNALS

**Signals control train movements. In MSTS they also provide distance information**: when the Track Monitor is used (toggled with F4 in MSTS), it shows the distance to the next signal.

By default, one type of signal is available in a new route (you can import others as objects, see section 3.10).

**To place a signal**, you can select the default Track object "Japan 3 Light Signal (Left)", and position it between the rails of a track section. The signal consists of two parts: a red handle and the signal post. Press F2 and select the signal post, then press F3, point where you want it to be (left or right of the track), press F4 and rotate it to face oncoming traffic (while pressing Ctrl, drag the mouse left or right), then press H to make it rest on the ground.

### 3.7.4 SPEED LIMITS

**Speed limits tell the driver what speeds not to exceed.** If the derail option is turned on in MSTS, a train going faster than the local speed limit will soon derail. The speed limits also show up on the Track Monitor (seen by pressing F4 while driving): these are automatically converted between mph and km/h as needed by each particular train.

[NEW SINCE V1.106] A route has a general speed limit, called default speed limit, which is valid wherever no other speed limit is specified. There are also speed limits that are specified locally by speed limit signs placed along the tracks: these may not exceed the default speed limit. And there is finally a restricted speed limit that is used in Activities that impose a temporary speed limit (typically in construction zones).

[NEW SINCE V1.106] **If you want to change the route's Default speed limit and/or the Restricted speed limit, you can do this within RE after loading the route:**

- select Route | Properties;
- under Units, enable Metric (for km/h) or Imperial (for mph);
- change the Default speed limit, if desired;
- change the Restricted speed limit, if desired;
- click OK, and save the route.

By default, RE offers 3 (Japanese) speed limit signs in a new route, including one speed warning sign and a resume speed sign (the warning speed sign does not work, because of a missing file). The resume speed sign shows a kind of cross (if you try to show a speed or milepost distance on the resume speed sign, it is written at the foot of the post, not on the sign, so this option is unrealistic).

You add a speed limit sign very much like a signal: place it first between the rails; you can then press F6 or right-click to specify the speed limit (in mph or km/h); next place the sign itself off the track, rotate it to face oncoming trains, and press H to rest it on the ground.

### 3.7.5 MILEPOSTS

**Mileposts show the distance along the track, measured from a reference point.** (In reality, they may not show the exact distance, because railways may change track layouts, such as building a shortcut through a tunnel, without adjusting the mileposts along the rest of the route.)

NOTE: Mileposts are often hard to read in MSTS from the driver's perspective: the cab-free view helps (press Shift-1).

RE offers by default one type of milepost. Others can be imported like objects, see section 3.10.

**It is easy to place mileposts, but is not so easy to know where to place them:** That requires knowing the distance to some reference point ("milepost 0"). For relatively straight routes, this can be done by reference to the underlying tile and patch grid in MSTS, given that they form squares of 2048m x 2048m and 128m by 128m, respectively. Another way, more laborious, but more precise and suitable for any route, is to use signals, the Track Monitor, and the "head-up display" (opened by pressing 0 - zero) in MSTS: the Track Monitor tells you the distance to the next signal, while the head-up display tells you the coordinates. You should note those coordinates at each location where you want to place a milepost, then start up RE and place mileposts at those locations.

TIP: The quickest way to note coordinates is to do a screen grab while you drive: press PrintScreen to record all the information shown on the MSTS screen, including a visual reminder of what the place looks like; you will find the various screen grabs as PCX files in the main Train Simulator folder, with names scrngrb*.pcx, and you can open them with most graphics viewers.

**One practical way to do this is the following.** In RE, place a signal at the position where you want "milepost 0" to be. Then, in MSTS, drive along the route toward that signal and watch the Track Monitor as it tells you the distance to it: when the shown distance corresponds to where you want a milepost, record the coordinates (latitude and longitude) shown by the head-up display. Next, in RE, place mileposts at all those locations.

NOTE: the Track Monitor does not "see" signals farther than 20 km (12.4 miles) ahead. For a stretch that is longer than this, you would have to add one or more intermediate signals to measure distances.

[NEW SINCE V1.106] A similar method works for longer distances: if you place a platform name near your "milepost 0", you can create an activity that includes that platform as a stop. Then, when running in MSTS toward that platform, press F10 to open the Timetable window: it will show the distance to that platform. You can use this distance exactly as above with the Track Monitor.

[NEW SINCE V1.106] NOTE: To change the units (miles vs. kilometers) shown for distances, you may either use the RGE (select Edit then Route Values) or edit the route's *.trk file, so that it has a line

```
MilepostUnitsKilometers ( )
```

for kilometers, or a line

```
MilepostUnitsMiles ( )
```

for miles (the line for Miles is absent by default and can be omitted).

[NEW SINCE V1.106] NOTE: Another approach for locating positions for mileposts has been suggested, but does not work correctly: drive at constant speed, time the distance between two mileposts (for example, at 60 km/h you cover 1 km every minute), and note the locations as you pass. The reason that this does not work is that, apparently, the speed reading is not correct in MSTS: it is reported that it takes about 40-50 seconds to cover 1 km at 60 km/h in MSTS, instead of the correct 60 seconds.

**To place a milepost**, select it from the Interactive objects in the Object selector. Then place it right between the rails of a track segment: you will see a multicolored handle and the milepost itself under the handle. Press F6 or right-click, then select Milepost / Speed limit: enter the number that the milepost should display (use "Show dot" if you want to display fractional numbers, like 35.00; otherwise the number will be rounded off to an integer, like 35); press F3, point at where you want the milepost itself placed (about 1-2m from the track), and left-click there; press F4 and rotate the milepost to face the oncoming trains, if needed; then press H to make sure it rests on the ground.

(The speed limit option of the default milepost will not show trackside in MSTS: use the speed limit signs instead.)

## 3.7.6 DEER

Animated deer can be added just like many of the other objects discussed here. However, they are seen in MSTS only when running an Activity in which they have been activated. They will show up in more or less randomized positions near where you placed them. The animation causes movements due the approach of a train or due to the train's horn/bell. An approaching train will make deer jump and run a short distance, and they will jump and run again each time the horn or bell is sounded.

To place a deer in the RE, select "Deer_on_track" in the Hazards class: then place it between the rails along your track; you may move the deer itself off the track, and orient it as you like (it will run away in that direction).

[NEW SINCE V1.106] You can set how far and how fast the deer will run due to sounding the bell or horn: in the file deer.haz (which controls all the deer in the route in the same manner), you can change the Distance (given by default as 10 meters) and the Speed (by default 10 meter/second). I prefer 30 meter and 3 meter/sec.

In the deer.haz file, you will also find the following lines:

```
Idle_Key                ( 59 178 )
Idle_Key2               ( 46 59 )
Surprise_Key_Left       ( 36 46 )
Surprise_Key_Right      ( 25 35 )
Success_Scarper_Key     ( 0 24 )
```

According to the MSTS Tech Docs, these five lines are timed actions: the pairs of numbers on each line may represent the time interval during which that line's action can be triggered. You may be able to change the relative probability of each action by changing these times.

The two Idle_Key lines are idle times that run randomly and keep the deer static for a while (but probably include the head-down bucking motion); only these two actions operate when the train is far away. The two Surprise_Key lines control the reactions to the approach of a train coming from the left or the right, triggering the deer to turn their head. The Success_Scarper_Key line triggers the jumping and running away.

To activate deer in an Activity, move the Animals frequency hazard slider to the right in the Conditions and Hazards box of the Activity Editor. The farther toward 100% you move the slider, the more likely will the deer appear in an MSTS Activity: at 50% only half the deer will show up.

NOTE: You can also add static deer, which don't move or run - see section 3.12.5.

**R. Building "First Route": adding objects, such as telephone poles, forests and bridges**

You will next learn to add other types of objects near your tracks. These will start to fill the emptiness of the bare terrain. But it is best to finish shaping your terrain before you add objects!

TIP: Place objects only where people in a train can see them. Every additional object slows down MSTS, so you should not place objects out of sight!

By default, RE prepares only a few objects for use by a new route, such as telephone poles, a forest, a tree and a bridge segment, as well as the signal and milepost that you already know. We will use these now. Roads are also available, and we will deal with them later. Other objects can be imported from default MSTS routes, but importing is somewhat complicated.

See section 3.10 for more details on importing and placing other objects.

WARNING: **The default telephone poles are dangerous** and can cause both MSTS and RE to crash, as others and I have found! Some people (including me) have found that they can't remove telephone poles, once placed. Twice I tried to install telephone poles in our First Route: the first time, RE crashed when I moved the camera around; the second time, MSTS crashed upon terminating. In another trial route, a few telephone poles had no negative effect. Others have been able to use them without problems, however. One possible reason for problems is that telephone poles do not turn to wire frame when selected: so it is not easy to tell whether you have unselected them before saving your route. So **it is important to unselect telephone poles before saving** (press F2 and left-click on bare terrain or in the sky).

If you want to try adding telephone poles, be sure to **first back up** the complete FirstRoute folder in a safe place. Then, after placing telephone poles, thoroughly test the route. If it fails, delete the newest version of the FirstRoute folder, and copy the backup into its place.

NOTE: For safety, I have not placed telephone poles in the version of First Route that accompanies this Guide.

**AT YOUR OWN RISK:** Place **telephone poles** along the tracks that leave our Central London station. By default the telepole.s object contains three poles separated by 10m: that is much too short! To change the pole separation to a more reasonable 30m, do this:

- exit the Route Editor;
- look for a file called telepole.s in the main folder of your First Route;
- edit it (with WordPad) so that one line reads
```
Separation ( 30 )
```
- restart Route Editor.

You need not make that change to place telephone poles, but then you will have to place many more of them.

**Do this as follows:**

- move to the first switch of your track;
- select telepole.s in the Track objects class;
- press F5;
- place a trio of poles a few meters to one side of the track;
- you can rotate the trio (around the central pole) by pressing F4 and dragging the mouse left or right;
- you can move the trio by pressing F3 and dragging the mouse in any direction;  (the poles stay correctly on the ground, even on sloping terrain);
- repeat the last three steps for the next trio, etc.;
- **unselect your last telephone poles!**

[NEW SINCE V1.106] NOTE:  You cannot copy a trio of telephone poles.  This means that each new placed trio has to be placed individually, and must be aligned and oriented separately.

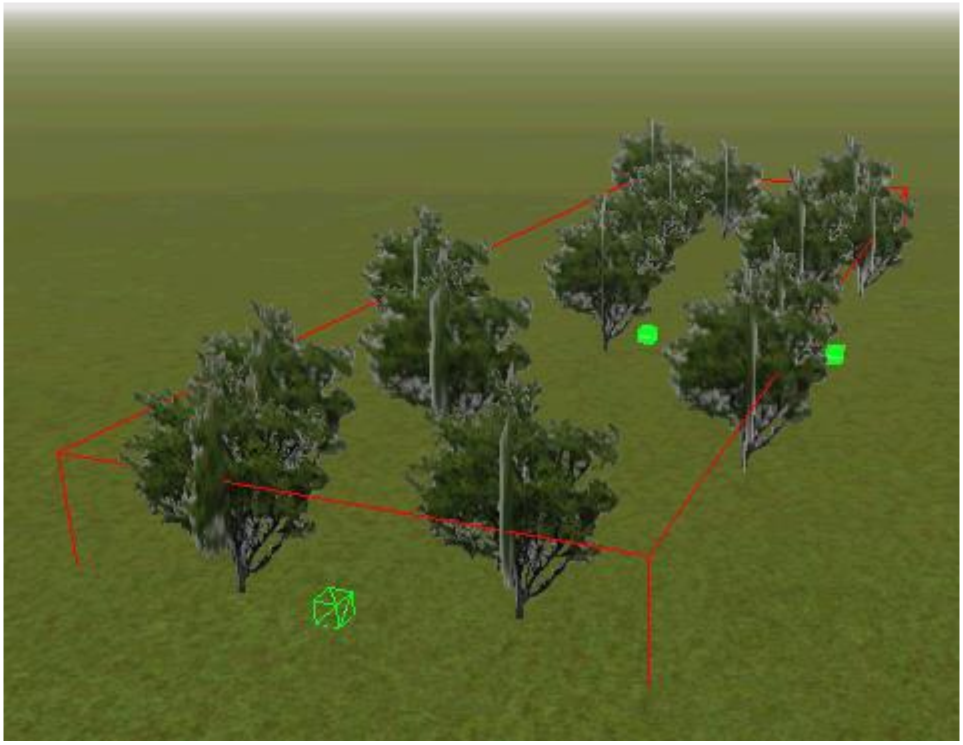
[NEW SINCE V1.106] NOTE:  You can apparently lengthen an existing line of telephone poles by editing the world file of its tile;  this will multiply the number of poles, using the previously set separation between them.  This can be done as follows (warning:  this process is complicated and can easily lead to crashes of RE!).  In RE, identify the tile and coordinates on the tile where those telephone poles are (see section 3.10.9 for instructions to do this);  also determine the desired start and end coordinates of the line of telephone poles (these are tile coordinates, ranging from -1024 to +1024, not latitude and longitude);  close RE, then open the corresponding world tile and look for the corresponding item `Telepole ( ... )` with the correct `Position ( ... )`;  now edit the coordinates in the entries `StartPosition (... )` and `EndPosition (... )`;  save the world file and restart RE to check the new positions.  Caution:  do not extend a line of telephone poles across tile boundaries (the blue lines showing after you press F7 in RE).

**Save your work, after deselecting any selected track or objects!**

Next **place forests** along your tracks.  A forest is defined by a rectangular box, for which you can choose the edge lengths, the orientation and the tree density.  This is a convenient way to add a lot of trees to your terrain.  You should place forest objects primarily along the tracks for the people on-board to see.  **Do this as follows:**

- select "Forest - JP1tree1" in the Track objects class;
- press F5;
- place the center of the forest where you want:  a red box appears, which probably contains only one tree;  it should also have 3 green "handles", one in the center (shown as wire-frame now), and two toward the edges (if you don't see them, they may be hidden by the tree, or they may be underground, so turn the camera around the forest or press W);
- right-click:  a pop-up menu opens;
- select Forest and enter a Density between about 500 and 5000 trees per square km;  press OK: more trees will probably show up;
- press F2;
- select one of the two off-center green handles:  it turns to wire frame;

- press F3;
- drag the selected green handle to stretch and/or rotate the red box:  see the next figure;



- press F2 and select the other off-center handle, then press F3 and adjust the width of the forest;
- press F2 and select the center handle, then press F3 and move the forest as needed;
- press H so the red box and green handles adjust their height to the local terrain;
- repeat these steps (starting with pressing F5) to place and adjust another forest object.

**Save your work, after deselecting any selected track or objects!**

For more variety in types of trees, you may also **place small individual trees**:  there is one type of single tree available by default.  You may place them anywhere, including inside a forest;  put them especially in gaps between forests or in gaps within a forest;  put them closer to your tracks rather than in the back of forests, so they are more obvious.  **Do this as follows:**

- select "Tree_1" in the Vegetation class;
- press F5;
- place a tree;  repeat as many times as you wish.

NOTE: A nice feature of the trees in forests and the single trees is that they stick to the ground: the trees follow terrain up and down when you change its altitude and shape.

**Save your work, after deselecting any selected track or objects!**

Next we will **place a bridge**:  it is an object that we will place over a river.  By default, one type of bridge section is available (it is really meant to carry a road, but can carry a track instead):  it is composed of one pillar and one horizontal segment leading out to one side of the pillar.  By lining up several such bridge sections, you can assemble a longer bridge.

Placing objects can be painstaking and exacting, due to the many degrees of freedom that they have (up to 6!):  you need to adjust every degree of freedom.  The following procedure works relatively fast for horizontal bridges.  (For inclined bridges, see section 3.4.1.)

Build a bridge in First Route under the short stretch of single track between the siding and the high-speed dual-track line (about halfway between the two hills).  We will make a river flow under it.  More precisely, we will lay one section of bridge starting exactly at the end of the switch that terminates the siding;  then we will add three more such sections to make a longer bridge.

**To place a bridge, do this:**

- move to the place where the bridge will be placed;  do not use wire-frame mode for the terrain;
- select JP2bluebrg in TrackObjects;
- press F5;
- left-click near the track where you want the first bridge section to start (near the red pole marking the siding switch):  you get a wire-frame pillar standing on the ground, with a horizontal segment pointing due North;
- press F6 or right-click to open a pop-up menu:  select "General (cont.)" and put a check mark before "Terrain object";  if you don't, the track ties and ballast will not be visible after the bridge is placed;  (you could at the same time also change the Shadow type temporarily to "None", so as to remove the gray shadow that you see in the figure below);
- inspect this bridge section to understand its shape and size (deselect it to see it better);  we will of course line it up with the track, but we will also push it down so its track bed fits right under the track;  that buries the pillar, but it will reappear when we depress terrain to form a river;
- now choose a vertical bird's eye view, so the camera looks straight down on the bridge (you know that you look vertically down when the compass jumps by 180°!);  turn the camera so the track runs up and down the screen, as shown in the left part of the next figure;
- select the bridge section again (press F2, point at it so it turns red, and left-click);
- press F4;
- turn the bridge to be parallel to the track:  first move roughly (using the keypad's left- and right-arrows), then fine-tune (dragging the mouse left and right while pressing Ctrl);  fine-tuning the orientation is quite important so that other bridge sections fit the first one correctly;  the bridge may be off the track centerline for the time being, as shown in the second part of the figure below;
- push the bridge down to just under track level:  first push it down fast (using the keypad's down-and up-arrow keys while pressing Shift), then slowly (using just the down- and up-arrow keys):  when it reaches track level, its wires that run across the track will become invisible under the ties and ballast, while the "handrails" on both edges of the bridge remain visible above ground, as shown in the third part of the figure below;

- move the bridge, so its pillar is near the switch's red pole (you may zoom in with the camera): press F3, keep pressing Ctrl (to freeze the bridge's altitude), and tentatively left-click a bit further along the track;  to position better, left-click again, while keeping Ctrl pressed:  as you try to place the bridge in the correct location, you can try to also center it on the track centerline;
- fine-tune the horizontal position of the bridge:  keep Ctrl pressed, and use the left-, right-, up- and down-arrow keys to move the bridge to its final position;
- press F2, and deselect the bridge to see it better.

**Save your work, after deselecting any selected track or objects!**

Now check your placement, by moving the camera down low:  it should look like the right part of the figure below.

To place the next bridge segments, we will copy and paste the existing one, to save ourselves the task of orienting each one again.  Do this as follows:

- take the bird's eye view again;
- press F2;
- select the existing bridge section;
- press Ctrl-C;
- press F5;
- point roughly where the new bridge section will go;
- press Ctrl-V:  this places a copy of the bridge segment (with the same orientation) above ground;
- if you don't like the copy's position (it need not be precise yet):  press Delete, point at a better position, and press Ctrl-V again;

- push the track to just under the track level, as for the first bridge section;
- fine-tune its horizontal position, close against the end of the first bridge section;
- check that its "y" altitude (shown in the Terrain window) is the same as that of the first bridge section:  adjust as necessary.

**Save your work, after deselecting any selected track or objects!**

Now you have two sections of the bridge in place.  How to extend it further?

TIP:  Doubling and redoubling.  To place two more bridge sections, copy the existing pair, copy and paste that pair, and adjust the position of that pair!  For a longer bridge, you could double again:  copy a quartet of bridge sections, copy, paste and adjust the quartet.  Then double again, etc.

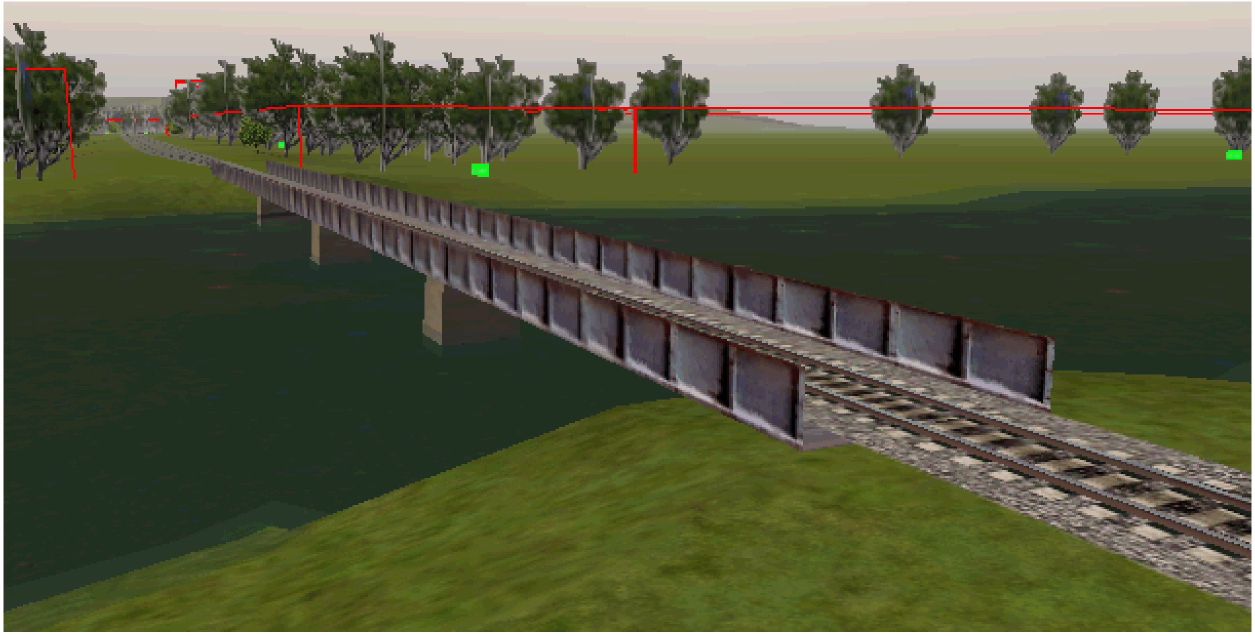
For doubling and redoubling, you just need to know how to select more than one object:

- press F2;
- point at one object and left-click;
- while pressing Ctrl, point at another object and left-click (repeat this to select more objects).

With four bridge sections, we have enough to reach to the next switch, so we stop there.

A final check:  make sure the track ties and ballast show up on the bridge;  if they don't, select each bridge section in turn, press F6 or right-click to open the pop-up menu, select "General (cont.)" and put a check mark before "Terrain object".  At the same time, you can reset the Shadow type to "Rectangular" or "Dynamic" (see section 3.10.8 for details about the shadow types).

Your bridge is complete.  Now you can make a river, in exactly the same way you made one earlier (toggle water on in the patches where you will put your river, and depress terrain to form the riverbed).  The result might look like this in RE, including forests:



**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will add roads, road traffic and level crossings.

**To continue building our First Route, jump to the next blue box.**

## S. Building "First Route": adding roads, road traffic and level crossings

You will next learn to add roads, road traffic and level crossings to your First Route. These will add a lot of life to your route. See section 3.8 for more details.

WARNING: It is best to add roads only after all tracks have been laid.

Roads are handled in RE in a way quite similar to tracks: you place a first road section where you want, and then add other road sections to the ends of the first one. To minimize roads, don't extend them out of sight. You can start new roads that are unconnected to older roads.

By default, a new route has a good choice of road sections. Let's **place a two-lane road** just North of our Central London station: we make it cross the tracks, in preparation for placing a level crossing there later.

**Do this as follows:**

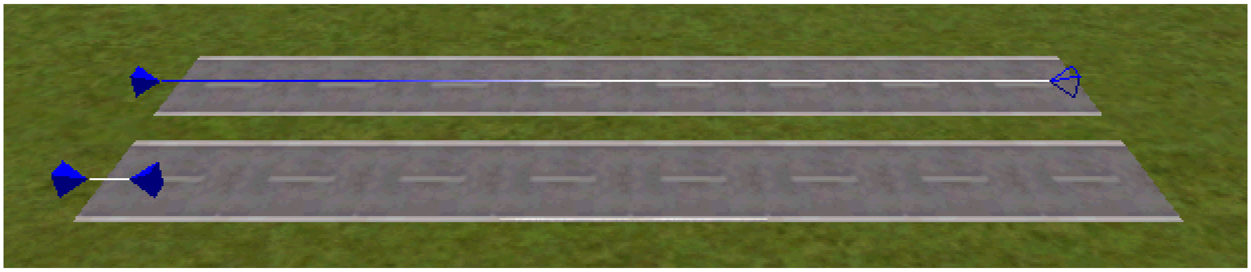
- move a hundred meters or so beyond the first switch of your track;
- select Road2Lstrt100m.s in the Road objects class;
- press F5;
- place one end of the road section about 50m to one side of the track;
- press F4;
- rotate the road section so it crosses the track at about right angles (drag the mouse left or right);
- press N to make the road section lie flat on the ground: it should pass underneath the tracks, and be hidden by the ballast.

Now you can build a longer stretch of road in both directions, with the same or other road sections (you can make right curves by pressing T: also press T to make the sections connect properly!). Don't go far.

**Save your work, after deselecting any selected track or objects!**

Next, **add moving traffic to this road**. For that, we use the **car spawner**, which creates cars at random intervals. **Do the following:**

- select the "Car Spawner" in the Interactive Objects class;
- press F5;
- place a car spawner near the end of one lane of your road: you will see a pair of blue handles appear (as at bottom left in the next figure);
- press F6 or right-click: choose the average time interval between cars, and the average speed of cars.



NOTE: If, when placing a car spawner, you miss pointing correctly at the road, try again. You will have to reselect the car spawner from the Interactive Objects class.

The idea is to move one of the two blue handles to the other end of that same lane: that will stretch a line from one end of your road to the other (as shown in the upper part of the figure above). This line defines a stream of traffic. As you will see, the traffic is represented in RE by a blue/white wave traveling along that line. However, for some reason, the wave travels in the direction opposite the traffic direction. Since our route is in the UK, road traffic drives on the left, and we will arrange for that. Do the following:

- press F2;
- select a blue handle;
- press F3;

- left-click about 20m further along that lane (or drag the handle): you see the blue/white wave traveling along a line linking the two handles;
- now you need to think: if this wave travels in the direction opposite to UK road traffic, you have chosen the correct blue handle!
- otherwise, bring the displaced handle back where it was originally by left-clicking near where it was (or dragging it back); then select the other handle and move it in the same way: check that the wave travels the other way now;
- press F3;
- move the camera to the other end of the road;
- left-click on that far end of the same lane: the line with the blue/white wave should trace the road perfectly (if you clicked the wrong lane, the line turns red); if this doesn't work, you should "guide" the blue-white line bit by bit along the road;
- repeat this process for the other road lane, starting from either end: make sure you end up with traffic traveling in opposite directions.

**Save your work, after deselecting any selected track or objects!**

If you run MSTS at this point (for which you need an activity!), you should see cars drive on the correct (British) side of the road. Only one type of vehicle (red cars) is available by default in a new route (see section 3.8 to add other types of vehicles).

There are two problems: these cars drive over your track's ballast and ties, and don't stop for your train. We thus need a special road piece for the level crossing, and crossing gates that close when your train approaches.

No level crossing road piece is available by default in a new route, so we will have to **import a level crossing road piece and its associated files**. At the same time we will **import a sound that the default level crossing gate needs**. Do the following, using Windows Explorer or My Computer (don't worry about lower- vs. upper-case letters):

- copy crossing.sms from the JAPAN1\Sound folder to the FirstRoute\Sound folder;
- copy US2Crossx1Nbar.s from USA2\Shapes to FirstRoute\Shapes;
- copy US2Crossx1Nbar.sd from USA2\Shapes to FirstRoute\Shapes;
- copy crossing.ace from USA2\Textures to FirstRoute\Textures;
- copy crossing.ace (same name, but different contents!) from USA2\Textures\Snow to FirstRoute\Textures\Snow.

(Notice that several files are needed for one object, 4 in this case: see section 3.10 for more details.)

Also, using a text editor like WordPad, **insert the following 6 lines at the end of the file "FirstRoute.ref"** (found in the First Route's main folder):

```
Static (
	FileName ( US2Crossx1NBar.s )
	Class ( "Track Objects" )
	Align ( None )
```

```
      Description ( US2Crossx1NBar )
)
```

These 6 lines can also be copied directly from the file marias.ref in the USA2 main folder. Make sure you include that last parenthesis on the 6th line!

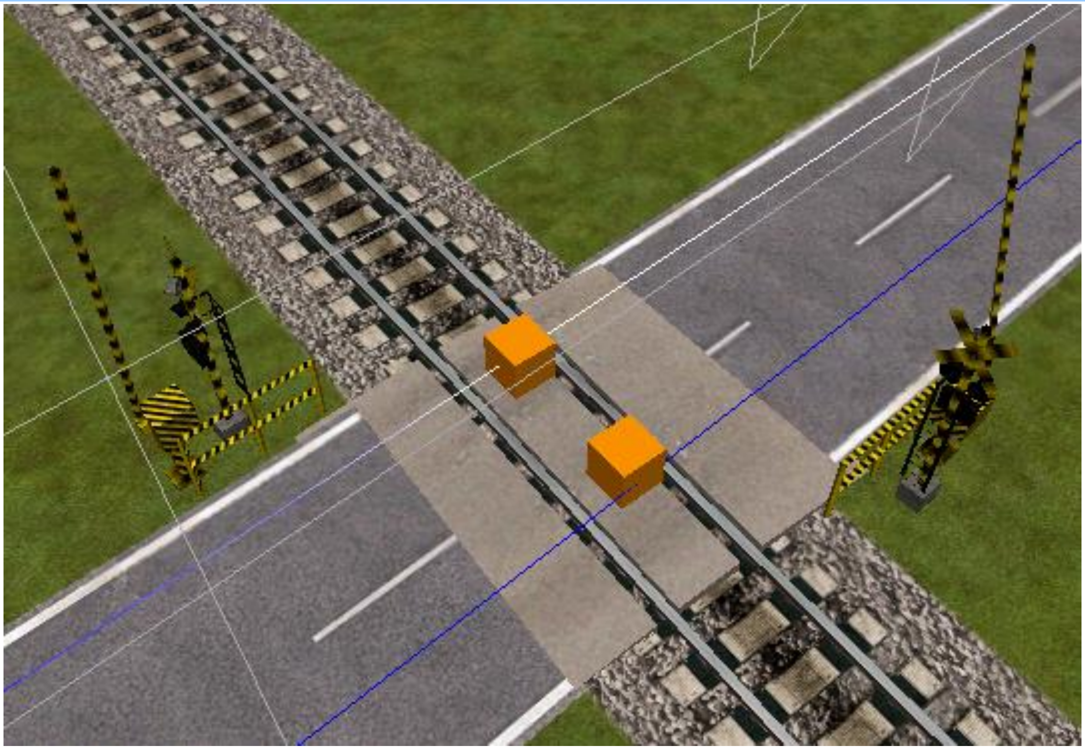
Now **place a level crossing road piece**, as follows:

- select US2Crossx1Nbar in the Track Objects;
- press F5;
- point at the middle of the level crossing, and left-click: this road piece becomes visible, but oriented North-South;
- much like the case of the bridge section (but without vertical motion here), rotate and move this road piece until the rails fit perfectly in its grooves; the result should look something like the figure below (without the orange cubes and crossing gates yet).

**Save your work, after deselecting any selected track or objects!**

Now **add the crossing gates**: there will be one on each side. Do the following:

- select the "Level Crossing (JP)" in the Level Crossing class;
- press F5;
- place it exactly at the intersection of the track and one lane of the dual-lane road (between the rails and over the lane): an orange cube shows up, together with a gate;
- press F3;
- move the gate off the track and off the road on the side of oncoming traffic (see the figure below for left-side British traffic);
- press F4;
- rotate the gate to face oncoming traffic (see figure);
- press H, to make the gate rest on the ground;
- repeat this for the other gate.



Now, when you run MSTS, the gates will close as your train approaches, and the cars will stop until the gates open again after your train has left. This is shown in the next picture, after more vehicles have been added to the car spawner (see section 3.8), in my final version of First Route.


**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will import textures to change the uniform green landscape color to something more varied and realistic.

**To continue building our First Route, jump to the next blue box.**

## 3.8 Roads, road traffic and level crossings

WARNING: It is best to add roads only after all tracks have been laid. Severe problems with the track database have been observed by a number of people (including me) who placed tracks after placing roads.

### 3.8.1 LAYING ROADS

**Roads are handled in RE in a way quite similar to tracks**: you place a first road section where you want, and then connect other road sections to the ends of existing ones. You can start new unconnected roads elsewhere.

A complete list of the available road sections is given in Appendix C.

There are a variety of straight and curved road sections, as well as T-junctions and intersections (crossings). However, there is no dynamic road section, and there are no tunnel versions. Road sections can be made to slope up or down within the limits of +20º and -20º (or more, if you use essentially the same trick that works for track sections to go beyond +3º and -3º). You can create any number of unconnected roads.

The more roads you build and the longer they are, the slower the frame rate of MSTS will be (as with any added objects). It is recommended to make roads short: make them disappear and stop out of sight behind hills, trees, buildings, etc., especially if you add road traffic, so vehicles do not pop up and evaporate unnaturally.

TIP: Show the frame rate in RE by pressing Shift-Z.

### 3.8.2 ADDING ROAD TRAFFIC

You can **add moving vehicles on any road by placing one or more "car spawners" along the road**. A car spawner creates cars and other vehicles at random intervals and with random speeds, and makes them travel along one road lane between points that you specify (vehicles may not turn corners at T-junctions or intersections). You can control the average interval and the average speed for each car spawner.

You place a car spawner as follows (also see the illustration in section S):

- select the "Car Spawner" in the Interactive Objects class;
- place a car spawner on one lane of an existing road (typically near its end): a pair of blue handles appears;
- press F6 or right-click: choose the average time interval between cars, and the average speed of cars.

NOTE: If, when placing a car spawner, you miss pointing correctly at the road, try again. You will have to reselect the car spawner from the Interactive Objects class.

You must now move one of the two blue handles further along the same lane (typically to the other end of your road): that stretches a line along your road. This line defines a stream of traffic, which is represented by a blue/white wave traveling along that line. However, the wave travels in the direction opposite the traffic direction. Do this as follows:

- select one of the two blue handles;
- left-click about 20m further along that lane (or drag the handle) to see the blue/white wave;
- if this wave travels in the direction opposite the direction you want for the road traffic, you have chosen the correct blue handle;
- otherwise, bring the displaced handle back where it was by left-clicking near where it was originally (or by dragging it back); then select the other handle and move it in the same way;
- left-click at the other end of the same lane, or further down the road on the same lane, to make the line with the blue/white wave trace the road (if you clicked the wrong lane, the line turns red); in this way you "guide" the blue-white line along the road as far as you want.

By default, in a new route, the car spawner will only produce one kind of vehicle: red cars. To vary this, you need to do two things: import other vehicles from default MSTS routes, and change the file carspawner.dat (in your route's main folder) to list those additional vehicles. You import vehicles the same way you import objects. See section 3.12.3 for details.

### 3.8.3 ADDING LEVEL CROSSINGS

**Level crossings occur where a road crosses a track.** Two kinds of level crossing are available: those built into tracks, and those that are added to normal track.

By default, a new route has access to two track sections which have a road section built into a pair of 10m long dual tracks (A2t10mLvlCrs.s with gate and A2t10mLvlCrsNoGate.s without gate). You can add normal road sections under these crossings, as well as road traffic. However, the gate does not seem to respond to the presence of a train. To either of these two level crossings, you may add separate gates and signals (see below).

The second approach is to first lay track and then add a road that crosses the track: the road should pass under the track, becoming invisible under the ballast. Two more things are needed here:  one or more pieces of road to cover up the track's ties and ballast, and one or more sets of gates and/or signals.

In principle, a road can cross a track at any angle, but the road pieces that you need to add will in practice limit the range of angles.

By default, no road pieces are available in a new route for level crossings:  you need to import them from default MSTS routes (see section 3.10).

By default, one type of level crossing gate with signals is available in a new route:  "Level Crossing (JP)".  It consists of a swinging arm and signals for vehicles.  You need one of these on each side of the track(s).  This gate is meant for left-handed Japanese traffic;  it will work on right-handed roads, but will look a little bit odd to the sharp observer.  See also the illustrations in section S.

To place the "Level Crossing (JP)", you point and left-click between the rails of a track at the intersection with one road lane:  this places an orange cube and the gate itself.  After pressing F3, you can position the gate off the road and off the track, and after pressing F4, you can rotate it to face oncoming road traffic (also press H to rest it on the ground).

A crossing gate will stop spawned vehicles:  you can control timings, etc. by pressing F6 or right-clicking when the crossing gate is selected.

NOTE:  Some level crossings have bell sounds.  You may not hear them in RE, but you should hear them when running in MSTS with an outside view.

WARNING:  Use of this crossing gate requires copying the file crossing.sms from the JAPAN1\Sound folder to your route's Sound folder.

WARNING:  The gate of the level crossing UKCrossGate.s (called "Level Crossing" in the EUROPE1 default route) is open when it should be closed, and vice versa.  A corrected level crossing is available in the Train-Sim.com library as file ukcross.zip.

[NEW SINCE V2]
### 3.8.4 COVERING THE TRACKBED WITH ROADS

By default, the MSTS tracks always show the underlying trackbed, including ties.  At level crossings and when tracks run along streets, the asphalt usually makes the trackbed and the ties invisible.

You can **make the trackbed and ties invisible** by simply raising road sections by a few centimeters until only the top of the rails remains visible.  This approach works best for horizontal tracks and roads.  This is illustrated in the picture of section 3.5.5:  at lower left a section of dirt road has been placed just below the top of the rails;  it could have been raised higher to eliminate the dark side of the rails.  (At bottom right another section has been placed below the road bed.)

## T. Building "First Route": changing terrain textures

You will next learn to change the default green terrain texture, and also to vary the terrain texture from place to place. This will make the landscape much more realistic. See section 3.9 for more details.

First you will change the basic texture that is used everywhere by default in First Route. Then you will add other textures that allow local variations along the route.

It will be necessary to import terrain textures from a default MSTS route. Since the First Route is located in the UK, we choose textures from the Settle-Carlisle route. I made a choice by opening that route in RE and looking at the textures used there (we will only use a fraction of its textures).

**To import textures, do this:**

- exit from RE;
- copy the file UKHedge1.ace from the EUROPE1\TERRTEX folder to the FirstRoute\TERRTEX folder;
- copy the file UKHedge1.ace (same name, different contents!) from the EUROPE1\TERRTEX\SNOW folder to the FirstRoute\TERRTEX\SNOW folder;
- in the FirstRoute\TERRTEX folder, delete terrain.ace, then make another copy of UKHedge1.ace and rename it to terrain.ace;
- do exactly the same in the folder FirstRoute\TERRTEX\SNOW;
- copy the files OEDirt.ace, OEHalfScrub-Dirt.ace, OEQuartScrub-Dirt.ace, OEQuartDirt-Scrub.ace, UKBScrub.ace, UKHalfBScrub-GGrass.ace, UKQuartBScrub-GGrass.ace, UKQuartGGrass-BScrub.ace, UKGGrass.ace, UKHalfGGrass-Hedge.ace, UKQuartGGrass-Hedge.ace, and UKQuartHedge-GGrass.ace from the EUROPE1\TERRTEX folder to the FirstRoute\TERRTEX folder;
- copy the same files (different contents!) from the EUROPE1\TERRTEX\SNOW folder to the FirstRoute\TERRTEX\SNOW folder.

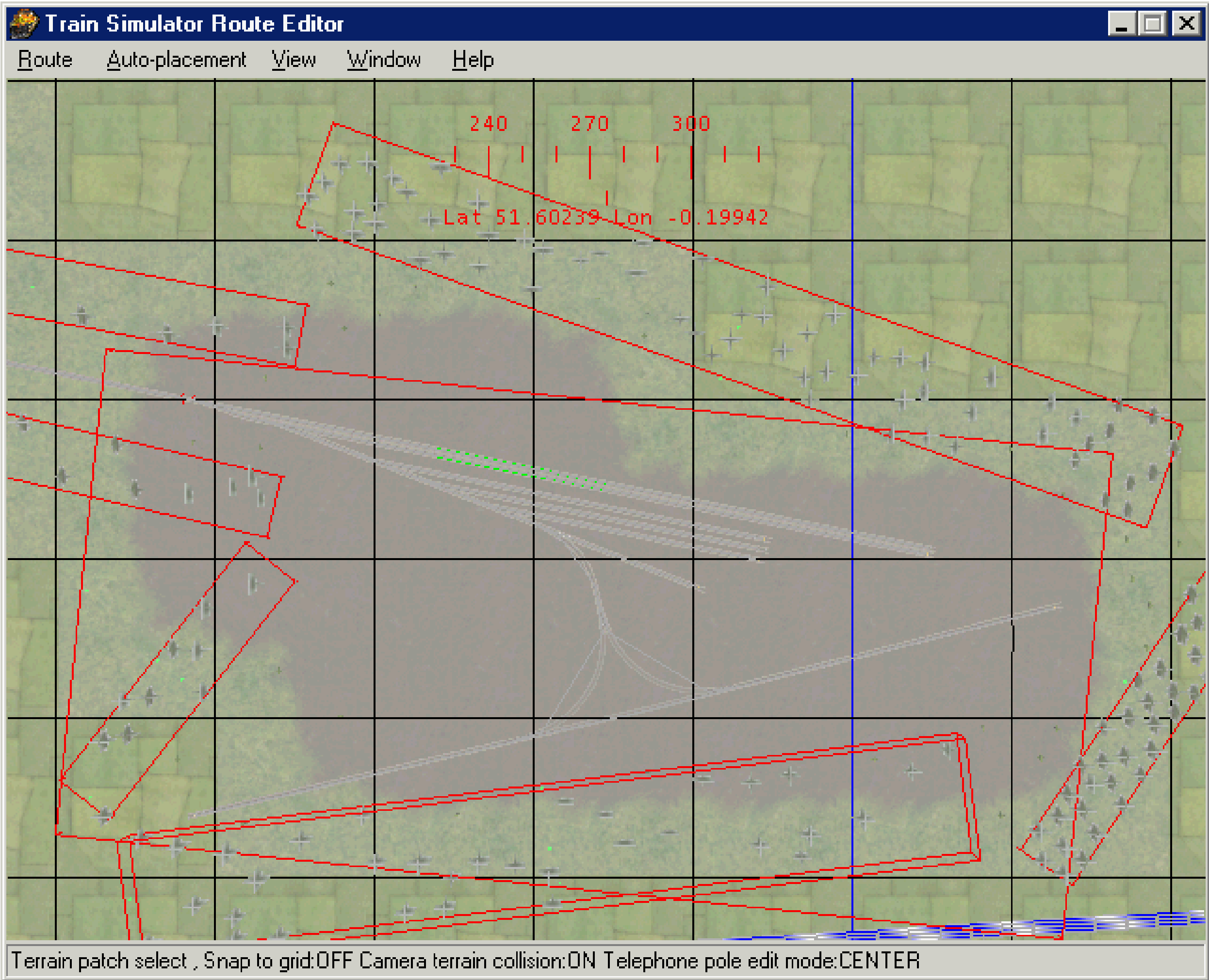
Now open your **First Route** in RE: it **has a new basic texture everywhere**, due to the change of the terrain.ace files. This texture should show green fields with hedges, a typical UK landscape.

Next you will **add different textures to different locations**. First, add "dirt" (in OEDirt.ace) in and around the North London station to obtain a result like that shown in the figure below (the exact location of your tracks with respect to the patch boundaries is important here, and will depend on your particular track layout; note that this view from a high altitude is very hazy!). Pay close attention to the relationship between patch boundaries and textures! Do this as follows:

- select through the RE menus:  Window - Texture select;  this opens the Texture selector window, which shows the new default texture numbered as 1;  the Texture selector window is illustrated below (in a later state after a dozen textures have been read in)
- using the Texture selector menu, select Edit - Insert:  this opens a box of *.ace files, including mainly those that you just imported;
- select file OEDirt.ace, and click Open:  this texture (dark gray soil) is now shown numbered 2;
- raise the camera high up so you have a bird's eye view (not necessarily vertical);
- press F7:  the tile and grid lines show up;
- left-click on a patch in the middle of the station:  its outline turns red;
- press the number 2:  that patch turns dark gray:  it has received the desired texture;
- repeat this for the other patches that you want to turn completely into dirt.


Now select and place the textures that will serve as transitions to the greener scrub areas outside the station:

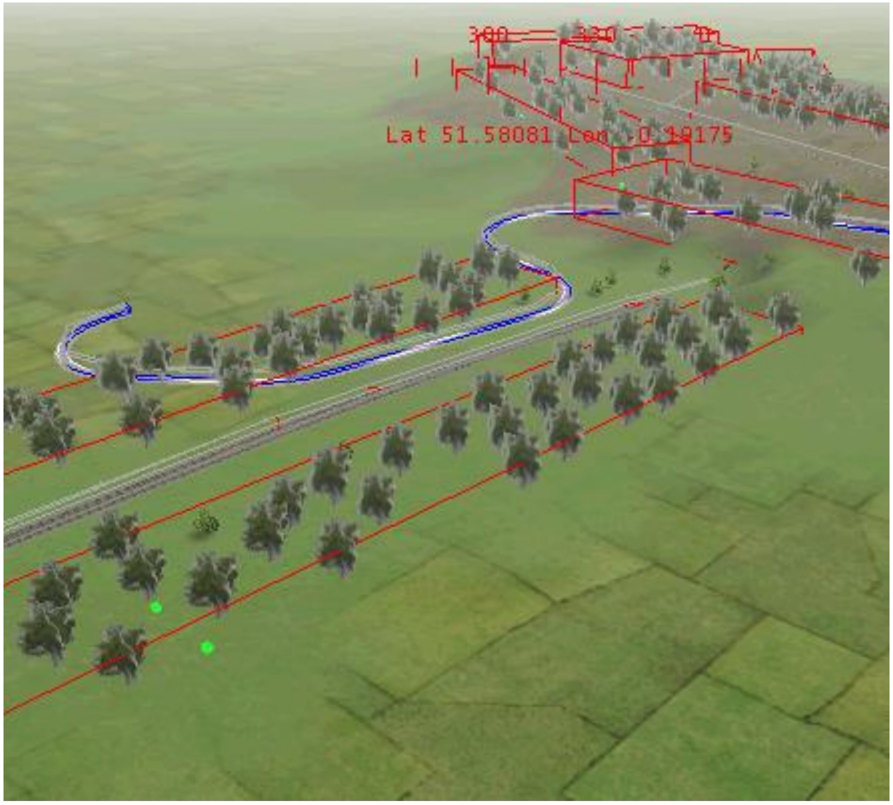
- through Edit - Insert, select OEHalfScrub-Dirt.ace;  then select OEQuartScrub-Dirt.ace;  then select OEQuartDirt-Scrub.ace;  these 3 new textures show up in the Texture selector window, numbered 3 through 5;
- click on a patch that you want to be half dirt and half scrub, at the edge of the station area; press the number (3), since it corresponds to that "combination texture";
- probably the new texture in that patch now has the wrong orientation:  it may need to be turned by 90º, 180º or 270º;  this is done by right-clicking and choosing one of the Rotate options in the pop-up menu;
- repeat the last two steps for all the "half-half" texture combinations around the station;
- in a similar fashion, now do the inside and outside corners, where you need either 3/4 dirt and 1/4 scrub, or 1/4 dirt or 3/4 scrub.




**Save your work, after deselecting any selected track or objects!**

The edges of the figure above show the default hedge texture that First Route now has. The sudden transition from the green scrub texture to the hedge texture is unnatural but minor, and generally out of sight from the tracks: you could make it gradual by importing other texture files from the EUROPE1 route; many texture combinations may be needed to achieve such gradual transitions in general! To achieve that you might copy all terrain textures from the EUROPE1 route (at the cost of more hard disk usage).

The figure below shows one way to treat the main tracks and the hill on which the loop runs: I have arranged for the main tracks to run through green grass (GGrass.ace), while the hill is covered with brown scrub (BScrub.ace) surrounded with green grass. All the needed transitions (half/half, quarter/three-quarter and three-quarter/quarter) are available because you imported them. (Notice also my dirt road that runs up the hill, with low-density slow-moving traffic: this has nothing to do with textures!)




**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will place a few station platforms and fix up the tunnel entrances: then the First Route will be nearly finished.

**To continue building our First Route, jump to the next blue box.**

## 3.9 Importing and using terrain textures and transfers

**Textures give color and a realistic appearance to terrain and objects.** Each of the 16x16 patches in each tile can be given a different "terrain texture", as illustrated in section T. **Transfers** (also called texture stamps) **are texture pieces that you can lay over the terrain texture to create more variation, or to cover up holes in the terrain**: the transfers have variable size and orientation for great flexibility. Of course, the more textures and transfers you apply, the more disk space will be needed, and the slower the frame rate will become in MSTS.

[NEW SINCE V2] Terrain textures and transfers don't have a 3D shape of their own: they simply take the shape of the terrain on which they are laid, like paint. One important difference between terrain textures and transfers is that transfers usually have transparent edges, so that they can blend into the underlying terrain textures without showing sharp boundaries, while terrain textures are totally opaque, with sharp edges. Another difference is that transfers need to be declared as transfers in the route's *.ref file, so they can be placed in the route like an object.

One other class of textures gives color to the various objects in MSTS (such as houses, bridges, trees, etc.): we will not deal with these here.

[NEW SINCE V2] One very important use of transfers is for making tunnel entrances, because these require making holes in the terrain, which must be covered up: see section 3.11. The next picture shows another use of transfers: for covering terrain to simulate rocks (around the upper viaduct), vegetation (mixed with the rocks), trails (the narrow brown line below the viaduct), mountain streams and waterfalls (the steep white lines at top left).

The next picture shows in more detail how transfers can be used to create trails and streams. At right is a 2-tire track that turns into a vehicle-wide trail and then (past the stream) into a footpath. The waterfall at top left becomes a more gentle stream under the bridge and speeds up again to bottom right.

With transfers, you can also model rivers, rapids, gardens, parking lots, fields, and almost everything else that can be rendered two-dimensionally: use your imagination with existing transfers or design new transfers for a wide variety of visual effects. Section 3.9.6 explains briefly how to make transfers.

Unfortunately, by default, when you build a new route with RGE, only one terrain texture is provided: that is the green soil that you see everywhere. And not a single transfer is provided.

However, many textures and transfers are available in the MSTS default routes. To use textures and transfers will mean importing them from the default routes.

To import textures and transfers requires several steps (similar to importing objects), described in detail below: first you need to find which textures and transfers you want to use; then you need to copy them to your route's folder; next, in the case of transfers, you need to change a file to declare them, so they can be used in your new route.

### 3.9.1 IMPORTING TERRAIN TEXTURES

Texture files are identified by their names of type *.ace. If you look in the Terrtex (terrain texture) subfolder of your route, you will see there a file called terrain.ace: by default, this contains the green texture that colors your terrain. If you look in the Terrtex\Snow subfolder, you will also see a file called terrain.ace: it contains the snow-covered version of the green texture, used by MSTS for the winter season.

You can change the default texture alone, so your entire route will have a different texture everywhere.

Or you can add one or more other terrain textures to vary the appearance of the route from place to place.

NOTE: It appears that there is a limit of 64 textures that can be handled within an area of about 2x2 km (about one tile).

WARNING: **If you distribute your route to other users, you must tell them to also import the necessary terrain textures from one or more other default routes.**

WARNING: **It is reported that changing the terrain texture in a tile patch also resets the error bias to its default value of 1. If this happens to you, set the error bias to your desired value again.**

### 3.9.1a Changing the default terrain texture

**It is relatively simple to change the texture of your entire route, uniformly.** This will replace the green default texture by another texture everywhere in your route. To do this:

- identify a terrain texture that you like in a default route: follow the procedure given in section 3.9.1c to find it;

- copy that terrain texture's file to your route's Terrtex subfolder;

- delete your old terrain.ace file;

- rename the imported file to terrain.ace;

- copy the terrain texture of the same name in the Snow subfolder of the default route;

- copy that terrain texture's file to your route's Terrtex\Snow subfolder;

- delete your old terrain.ace file;

- rename the imported file to terrain.ace.

### 3.9.1b Importing ALL terrain textures from default routes

**One relatively easy way to get several terrain textures is to copy all terrain textures from one or more default routes into your route's folder.** To do this:

- copy the entire contents of the Terrtex subfolder of one or more default routes into the Terrtex subfolder of your route (this includes the Snow subfolder within Terrtex).

The main advantage of this approach is that you don't need to first find which terrain textures you want to copy.

The main disadvantage of this approach is that you may not use all those textures, so you will be wasting some disk space.

### 3.9.1c Importing SELECTED terrain textures from default routes

To minimize the number of terrain textures you import from default routes (and thus to save disk space), you must **first identify which terrain textures you want**. To do this:

- load a default route into RE;

- click on Window - Texture select to open the Texture selector;

- in the Texture selector window, click Edit - Insert to open a subfolder of *.ace files;

- click on a *.ace file and see the texture appearing to the right: what you see there is designed to fill one patch of terrain;

- view the textures and decide which ones you want to use: I suggest you copy their names (from the "File name" box) to a WordPad or similar file, so you can later find them again (if you want to use these textures as transfers also, keep that WordPad file in a safe place for later!);

- you can repeat this for other default routes, noting from which route your chosen textures will come.

Now that you know which textures you will use (and now that you have made a list of their file names), you can **copy these terrain textures to your route**:

- copy the desired *.ace files from the Terrtex subfolder of the default route(s) into the Terrtex subfolder of your route.

NOTE: Many terrain textures come in sets. For example, many textures combine two textures, like forest and grassland: in such a case, you would need both the "pure" forest and the "pure" grass textures, but also the half/half, quarter/three-quarter and three-quarter/quarter combinations: these mixed combinations allow you to change over from one texture to the other without sharp unnatural boundaries.

WARNING: **Many terrain textures have more than one version**, such as both a "normal" version and a "snow" version (these are stored in the Snow subfolder of the Terrtex subfolder). If you import a texture, **you should import both versions**. Otherwise, when you switch to a situation in which MSTS requires the "snow" version, but does not find it, it will crash.

[NEW SINCE V2]

### 3.9.1d Showing deep snow

**The default snow terrain textures show mostly light snow on the ground**, such that grass, dirt, etc., are still partly visible through the thin snow cover. To give the **appearance of deeper snow**, you need a texture that hides the vegetation and ground under the snow. Note that this will not actually raise the surface level, so it will not cover tracks, roads, etc.

MSTS offers one such "deep-snow" terrain texture:  US2TarGnd.ace in the USA2\TERRTEX\SNOW folder.  It is not ideal, as it has a streaked appearance.  To use it in your route, you should copy it over all the texture files present in your route's TERRTEX\SNOW folder, without changing their file names.  That may be a bit tedious if you have many textures there!  Mike Simpson's Route-Riter freeware (available at Train-Sim.com) automates this process for you, using its "Improve snow" option.

Ruben Geerling provides both a **better deep-snow terrain texture** and a batch file to help with this process:  download both snowpack.zip and sppatch.zip from Train-Sim.com.  One way to install this texture is to adapt his instructions to your route's case (see further below for an easier approach using Route-Riter).  By running Geerling's "Improved snow texture.exe" you can send his version of US2TarGnd.ace file to your route's TERRTEX\SNOW folder (by changing his default destination).  The rest of his installation, however, is designed for the default USA2 (Marias Pass) route:  his USA2Snow.bat copies the file US2TarGnd.ace from the USA2\TERRTEX\SNOW folder onto all the other files of that folder.  You can adapt his batch file to operate on your route by replacing all the destination file names on the right with those present in your route's TERRTEX\SNOW folder (you may have to add or delete lines).  You will need to place this batch file in your route's main folder and run it from there.  Note that Geerling's batch file backs up the original textures in a BACKUP folder, for the case you later change your mind.  You could combine his texture with Route-Riter's capability

CAUTION:  Ruben Geerling's batch file also copies all object texture files from your route's \TEXTURES\WINTERSNOW folder to its TEXTURES\WINTER folder, performing the operation discussed in section 3.10.12.

To combine Geerling's better texture with Route-Riter's efficiency, you should first run Geerling's "Improved snow texture.exe" to send his modified US2TarGnd.ace file to the USA2 route's TERRTEX\SNOW folder (by keeping his default destination).  (You may want to first back up the original US2TarGnd.ace file for safety.)  Then apply the "Improve snow" option of Route-Riter after selecting your route.

NOTE:  You should combine this deep-snow texture with **deep-snow tracks**, so the trackbed and ties also appear covered by snow:  see section 3.3.14.

NOTE:  **You cannot change in which combinations of weather and season the two different varieties of terrain textures will appear.**  The Snow variety (in folder TERRTEX\SNOW) only shows up for certain combinations of weather and season, namely:  clear winter, snowy autumn, snowy winter, and snowy spring.  Otherwise, the user gets the Normal "summerlike" variety (in folder TERRTEX).  This can have a strange result:  if you have copied the WINTERSNOW textures over the WINTER textures, and then run MSTS in rain during winter, you will see snow on the trees and other objects, but not on the ground!

### 3.9.2 USING TERRAIN TEXTURES

After being imported into your route (see section 3.9.1), **the terrain textures can be applied to individual patches of your terrain, as follows**:

- open your route in RE, and open the Texture selector window (as above);

- to display textures in the Texture selector window, click Edit - Insert, click on a *.ace file and click Open;  repeat for other textures as needed (but don't open more textures than will fit in the texture display window);

- press F7 to show the tile/patch grid;

- select one patch (left-click on it) or a rectangle of patches (while pressing Shift, left-click on two patches at opposite corners of the rectangle);

- press the number of the displayed patch which you want to apply to the selected patch(es):  the number runs from 1 up to 9;  if you have selected more than 9 patches, you can cycle through them by pressing Q (you may be able to enlarge the Texture selector window and thereby make more textures visible:  I can only enlarge the window to the right when it contains too many textures to show them all).

After applying the new texture, you have **several more options, obtained by selecting the patch(es), and right-clicking to open a pop-up menu**:

- you can **copy and paste a texture from one patch to another** (you do this faster with Ctrl-C and Ctrl-V, without using the pop-up menu);

- you can **rotate, mirror or randomize the texture(s)**:  this allows more control and more variety;  randomizing gives a random rotation to a rectangle of selected patches;

- the tiling factor allows you to **fill a patch with smaller copies of a texture**, such as 2 x 2 = 4 copies of half-sized textures;  however, these cannot be separately rotated, etc.;

- the error bias allows you to **impose greater detail**:  use zero error bias for the best detail.

[NEW SINCE V2]

### 3.9.3 DRAWING ON TERRAIN TEXTURES

It is possible to **draw lines on the terrain textures** in the RE, by using the Draw on Textures tool, also called Draw Tool or Pencil Tool (after pressing F8 or clicking on its icon shown in section 3.1.7).  This can be useful in several ways:

- you may draw lines on textures as a guide for laying tracks and/or roads, or other objects, perhaps after you have shaped terrain and/or made rivers or shorelines;

- after laying tracks and/or scenery, you may draw important lines on the underlying texture as a guide for "repainting" the texture to match what you placed there;  for example, you could then paint on the texture specific trackbeds, rivers, lakes, roads, gardens, rocks, or any other "2-dimensional" features you wish.

Keep in mind that each modified terrain texture takes additional space.  For instance, if you paint a river on a terrain texture, you will likely need to create one new texture for each patch along the river (since a terrain texture fills one 128m x 128m patch).  Also there is a limit of 64 different terrain textures within an area of about 2x2 km (about one tile).

In some cases, it may be more effective to use transfers that you repeat wherever you want (exploiting their flexibility of placement, orientation and scale).  For example, you may create a single transfer showing a short section of river or road, and place copies of it with variable orientations and sizes to form a meandering river or winding road.  On the other hand, transfers may slow down the frame rate.

**To draw on a terrain texture**, do the following:

- load your route into RE;

- identify the terrain texture on which you wish to draw, like this:  make sure the Texture window is open;  press F7;  click on the desired patch (bordered by black lines);  see the texture name that appears at the top of the Texture window;

- find the terrain texture with that name in the TERRTEX folder of your route;

- uncompress that terrain texture (for example with TGATool2, see Appendix H:  load the *.ace file into TGATool2, then export it without any compression, and without any alpha channel);  make sure it ends up again in the route's TERRTEX folder (it will be much larger now);

- you may repeat the above for other terrain textures that you wish to draw on (but if you wish to make different versions of a single terrain texture, you only need to identify and uncompress that single one);

- reload your route:  this makes the uncompressed texture(s) available to RE;

- move to the patch whose texture you want to draw on (you may press F7 to see the patches);

- press F8;

- draw lines with the mouse (they will be black);  apparently you can't erase lines that you have drawn, except by reloading the route (without saving any textures);

- if desired, draw lines in other patches (if you try to draw in a patch whose texture is compressed, you will see no lines drawn there);

- save the route:  answer yes to "Save modified texture?";

- enter a new file name for each modified version of the terrain texture(s), such as new1.ace, new2.ace, etc. (if you use an old name, you will overwrite the original texture and your drawing will later appear all over your route in every patch that uses that terrain texture!);

- if desired, repaint the new terrain texture(s) (see Appendix H), making sure you place the result in the route's TERRTEX folder;

- produce SNOW versions of the new terrain textures, if the route should work in snowy conditions in MSTS (they may be simple copies, with no snow effect, but they must be present); the SNOW versions should have exactly the same file names (such as again new1.ace, new2.ace, etc.);  place these in the route's TERRTEX\SNOW folder;

- if the repaints are final, compress the texture(s) to save space (see Appendix H);

- you still need to place the modified terrain textures in your route:  reload the route;

- place the new terrain textures as described in section 3.9.2;

- save the route as usual.

NOTE:  You may see gaps in drawn lines near patch boundaries, or if you draw too fast.

### 3.9.4 IMPORTING TRANSFERS

While terrain textures fill complete square patches, **transfers can be made to cover any rectangular piece of terrain, oriented in any direction**:  they are pieces of texture that can be placed, moved and rotated like objects;  however, they conveniently stick to the ground, so you don't have to worry about them floating in the air.

Transfer files, like texture files, are identified by their names of type *.ace.

Transfers come in two types:  the terrain texture files that we discussed in sections 3.9.1 and 3.9.2, and "true" transfers whose file names contain the letters "Trans" for "transfer" (such as US2GrassTrans.ace).  These *Trans*.ace files are specifically designed as transfers.

One difference between terrain textures and "true" transfers is that terrain textures have four sharp edges, while "true" transfers have only one sharp edge and 3 semi-transparent edges.  The semi-transparent edges allow a transfer that is laid over a patch texture to more smoothly merge into the underlying texture.  The sharp edge of a transfer is useful for covering holes near tunnel entrances (see section 3.11).

While "true" transfers are very similar to terrain texture files, the "true" transfers are placed in a different folder (Textures instead of Terrtex), and they have to be listed in the *.ref file of your route.

**To identify which terrain textures you want to use as transfers:**

- perform the steps described for selecting textures in section 3.9.1;  IMPORTANT:  keep a list of their file names in a WordPad or similar file;

**To identify which "true" transfers you want to use:**

- one (laborious) way to view transfers is to open a default route, visit its tunnel entrances, press W to see the opaque transfers, select a transfer placed there (by left-clicking its green handle) and find its name by right-clicking;  record the names of the transfers you want in a file for later use;

- an easier way is to place transfers on the default route to see them, using the placement tool, after pressing F5 and selecting a transfer in the Object selector;  however, not all transfers are listed in the Object selector, for example none are listed in the JAPAN2 route, although many are used there!  record the names of the transfers you want in a file for later use;  **don't save your changes to the default route!**

- an even easier approach is to choose all *Trans*.ace files from a default route;  record all their names in a file for later use.

Now that you know which *.ace files you will use as transfers, you can **copy these to your route**:

- copy the desired *.ace files **from the Textures subfolder of the default route(s) into the Textures subfolder of your route**:  NOTE:  the destination is now the **Textures subfolder, not the Terrtex subfolder**!

Next you must **modify the *.ref file that you will find in the main folder of your route** (if your route is called NewRoute, the file will be called NewRoute.ref).  The *.ref file lists objects that your route can use, and where you will find them in the Object selector (transfers are handled as objects):  so you must add to the list the new files that you will use as transfers.  Do the following:

- make a backup copy of your route's *.ref file and put it in a safe place;

- using WordPad or a similar editor, open the *.ref file;

- look for the item(s) entitled Transfer;  if there are none, go to the end of the file;

- in the *.ref file, you will now need to add, for each file that you will use as a transfer, a set of 6 lines like these (this example assumes that you are adding the file US2RockTrans.ace):

```
Transfer (
	FileName ( US2RockTrans.ace )
	Class ( Transfers )
	Align ( None )
	Description ( US2RockTrans )
)
```

**Make sure that you include the lonely closing parenthesis on the 6th line!** (It seems that the number of leading spaces is not important: I used tabs here because the MSTS files use tabs.)

WARNING: In my experience, **the new transfer may not show up the first time you restart RE again** (although it may show up after another restart or two). I suspect that this has something to do with the various ways that the Class name is listed, like Transfers or "Transfers" or <Transfers> or "<Transfers>": these all seem to be legal, but mixing different versions may result in some entries being lost. So try to use the same uniform style for the Class in the *.ref, for example only <Transfers>.

You are now ready to use these textures as transfers in your route, as described next.

### 3.9.5 USING TRANSFERS

You place, move and rotate a transfer in much the same way that you would an object. The major difference is that transfers stick to the terrain surface, and adopt its every shape like a skin.

In addition you can choose the rectangular sizes of a transfer.

**To place a transfer:**

- press F5;

- click the More.. button of the Placement window;

- find the transfer in the "Transfers" Object class;

- point the + sign with the mouse where you want to place the transfer, and left-click.

**To size a transfer:**

- if the transfer is not selected yet, press F2 and select the transfer (by left-clicking on its handle, which may be hiding in strange places on a steep slope! it may even lie deep underground!);

- press F6 or right-click to open a pop-up menu;

- choose Transfer and enter a height and width (in meters), and click OK (initially, the height refers to the North/South length, while the width refers to the East/West length).

**To move a transfer:**

- if the transfer is not selected yet, press F2 and select the transfer;

- press F3;

- drag the transfer with the mouse (it helps to position the camera so that you move the transfer from left to right, or from right to left;  on steep slopes it helps to make the camera look straight down).

**To rotate a transfer:**

- if the transfer is not selected yet, press F2 and select the transfer;

- press F4;

- rotate the transfer by dragging the mouse to the left or to the right (note that true transfers, as opposed to terrain textures, have only one sharp edge).

CAUTION:  A transfer can be distorted, in particular by "tilting" it:  if you drag the mouse up and down after pressing F4, you will see it stretch and change in strange ways, especially in steep terrain.  If it becomes too distorted, better delete it and start with a fresh one!

[NEW SINCE V2]
## 3.9.6 MAKING NEW TERRAIN TEXTURES AND TRANSFERS

CREDIT:  I am endebted to Steve Thompson for teaching me how to make transfers, including those illustrated in this section.

You make terrain textures and transfers in essentially the same way, as square textures.  The main difference between terrain textures and transfers is that transfers normally include an "alpha" channel that shows which parts are transparent.

The next image shows several transfers:  each is composed of a texture (upper half of each pair) and an alpha channel (lower half of each pair).  Take for example the transfer labeled "rock-bush 4-edged":  you may recognize it in the first figure of section 3.9, at top left and top center (above and below the viaduct);  three copies of it were indeed placed there.  But notice that the transfer has soft edges on the terrain there:  that is due to the transparency defined by the alpha channel along the four edges of the transfer.  The variable transparency makes the visible image change gradually from that of the transfer texture to that of the underlying terrain texture.  Black in an alpha channel means fully transparent, while white means totally opaque;  gray means partly transparent.

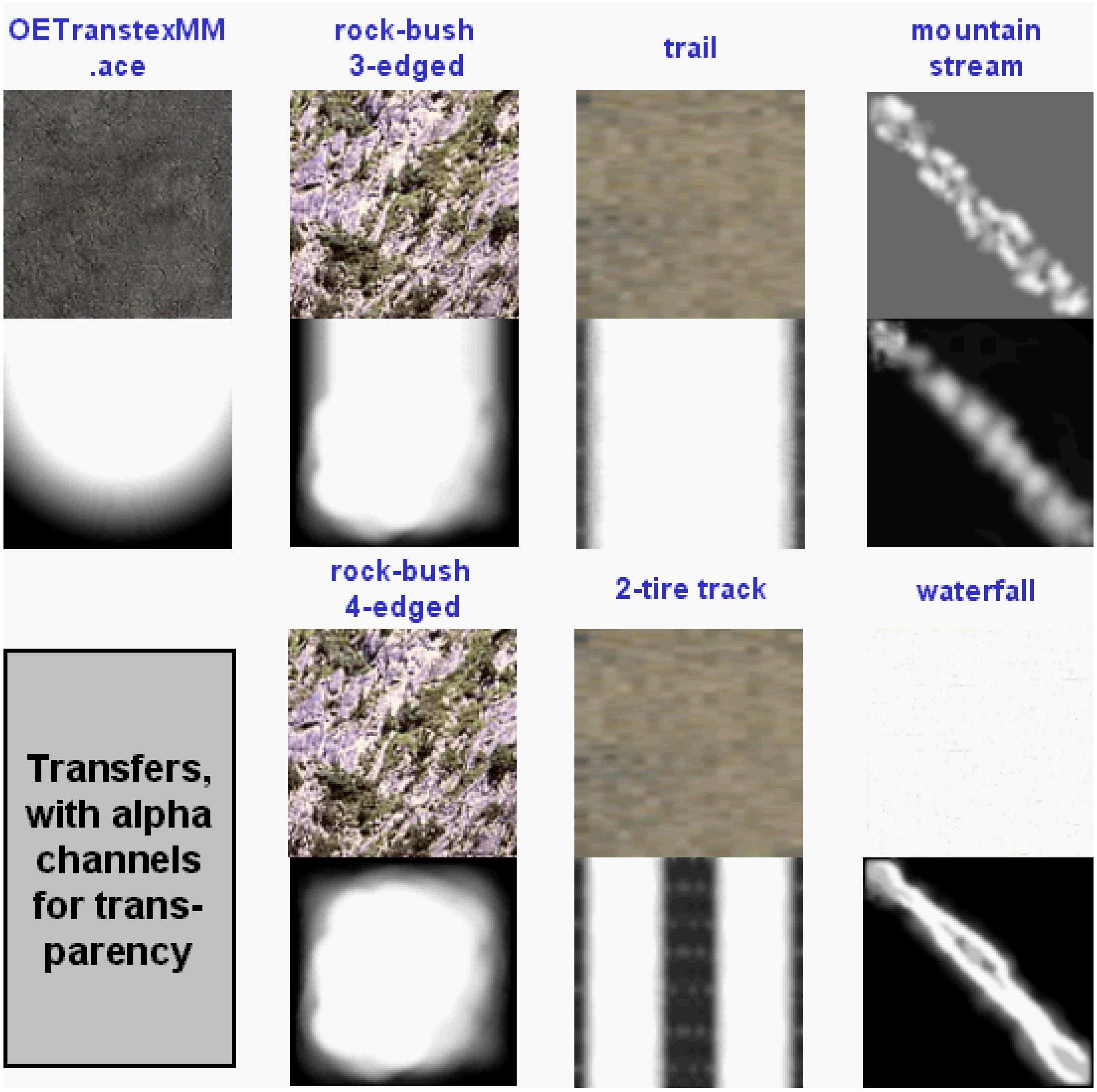
Below you also see a transfer labeled "rock-bush 3-edged":  its "3-edged" alpha channel makes one of the four edges sharp, and is to be used around tunnel entrances.  Another example below is the OETranstexMM.ace transfer (a default transfer from the EUROPE2 route): its alpha channel shows clearly a sharp edge at its top, also used for tunnel entrances.

The transfer labeled "trail" below is designed to make the footpaths seen in the first two pictures of section 3.9 and the wider trail of the second picture there: it has soft left and right edges to mark the edges of the trail. It is used by giving it a width of 1 to 2 meters and a length that can be anything from a few meters to 100 meters or more. The transfer labeled "2-tire track" below is used in the same way to produce the 2-tire track visible in the second picture of section 3.9.

The "mountain stream" and "waterfall" transfers below use the same principle as the trail transfer. They are drawn diagonally to make better use of the size of the transfer. They also show that you have some freedom in whether you put the main feature (white water in this case) in the texture itself, or in the transparency, or in both.

These examples can help guide you in designing your own transfers, and also your own textures (which don't have alpha channels). See Appendix H for details about the procedures involved in achieving this.

The pixel size of textures can be varied to some extent, as long as you keep them square and stay with "powers of 2" like 128x128, 256x256, 512x512 or 1024x1024. The choice will depend on the use, especially on how much the texture or transfer is stretched out on the terrain: a smaller pixel size helps reduce file size, but also may require stretching out a lot, which magnifies the pixel character of the texture; if this is not satisfactory, use a larger pixel size.




**U. Building "First Route": importing and using objects - platforms and tunnel entrances**

You will next learn to import and use objects from default MSTS routes. This will allow you to fill the landscape with unlimited everyday items that make it look inhabited. See section 3.10 for more details.

First you will **import and place platforms** at the two stations of First Route. Then you will import and place tunnel entrances.

Since the First Route is located in the UK, let's choose the "UK Station 1" from the Settle-Carlisle route. I made this choice by opening that route in RE and looking at its stations. This station actually consists of two platforms and two station buildings, as well as a few fences, all in one piece.  It resides in the files station1.s and station1.sd, and requires the texture Stationtex2.ace (which has a Night and a Snow versions in addition to the normal version).

For efficiency, let's already import the necessary files for the tunnel entrance that you will be placing.  Let's choose the "JP1 Tunnel Entrance", which requires the files JP1tunnel1t.s, JP1tunnel1t.sd, JP1tunnel1t.ace (one version) and tunnel_single.ace (two versions).  We will also need transfer textures for the tunnel, in the colors of the terrain where the tunnel entrances will be placed:  UKBScrub.ace, UKGGrass.ace, and OEDirt.ace;  these we have already copied (in two versions) to the Terrtex and the Terrtex\Snow folders, but we need to also copy them to the Textures and Textures\Snow folders, respectively and declare them as transfers.  You can see why many people prefer to copy complete folders from a default route to their new route!

**To import these needed files, do this:**

- exit from RE;
- copy station1.s and station1.sd from the EUROPE1\Shapes folder to the FirstRoute\Shapes folder;
- copy Stationtex2.ace from the EUROPE1\TEXTURES folder to the FirstRoute\TEXTURES folder;
- copy Stationtex2.ace from the EUROPE1\TEXTURES\Night folder to the FirstRoute\TEXTURES\Night folder;
- copy Stationtex2.ace from the EUROPE1\TEXTURES\Snow folder to the FirstRoute\TEXTURES\Snow folder;
- copy JP1tunnel1t.s and JP1tunnel1t.sd from the JAPAN1\Shapes folder  to the FirstRoute\Shapes folder;
- copy JP1tunnel1t.ace from the JAPAN1\TEXTURES folder to the FirstRoute\TEXTURES folder;
- copy tunnel_single.ace from the JAPAN1\TEXTURES folder to the FirstRoute\TEXTURES folder;
- copy tunnel_single.ace from the JAPAN1\TEXTURES\Snow folder to the FirstRoute\TEXTURES\Snow folder;
- copy UKBScrub.ace, UKGGrass.ace, and OEDirt.ace from the FirstRoute\TERRTEX folder to the FirstRoute\TEXTURES folder;
- copy UKBScrub.ace, UKGGrass.ace, and OEDirt.ace from the FirstRoute\TERRTEX\Snow folder to the FirstRoute\TEXTURES\Snow folder.

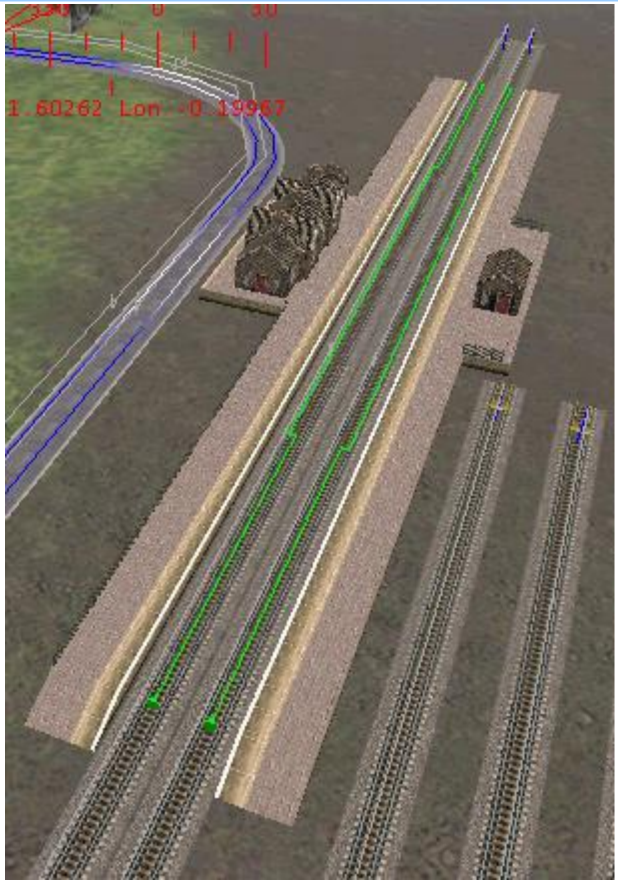
Now we need to **declare the two new objects and the three transfer textures in the *.ref file** of First Route.  To do this, open the file "FirstRoute.ref" (it is in the main FirstRoute folder), and add the following lines (you can copy these lines from inside the file settleca.ref in the EUROPE1 folder, and inside the file odakyuse.ref in the JAPAN1 folder):

```
Static (
      FileName ( station1.s )
      Class ( "Track Objects" )
      Align ( None )
      Description ( "UK Station 1" )
)
Static (
      FileName ( JP1tunnel1t.s )
      Class ( "Track Objects" )
      Align ( None )
      Description ( "JP1 Tunnel Entrance" )
)
Transfer (
      FileName ( UKBScrub.ace )
      Class ( <Transfers> )
      Align ( None )
      Description ( BrownScrub )
)
Transfer (
      FileName ( UKGGrass.ace )
      Class ( <Transfers> )
      Align ( None )
      Description ( GreenGrass )
)
Transfer (
      FileName ( OEDirt.ace )
      Class ( <Transfers> )
      Align ( None )
      Description ( Dirt )
)
```

Now you may restart RE to **place the platforms**, as follows:

- select "UK Station 1" in the Track Objects;
- press F5;
- point between the two tracks of the Central London station:  the complete station appears (as wire frame);
- press F3 and move it in place (it is already oriented North/South, parallel to the tracks);
- press F5;
- repeat the process at the North London station;  but here you probably need to rotate the platforms;  also place it sufficiently far along the tracks to not obstruct the nearest yard rail;  and press Y to adjust the terrain.

The result at North London should look something like the following figure.  This screenshot was taken after adding a station road, and after adjusting the green platform definitions to match the platform ends (just pull their handles along the track, and set the side on which the passengers will get on and off, by selecting a handle, pressing F6 or right-clicking, and entering your choice).




**Save your work, after deselecting any selected track or objects!**

Now **add a tunnel entrance** at the first tunnel North of Central London.  This placement involves three steps:  making a hole in the terrain for the tunnel, placing the tunnel entrance, and using transfers to cover up any pieces of the hole that the entrance does not cover.  The exact number and placement of the transfer textures depend on the particular placement of the tunnel entrance with respect to the wire grid of the terrain.

The detailed instructions needed to achieve this are given in section 3.11, together with illustrations:  please read them there.

You may repeat the same procedure to place the other three tunnel entrances in the First Route, but choose the correct texture transfer to match the terrain texture.

**Save your work, after deselecting any selected track or objects!  It would also be good to back up your FirstRoute folder again.  And you might as well reboot your PC after that.**

Next we will add sounds to the route.

**To continue building our First Route, jump to the next blue box.**

## 3.10 Importing and using objects

**By default, when creating a new route, the RGE places just a few objects in the folder of your new route.** Only those few objects are available for insertion in your route: these objects do **not** include tunnel entrances, platforms, stations, buildings, etc. Included are roads, only one type of bridge (JP2bluebrg), one type of forest, one type of individual tree, and a few other objects that you will find in the Object selector.

The 6 default routes of MSTS include a lot of objects that you can use. However, very few of them are accessible after the standard creation of your new route by RGE: unfortunately, **no route in MSTS can use objects that reside in other routes' folders**; **a route can only use the objects that reside in its own folder**.

To use objects from the default routes, **you must therefore import the needed objects to your route and list them in the *.ref file of your route**.

[NEW SINCE V1.106] NOTE: You can at any time change the "Class" of an object in the *.ref file. In fact, some objects are listed in different classes in a new route compared to default MSTS routes (such as dynamic tracks, which are listed under "Track sections" in default routes). This also means that when you import a *.ref file from another route, you may find your objects reassigned to different classes. In any case, you are free to change the "Class" of an object to anything you like; for example, you may decide to list the default bridge JP2bluebrg.s from being listed in the class "TrackObjects" to a new class "Bridges"; for that you would change in the *.ref file the Class line in

```
Static (
      Shadow ( RECT )
        Class           ( TrackObjects )
        Filename        ( JP2bluebrg.s )
        Align           ( None )
        Description     ( JP2bluebrg )
)
```

to

```
        Class           ( Bridges )
```

(use quotation marks if the class name contains spaces, such as "Bridge sections").

[NEW SINCE V2] **WARNING**: You will see different conventions used for naming the class of an object, such as Bridges, "Bridges", <Bridges>, and "<Bridges>". You can choose any one of these conventions for a given class, but **you should not mix different conventions in the same *.ref file**, or you risk losing some of the entries: they may not become available when you try to place them in RE. So use EITHER Bridges, OR "Bridges", OR <Bridges>, OR "<Bridges>", but not a mix of these conventions.

IMPORTANT: When you import files from other routes, **you must keep an accurate record of which files you copied from which folders**. Then you can tell other users of your route which files to use.

## 3.10.1 IMPORTING OBJECTS FROM OTHER ROUTES

WARNING: **If you distribute your route to other users, you must tell them to also import the necessary objects, or include them in your distribution package.**

WARNING: **If you distribute your route to other users and include imported or downloaded third-party add-on objects, you should keep a list of those objects. For third-party objects, you should keep track of their authors, ask for permission to distribute their objects (at the very least as a form of courtesy, even if they are labeled as freeware) and acknowledge the authors in your documentation.**

WARNING: **An object may use more than one file.** An object is defined in a *.s shape file, which is often accompanied by a *.sd shape definition file; it may also use one or more texture files of type *.ace. For example, the JP1Single Tunnel is stored in the file JP1tunnel1t.s, and the accompanying file JP1tunnel1t.sd, and uses the textures JP1tunnel1t.ace and tunnel_single.ace. Furthermore, **many textures have more than one version (with the same name)**, such as a "normal" (daytime summer) version, a "night" version, a "snow" version, an "autumn" version, etc. (these are stored in the Night, Snow, Autumn, etc. subfolders of the Textures subfolder). If you import a texture, **you should import all existing versions**. Otherwise, when you switch to a situation in which MSTS requires another version, but does not find it, it will crash.

One difficulty in importing objects is to find the corresponding textures: they often do not have corresponding names. This is discussed further in section 3.10.1d.

Besides importing objects from other routes, **you may also download add-on objects from the web** and install them in your route. Increasingly, add-on objects are becoming available for free download, for example in the library of http://www.Train-Sim.com. Such add-on objects are generally handled in the same way as objects imported from other routes: typically, they will consist of *.s and perhaps *.sd files, as well as *.ace files that must be inserted, respectively, into the SHAPES and TEXTURES folders of your route; in addition, you may have to insert a reference to them in your route's *.ref file.

There are **several options** for importing objects from default MSTS routes, which strike different balances between labor and disk usage.

The easiest way to add objects to your route is to restrict yourself to just **one** of the default routes, but to import all its objects: you will then be able to use only that route's objects. The disadvantage is that your route will use much disk space (something like 200 Mb).

Another way is to use all objects from several or even all default routes. That gives you the widest choice of objects, but uses a huge amount of disk space (something like 800 Mb if you import all objects from all default routes!).

To reduce disk usage, you may select just those objects that you will actually use. But this requires more work by you, because you must make a complete list of those imported objects and add it to your route's *.ref file.

These options are described in more detail in the next subsections.

### 3.10.1a Importing ALL objects from ONE default route

**To use objects from a single default route, do the following**. We use the example of your route called NewRoute, into which we will import all objects from the default Marias Pass route:

- build your new route (NewRoute) with the RGE;

- immediately or after laying tracks and shaping terrain, copy all the files (and subfolders) that are in the default route's subfolders EnvFiles, Shapes, Sound, Terrtex, and Textures (within the USA2 folder for Marias Pass) into the corresponding subfolders EnvFiles, Shapes, Sound, Terrtex, and Textures of your new route (within the NewRoute folder); note: you probably will not use all these files, but this makes them available and is easiest to manage;

- delete the *.ref file that is in your new route's main folder (such as NewRoute.ref), but remember its name (remember what that * stands for, namely NewRoute in our example);

- copy the *.ref file from the default route's main folder (such as marias.ref for the Marias Pass route) into the main folder of your new route (NewRoute);

- rename the *.ref file (such as marias.ref) so it becomes the same as the name of the *.ref file that you deleted earlier (NewRoute.ref; as a double-check, note that the * will be the same as for the *.ace file that should exist in your route's main folder, such as NewRoute.ace).

Now all those objects should be available through the More.. button of the RE's Placement window, while the textures and transfers also become available.

### 3.10.1b Importing ALL objects from SEVERAL default routes

If you want **to use objects from more than one default route**, you can copy them all as described above, but you will need to combine the *.ref files into a single *.ref file (without duplications), so that each object from the different source routes is listed there once.

If you combine all objects from all default routes, you must have a *.ref file that lists all those objects. For this purpose, a file called combined.ref is available for download as comboref.zip from Train-Sim.com. Follow its installation instructions. Make sure you have enough disk space for this!

### 3.10.1c Importing single objects

If you want **to use a restricted set of objects from one or more default routes**, you need to first identify their file names and folder locations, then copy them to your route's subfolder(s), and finally list them in the *.ref file of your route.  In the *.ref file, you need not include the accompanying files of type *.sd and *.ace:  you only need to list the shape file of type *.s.

**To identify objects in default routes:**

- a free add-on tool is available to view all the default objects from the different MSTS routes: the "HitchHiker's Guide to MSTS Objects" by Andrew J. Jones can be downloaded from Train-Sim.com (you need files mstsgde1.zip, mstspix1.zip, mstspix2.zip, mstspix3.zip and mstspix4.zip);  this catalogue gives you a picture of each object, together with everything you need to know about its associated file names and which default routes it can be found in;  note that version 1.0 has a few mistakes, in particular some objects are shown that cannot actually be used because they don't have a *.sd file;

- a related tool that both views AND moves an object from a default route (or even an add-on route) is the free add-on "Shape-Shifter" by Jacob Dicus, available from Train-Sim.com;  you must have installed the "HitchHiker's Guide to MSTS Objects" to view the objects;

- one (laborious) way to view objects is to open a default route, travel along the route to see objects, selecting an object placed there (by left-clicking it or its green handle) and finding its name by pressing F6 or right-clicking;  record the name(s) of the object(s) you want in a file for later use;

- an easier way to see objects is to place them on an empty region of the default route, using the placement tool, after pressing F5 and selecting an object in the Object selector;  record the file names of the objects you want in a file for later use;  **don't save your "additions" to the default route!**

**To identify the files that you will need to copy:**

- if you use Shape Shifter, follow its instructions and skip the rest of this section;

- after identifying an object that you want to copy, you should record its file name of type *.s (for example, the JP1Single Tunnel requires the file JP1tunnel1t.s);

- in the Shapes folder containing the desired *.s file, check whether there is a file by the same name but ending in *.sd (for example, JP1tunnel1t.sd);  this will also need to be copied;

- find whether texture files are needed by the object:  see section 3.10.1d for doing that;

- check whether other versions of the same texture files exist in subfolders of the Textures folder (for example, Night and Snow and seasonal versions):  also see section 3.10.1d for doing this.

**To copy objects from default routes to your route:**

- if you use Shape Shifter, it will automatically move the objects and their textures to the correct location;

- copy the files that you want from the default route(s) to your route, maintaining them in the appropriate subfolder: for example, many object files reside in the Shapes subfolder of a default route, and must therefore be copied into the Shapes subfolder of your route; make sure you also copy any *.sd versions, and any other versions (Night, Snow, etc.); you will also often need corresponding *.ace files.

**To list objects added to your route:**

- if you use Shape Shifter, it will automatically update the route's *.ref file; all you need to do is check that you can find the copied objects in the RE;

- copy the entry for each copied object from the *.ref file of its default route to the *.ref file of your route.

For example, assume that you have copied the file bridge2_usa.s from the USA2 route to your route; then you will find in the USA2 route's marias.ref file the corresponding entry

```
Static (
      FileName ( bridge2_usa.s )
      Shadow ( DYNAMIC )
      Class ( "Bridges" )
      Align ( None )
      Description ( bridge2_usa )
)
```

Copy that set of 7 lines (including the lonely closing parenthesis on the 7th line!) to your route's *.ref file (if your route is called NewRoute, that file will be called NewRoute.ref).

Repeat this for each object that you copied to your route.

### 3.10.1d Finding the corresponding texture files

**Many objects require texture files (of type *.ace), and these must be imported together with the *.s files (and *.sd files if they exist).** However, it is not simple to find those texture files: unlike the *.s and *.sd files, which have the same name (such as Tunnel1_noturrets.s and Tunnel1_noturrets.sd), the texture files normally use a different name (tunnel_turrets.ace in this example!).

[NEW SINCE V1.106] The easiest way to find those texture files is to use lists worked out by Dick van den Hoven and posted as file crossref.zip at Train-Sim.com: these lists will tell you, for each *.s file, what the corresponding *.ace files are and which Night, Snow or seasonal variations are present.

Another way to find those texture files is to try placing the object with RE, after you have imported the *s and *.sd files and declared the object in the route's *.ref file:  RE will warn you if a file is missing, giving its name, so you can copy it into the Textures folder of your route.  If more texture files are needed, this warning will happen again upon your next try:  copy the next file;  etc. (check for Night, Snow and seasonal variations!).

You may also decode the *.s file yourself:  it contains the name(s) of the texture file(s) that it needs.  The freeware Route-Riter has a convenient option to decode ("decompress") *.s files.  Such decoding can also be done with the ffeditc_unicode.exe utility contained in the Utils\Ffedit folder of MSTS.  This is a DOS program that cannot run directly under Windows;  you can run it in a batch file, as follows:

- write the following line in any word processor:

```
ffeditc_unicode in.s /o:out.txt /k
```

- save it to a new text file called decode.txt in the Utils\Ffedit folder of MSTS;

- rename this text file to decode.bat (this makes it a batch file);

- copy the *.s texture file that you want to decode into the Utils\Ffedit folder of MSTS;

- rename this texture file to in.s;

- in Windows Explorer or My Computer, double-click on decode.bat:  this executes ffeditc_unicode.exe in a DOS window;

- close the DOS window;

- open the new file out.txt (found in the Utils\Ffedit folder of MSTS) with WordPad or a better word processor (but not Notepad!);

- search for all occurrences of the letters "ace" (without the quotes):  that gives you the required texture file names.

[NEW SINCE V1.106] When Night, Snow or seasonal variations of the textures are present, these have exactly the same names as the "normal" (daytime summer) files, but are stored in different folders (Night, Snow, etc.).  So it is very important not to mix the different versions!  For example, tunnel_turrets.ace (in the US2 route) exists both in the "normal" form (in folder Textures) and in a "snow" form (in folder Textures\Snow):  the same name is used for both files, but they contain different variations.

[NEW SINCE V1.106] To determine whether Night, Snow or seasonal variations of the textures are present, you have several options:

- each *.sd file contains a line like "`ESD_Alternative_Texture ( 252 )`": the code number in this line tells you which alternate textures are present, as follows:

0: only "normal" (daytime summer) textures (in the Textures folder)
1: alternate textures in the Textures\Snow subfolder
252: alternate textures in the Snow and seasonal subfolders
(Snow, Autumn, AutumnSnow, Spring, SpringSnow, Winter and WinterSnow)
256: alternate textures in the Night subfolder
(CAUTION: this option may fail and should be avoided;
it is not used by the default MSTS objects; use 257 instead)
257: alternate textures in the Night and Snow subfolders

- the lists by Dick van den Hoven in file crossref.zip also contain the MSTS code number for each *.s file;

- you can also look yourself in the Night, Snow and seasonal folders to check whether *.ace files are present there for a given object.

[NEW SINCE V2] While the Autumn, AutumnSnow, Spring, SpringSnow, Winter and WinterSnow texture versions have obvious meanings, Night and Snow are actually less clear.

Night textures are obviously used at night (if ESD = 257 or 256), but they replace any other versions, such as snowy versions: so a Night texture will be used in all seasons and in all weather conditions, the night time being specified by the *.env files in the route's ENVFILES folder.

According to "OldieDaddy", who explored many combinations of season, weather and time, the Snow texture version is chosen by MSTS:
- only if ESD = 1 or 257;
- only with snowy weather in autumn and spring, or with any weather in winter
(not when it snows in summer);
- but if ESD = 257 at night, the Night version is used instead of the Snow version.

## 3.10.2 PLACING OR DELETING AN OBJECT

You **place an object** the same way that you place a track section (see section 3.3.4):

**- press F5;**

**- click the More.. button of the Placement window;**

**- find the object in one of the Object classes;**

**- point the + sign with the mouse where you want to place the object, and left-click**.

[NEW SINCE V1.106] It is possible **to restrict the placement of objects to a predefined grid of points**: This is useful, for example, to create rows of trees, as in an orchard, or to place bridge sections in perfect alignment. Pressing G, or checking the "grid" box on the Placement Tool, will restrict the + placement pointer to that grid, allowing you to place objects only at those grid points. By default the grid is a set of points forming 100m x 100m squares. You can change the square grid dimension by overtyping the value 100 visible in the box below the word "grid" on the Placement Tool (you may have to find the appropriate grid dimension by trial and error, for instance the bridge length if you want to place bridge sections one after another).

The grid is oriented north/south and east/west. After placing a set of aligned objects at such grid points, you may move and/or rotate them together as a block. To do that, press G or uncheck the "grid" box on the Placement Tool, then select all objects to be moved or rotated (by pressing Ctrl while left-clicking on each, as explained in section 3.10.5), and then move or rotate them as given in sections 3.10.3 and 3.10.4.

Some objects (like trees or houses) can be placed anywhere, while others are normally attached to existing objects (like road sections, similar to tracks).

[NEW SINCE V2] **Some objects can be snapped together, as toggled by the keypress X**: this applies in particular to some fences. The snap-on mode allows you to connect those objects into a perfectly straight string of identical objects. (The snap-on mode appears to be an unfinished option in RE: it works only with a few objects, and not very well.)

You start by placing and orienting the first object in the line (using the methods given in this and the following sections). Next you need identical objects of the same orientation. The easiest way to do this is to "copy and paste" the initial object (see Section 3.10.6), since that maintains the orientation of the initial object. Another method is the following:

- press X to toggle on the snap-on mode;

- place another identical object near the initial one;

- select the new object;

- press F3;

- with the mouse, drag the new object across the initial object until the new object takes the orientation of the initial object (this requires trial and error, and can be difficult with sloped objects).

Now the new aligned object(s) can be snapped onto the initial object to form a string:

- if you have not done so yet, press X;

- select a new aligned object;

- press F3;

- drag the new object into position at the end of the string of objects, until it snaps into place;

- after finishing the string of objects, press X to toggle off the snap-on mode.

CAUTION: It is easy in the snap-on mode to place multiple copies of the same object in exactly the same position, overlapping each other: that must be avoided. To check that you have no such overlapping objects, select each object in the string to check that it turns red; if it does not turn red, delete it; repeat the process until the object turns red.

[NEW SINCE V1.106] **To delete an object:**

- **press F2**;

- **left-click on the object** to select it;

- **press Delete**.

### 3.10.3 MOVING AN OBJECT

**To move an object, first do this:**

- **press F2**;

- **select the object (or its colored handle) by left-clicking on it**;

- **press F3**.

You now have several options (note that, if you use the mouse, the first mouse click will reposition the object to where the mouse is pointing!):

**To move an object horizontally:**

- **drag it with the mouse, while pressing Ctrl**: it will stay in a horizontal plane, no matter in which direction you drag it;

- or **use the left- and right-arrow keys to move it perpendicular to your view direction (pressing Shift accelerates the motion)**: you must view exactly sideways to make the object move in the desired direction;

**To move an object vertically:**

- **use the up- and down-arrow keys (pressing Shift accelerates the motion)**;

- or **drag it up or down with the mouse**, but **only if the camera looks horizontally**;

**To move an object roughly:**

- **drag it with the mouse, without pressing Ctrl**: **this moves the object perpendicular to your viewing direction (along the plane of your monitor)**; dragging left or right will move the object left or right, but dragging up or down will move the object up or down at an angle determined by the camera's viewing direction.

Often, the object will end up above or below the surface: **press H to put the object back on the surface**.

NOTE: It may be perfectly fine to place an object partly underground. This applies in particular to some bridge objects that have very tall pillars: make those pillars stick underground as needed.

### 3.10.4 ROTATING AN OBJECT

[NEW SINCE V1.106] NOTE: Rotating an object does not always work exactly as described below in RE. Some objects and some rotations behave differently. In particular, some rotations about one axis may also cause a rotation about another axis. As a result, the fine-tuning of the orientation of an object in RE can be very difficult.

We first indicate methods for rotating objects that are performed with the keyboard and/or the mouse in RE: these are relatively rough but quick. Then, we discuss more complex and slower methods for precisely turning objects to specific directions, by editing their direction in the route's world tile files.

In addition, a tool called **Object Rotator** is included with this Guide: it **allows precisely orienting objects in a variety of ways**. This approach, however, is also rather labor-intensive, because it involves finding and modifying direction data contained in a route's world tile files.

It will be helpful to think in terms of the bank, slope and heading of an object. Each object in MSTS can be imagined to have a "nose", a "tail" and "wings" (much like an airplane), whose directions together fully define the orientation of the object. Then:

- an object's bank is a sideways tilt, so that one wing drops down while the other rises up;
- an object's slope is a lengthwise tilt, so that the nose rises or drops, while the tail drops or rises;
- an object's heading is a horizontal rotation to a certain compass direction, so that the nose turns right or left in a horizontal plane.

**To rotate an object, first do this:**

- **press F2**;

- **select the object**;

- **press F4**.

You now have several options:

**To rotate an object around its vertical axis:**

- **use the left- and right-arrow keys** (on the number pad): the object will not tilt (it will keep its vertical orientation), but you can't fine-tune the rotation angle (actually, if you keep pressing End simultaneously, the object turns very slowly, but it is so slow as to be useless);

- or **drag the mouse left or right**: this allows fine-tuning; however, if you don't drag exactly left or right, but also a bit up or down, the object will tilt northward or southward;

**To rotate an object around the east-west axis:**

- **use the up- and down-arrow keys** (on the number pad): the object will tilt to the north or south, but you can't fine-tune the rotation angle (again, using the End key simultaneously does allow fine-tuning, but the rotation is too slow to be useful);

- or **drag the mouse up or down**: this allows fine-tuning.

**To rotate an object around the north-south axis:**

- **use the up- and down-arrow keys** (on the number pad)**, while pressing Ctrl**: the object will tilt to the west or east, but you can't fine-tune the rotation angle (again, it is possible but too slow with the End key);

- or **drag the mouse up or down, while pressing Ctrl**: this allows fine-tuning.

**Pressing Shift together with any of the above actions accelerates the rotation**.

**To rotate an object back onto the ground, press N**: this reorients it parallel to the ground (sloping if the ground is sloped).

**To reset the original orientation of the object, press O**: this gives it the orientation it had when you placed it.

[NEW SINCE V1.106] **To turn an object by exactly 180° around the vertical axis, without changing its bank or slope:**

If you want to roughly turn an object by 180° around the vertical axis, it is best to do it visually, using the methods given above.

But if you need to turn an object by exactly 180° around the vertical axis (as for an already correctly-aligned bridge or platform), you will have to edit the object's data in your route's world tile files. **Do this as follows:**

- find the corresponding world tile file in your route's WORLD folder (see section 3.10.9);

- open that world tile file with WordPad and find the object in it (see section 3.10.9);

- modify the parameters in the QDirection line from

```
QDirection ( a b c d )
```

to

```
QDirection ( -c d a -b )
```

(this means: change the signs of the last two parameters, labeled c and d here, then move them before the first two parameters);

- save the world tile file;

- reload your route in RE (without saving, but with the Advanced option to rebuild the track database if the rotated object is a track or road section), and go to the object: it now should have exactly the new orientation;

- shift it into place.

[NEW SINCE V1.106] **To change the slope direction of an object (from up to down, or vice versa), keeping exactly the same degree of slope, but reversing the bank (so the object still fits the oppositely inclined terrain):**

If you want to roughly change the slope of the object, it is best to do it visually, using the methods given above.

But if you need to keep exactly the same degree of slope while changing it from up to down, or down to up, you have to edit the object's data in your route's world tile files. **Do this as follows:**

- find the corresponding world tile file in your route's WORLD folder (see section 3.10.9);

- open that world tile file with WordPad and find the object in it (see section 3.10.9);

- modify the parameters in the QDirection line from

```
QDirection ( a b c d )
```

to

```
QDirection ( -a b -c d )
```

(this means:  change the signs of the first and third parameters, labeled a and c here);

- save the world tile file;

- reload your route in RE (without saving, but with the Advanced option to rebuild the track database if the rotated object is a track or road section), and go to the object:  it now should have exactly the new orientation;

- shift it into place.

[NEW SINCE V1.106] **To turn an object by exactly 180° around the vertical axis, and at the same time change its slope direction (from up to down, or vice versa), keeping exactly the same degree of slope, while reversing the bank (so the object still fits the same inclined terrain):**

You can combine an exact 180° rotation around the vertical axis with an exact change of slope direction, **as follows**:

- find the corresponding world tile file in your route's WORLD folder (see section 3.10.9);

- open that world tile file with WordPad and find the object in it (see section 3.10.9);

- modify the parameters in the QDirection line from

```
QDirection ( a b c d )
```

to

```
QDirection ( -c -d a b )
```

(this means:  change the signs of the last two parameters, labeled c and d here, then move them before the first two parameters);

- save the world tile file;

- reload your route in RE (without saving, but with the Advanced option to rebuild the track database if the rotated object is a track or road section), and go to the object:  it now should have exactly the new orientation;

- shift it into place.

[NEW SINCE V1.106] **To place an object parallel to another object:**

You may wish to place one object parallel to another.

If they are copies of the same object, the easiest way to do this is through copying and pasting (see section 3.10.6), because that gives the copy the same orientation as the original.

For objects of different type, a rough visual alignment is often sufficient (using the methods of this section).

The exact alignment of objects is much more difficult to achieve, for example to place a long bridge or platform parallel to a straight track. This is because rotating an object in RE around one axis often also makes it rotate around another axis: as you fine-tune the orientation around the vertical axis, the object also tilts in a complicated way around the North-South and East-West axes (whether you use the mouse or the keyboard).

To get exact alignment, you need to edit an object's QDirection parameters in your route's world tile files, by copying the orientation of one object onto another object. Let's describe the process for a bridge section that we wish to place parallel to a track section (so as to let the track run on the bridge). **Do this as follows:**

- place the bridge section anywhere near the desired location (within the same tile), and save (no need to exit from RE);

- find the corresponding world tile file in your route's WORLD folder (see section 3.10.9);

- open that world tile file with WordPad and find in it a track section along which you want to place your bridge (see section 3.10.9);

- copy the entire QDirection line of that track section; it will look something like this for horizontal tracks (note the two zeroes):

```
QDirection ( 0 -0.633265 0 0.773935 )
```

or something like this for sloping tracks:

```
QDirection ( -0.0180033 0.186487 -0.00341799 0.982287 )
```

- find, in the same world tile file, the entry for the bridge section that you placed and saved earlier;

- overwrite its QDirection line with that copied from the track section;

- save the world tile file;

- reload your route in RE (without saving, but with the Advanced option to rebuild the track database if the rotated object is a track or road section), and go to the new bridge section: it now has exactly the correct orientation (including the track slope);

- shift it into place (under the track for a bridge, or next to the track for a platform, etc.).

You can make further copies of this oriented object and position the copies one by one, for example to make long bridges and platforms. (For very long bridges and platforms, you can go faster by doubling, see section R: copy and paste a pair of parallel objects, then a quartet, then an octet, etc.).

### 3.10.5 MOVING OR ROTATING A GROUP OF OBJECTS

You can move or rotate a **group** of objects (this does not work for track or road sections!):

- press F2;

- select an object;

- select other objects: but press Ctrl while selecting them;

- then move or rotate the group of objects as if they were a single object (they will rotate about their separate rotation axes).

### 3.10.6 COPYING AND PASTING OBJECTS

You can **copy and paste** an object or group of objects, while maintaining its (their) orientation (this does not work for track or road sections, for track objects, telephone poles, and transfers, among others!):

- press F2;

- select an object or group of objects;

- press Ctrl-C;

- press F5;

- point the + symbol where you want to place the copy of the object(s);

- press Ctrl-V;

- fine-tune the position and orientation of the copy (copies), if needed.

### 3.10.7 OBJECT DETAIL LEVEL

**The detail level of an object allows the train driver to control how many of the installed objects are actually seen**. This is useful to let users with slower computers "turn off" objects to increase their frame rates and reduce their computer's stuttering. It also allows better performance at high train speeds when scenery is less important, while permitting the user to increase scenery details for low-speed runs and sightseeing.

TIP: Show the frame rate in RE by pressing Shift-Z.

The user controls the number of objects that are visible through the Advanced Display options of MSTS: the World Objects Density can be set from 0 (fewest objects seen) to 10 (all objects seen).

Each object can be assigned a corresponding detail level of 0, 1, 2, …, 10: level 0 is always shown, while level 10 is only shown at the highest World Objects Density setting of MSTS.

[NEW SINCE V1.106] In RE, you can easily check the detail level of all objects in the camera's field of view: on the Camera window, you can change the detail level shown, by clicking on the small up- and down-arrows to the right of its "detail" setting. As the detail level drops from 10 to 0, objects become invisible when their detail level is passed (their shadows remain visible to tell you where they were).

All objects have a default detail level which may be adequate for your needs. But you may prefer other values, for instance to make objects that are distant from your track less important and thus invisible at a different World Objects Density setting of MSTS than for similar nearby objects.

To assign a detail level to an object, do the following:

- select the object;

- press F6 or right-click: this opens a pop-up menu;

- enter the desired Detail level under Properties.

### 3.10.8 OBJECT PROPERTIES

**Each object has various properties that you may change**.

You set an object's properties by selecting it and then either pressing F6 or right-clicking: this opens a Properties window, which often has two or more tabs to select from. These tabs offer various property options. Note that some properties that seem to be adjustable actually have no effect in practice (for example, many objects don't produce a shadow, even after you select a shadow type).

The following describes the more common object properties and their functions:

- **Detail level**: see section 3.10.7;

- **Shadow**: the choices are "None" (used mainly for tracks, sounds, and small or narrow objects like mileposts, signals and gantries), "Round" (used mainly for trees and bridges), "Rectangular" (used mainly for buildings), "Treeline" (used mainly for lines of trees), and "Dynamic" (used mainly for large objects that cast long sharp shadows); note that dynamic shadows depend on the sun's orientation and slow down the simulation more than other shadows;

- **Animate this object**: enabling this option causes an object that has moving parts to perform the movements within RE, so you can see whether its orientation is correct and how much free space it needs; it does not seem to affect the behavior of the object in MSTS;

- **Terrain object**: some objects must have this option enabled: for example, many bridges must be declared as Terrain objects so as to show the ballast and ties under railway tracks;

- **Collision properties**: after enabling "Collide with objects", the choices are "Immovable" for hard objects that will cause a fatal derailment, and "Buffer" for buffers that allow low-speed bumping; enabling these choices slows down the simulation; by default, no collision takes place (a train will drive through the object).

In addition, many objects have specific properties that govern their operation or appearance in the simulation, such as car spawners, level crossings, pickup objects, animated deer and forests.

[NEW SINCE V1.106]

### 3.10.9 FINDING AN OBJECT IN THE WORLD FILES

**After placing an object in your route, you may want to change its properties in some ways, or delete it, by editing the world files** rather than by selecting the object. For example, it may happen that due to some error, you can no longer select or delete an object in RE: you may then be able to delete it by removing it from the world file.

**CAUTION**: Be very careful when editing world files; back up the route before doing this!

Each track section, road section and other object that you place is entered into the world files of the tile on which it is placed. You can open each tile's world files (found in the WORLD folder of the route) with WordPad. Each tile has two world files, of type *.w and *.ws: **the *.ws file contains the tile's sound objects, while the *.w file contains all other objects on this tile**.

The world files have names like `w-012930+014198.w` and `w-012930+014198.ws`. The two numbers in each name label the tile (see section 2.7): they match the numbers that appear in RE as "tile x" and "tile z" in the Camera window (when the camera is within that tile). In this example, "tile x" = -012930 and "tile z" = 014198.

So **you can identify the world file that contains a given object by moving the camera near that object and recording its "tile x" and "tile z": those numbers tell you the name of the corresponding world file.** (Note that you do not have to close RE before opening and editing a world file.)

**Within that world file, you can identify the types of objects by their file names**. For example, you may have placed a tree with file name tree1.s. It will appear something like this in the *.s world file:

```
Static (
      UiD ( 4 )
      FileName ( tree1.s )
      StaticFlags ( 00002000 )
      Position ( -716.541 0.444601 188.964 )
      QDirection ( 0 0 0 1 )
      VDbId ( 0 )
)
```

However, there may be many copies of the object that you want to find in that world file. **To distinguish the various copies of the object, you can use their Positions.** If in RE you select the object you want to identify, you will find its coordinates within that tile shown in the Object window as "x:", "y:" and "z:". These numbers also appear in the line `Position ( ... )` of the object you are searching in the world file, so you can search for the object by its coordinate(s).

If you now change some properties in a world file, save the file (in Unicode format, which WordPad should do automatically), and reload the route into RE.

[NEW SINCE V1.106]

### 3.10.10 MAKING NEW OBJECTS

**If you need an object,** but can't find it in another route or on the web, **you can make it yourself**. The simpler situation is the case where you only need to change the texture of an existing object, often called "reskinning". More complex is the case where you want to change the shape of an existing object or create a new shape yourself.

To reskin an object you need to change one or more texture files: these are the *.ace files associated with the object, which you can find as described in section 3.10.1d. Appendix H describes how to manipulate *.ace files. You must supply the new graphics that will paint the existing object the way you want. If you distribute the reskinned object, make sure you give credit to the author of the object's shape.

Creating or changing an object's shape is a larger job; this requires specialized add-on software. One very popular payware used for this task is "Train Sim Modeler", formerly by Abacus (http://trainsimulatorworld.com/). Also much used is "gmax" (http://www.discreet.com/products/gmax/), which has both a restricted freeware version and a more complete payware version.

[NEW SINCE V2]
### 3.10.11 RESIZING OBJECTS

The size of objects is contained in their shape file (*.s): it cannot easily be changed by editing such a file. However, **you can resize an object with a freeware tool** called Shape File Manager (by Paul Gausden and available at Train-Sim.com as sfm.zip). It allows you to resize an object independently in its three dimensions. This will also work with most rolling stock.

If you want to resize an object, especially a default MSTS object, you should first make a copy and give it a new name (so the old object can still be used unchanged). You should also change the name of the corresponding *.sd file and the name of the *.s file that appears within the *.sd file (the *.sd file is not compressed and can be edited with a Unicode-capable editor like WordPad).

You must uncompress the "new" object's *.s file, then resize it, and finally compress it again. (If you have trouble uncompressing and compressing with this tool, try using the freeware Route-Riter.)

You may need to scale the ESD_Bounding_Box in the object's *.sd file by the same factor: do that manually, or let Shape File manager do it for you.

The textures will "stretch" with the size of the object, so they need not be resized.

[NEW SINCE V2]
### 3.10.12 SHOWING SNOW ON OBJECTS WHEN IT DOES NOT SNOW

**By default, MSTS only shows snow on objects (as opposed to the ground) when snow is actually falling**: this unfortunately prevents the user from having snow-covered objects in winter under a sunny sky or while it is raining.

**This is easily fixed**: simply copy all the files from your route's TEXTURES\WINTERSNOW folder over the files in your route's TEXTURES\WINTER folder.

## 3.11 Tunnel entrances and underground structures

**Tunnels need to pierce the terrain through a hole, and need entrance structures and transfers to cover up the hole**.

See sections 3.4.2 and 3.4.3 for making underground and underwater tunnels. We here only deal with making the tunnel entrances, and with placing underground structures like stations and switches in tunnels.

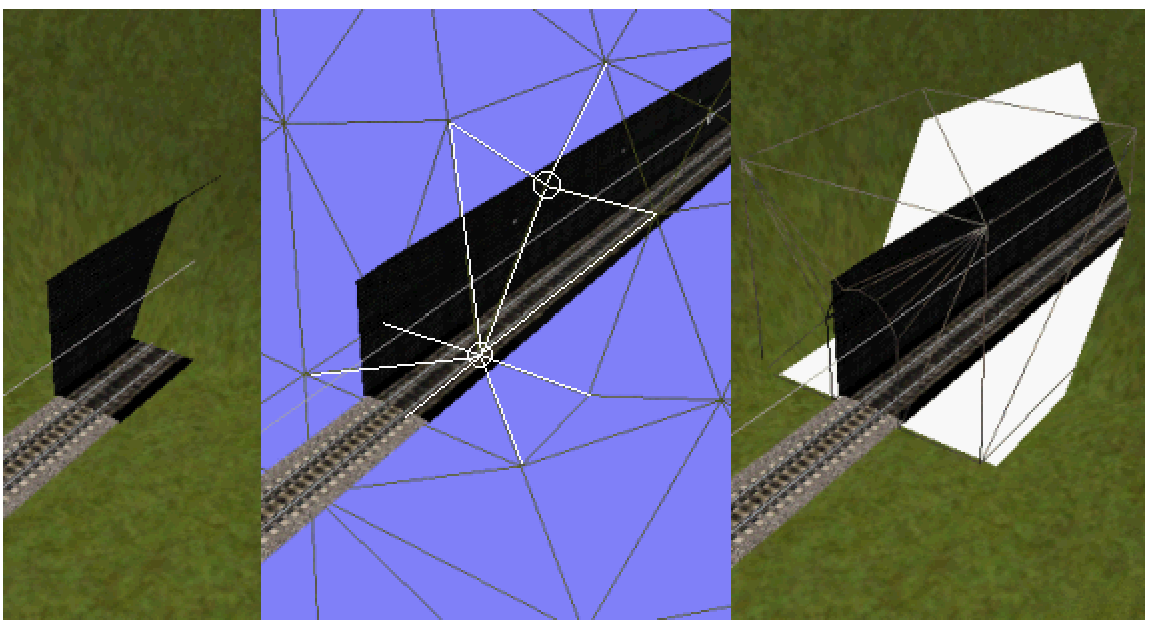
Without piercing the ground, the terrain forms a "curtain" that visually blocks the entrance to a tunnel (although it does not bother MSTS trains!): so we need to make a hole in the terrain where a tunnel emerges. The tunnel-covered track sections do not include proper structures to terminate a tunnel when it emerges from the ground:  so we need to add a tunnel entrance, which is an object.  In addition, a tunnel entrance may not cover up the entire hole:  so we will usually need to add "transfers" on the ground to cover up the remaining parts of the hole.

All this is a **rather complex task**:  it requires importing a tunnel entrance object and a transfer or texture from default MSTS routes, and placing them over a piece of terrain that has been made invisible (transparent).

### 3.11.1 AN EASY WAY OUT

By far the easiest way to make tunnel "entrances" is to **accept "curtains" of soil at tunnel entrances**, even if this is rough and unrealistic:  see the leftmost part of the next picture (the steep terrain piece that shows inside the tunnel is one such "curtain").  These "curtains" will be opaque as seen by the train driver.

To do this, manually shape the terrain at each tunnel entrance so as to make the curtains as vertical as possible;  try to position these curtains as close to the entrance as possible, but still on the inside of the tunnel (if they are placed outside, the train driver will be able to see the white underground emptiness that lies below the terrain).  You make these modifications by manipulating individual wire grid points (visible in the middle part of the figure), as discussed in section 3.5.1.



### 3.11.2 PROPER TUNNEL ENTRANCES

We now discuss how to make a hole in the terrain, add a tunnel entrance object and add transfers to cover remaining parts of the hole.  The details (like number and size of transfers) will change from one tunnel entrance to another.  In particular, **the details will depend on the location of**

**the tunnel entrance relative to the grid points of the terrain wire mesh, and on the mesh size** (this may depend on the value of the Terrain Detail Scaling Factor set in the RGE, see section 2.1, and on the error bias in effect for the tile, see section 3.1.5).

**Do the following:**

- we start from the tunnel entrance shown in the left part of the figure above;

- press W to show the terrain wire frame, as shown in the middle part of the figure above (I have highlighted several wires and wire junctions in this figure: they do not show like this in RE);

- press F9 and point at a grid point close to the middle of the tunnel entrance: two choices of such grid points are shown by the white circles in the figure;

- press V to make a piece of terrain invisible: in wire-frame mode you will just see grid lines disappear; press W to see the resulting hole in the terrain, as shown at right in the figure above; (pressing V is a bit tricky: it toggles the visibility back and forth at a very high speed, so you will succeed only half the time! just try again until you are lucky!);

- you may have to make different grid sections invisible, and you may need to make more than one grid point invisible, so as to eliminate any piece of terrain that might be visible inside the tunnel (look into the tunnel to make sure it is not obstructed!), but you also want to make invisible the least possible amount of terrain: in the figure above, both circled grid points have been made invisible, such that all the grid lines which I artificially highlighted in white become invisible;

- the result after pressing W is shown above at right (for now, ignore the wire grid showing the future tunnel entrance structure): this shows the hole in the terrain, with the black tunnel sticking out of it;

- once satisfied that you have a proper hole, save your route and quit RE.

We now need to import a tunnel entrance and a transfer into your new route.

Let's use the JP1SingleTunnel as a tunnel entrance: it resides in file JP1tunnel1t.s, which also needs files JP1tunnel1t.sd, JP1tunnel1t.ace, and tunnel_single.ace (in a normal version and a Snow version). See section 3.10.1 for details on how to find the necessary files.

- copy JP1tunnel1t.s and JP1tunnel1t.sd from the JAPAN1 route's Shapes subfolder to your route's Shapes subfolder (we will change the *.ref file a little bit later, when we have imported a transfer);

- copy JP1tunnel1t.ace from JAPAN1's Textures subfolder to your route's Textures subfolder;

- copy tunnel_single.ace from JAPAN1's Textures subfolder to your route's Textures subfolder;

- copy tunnel_single.ace from JAPAN1's Textures\Snow subfolder to your route's Textures\Snow subfolder.

Let's now import a texture transfer to cover up the hole. In this example I will use the terrain.ace texture that has the default green texture used for new routes (for some reason, it will look slightly different as a transfer than as a texture). That terrain.ace is already present in the new route's Terrtex subfolder, but needs to be copied to the Textures subfolder to serve as a transfer (in general you would import a transfer from another route, as discussed in section 3.9.4). This transfer and the shape file for the tunnel entrance also must be declared in your route's *.ref file, so **continue as follows**:

- copy the terrain.ace file (with the default green texture) from your route's Terrtex subfolder to your route's Textures subfolder;

- edit the *.ref file of your route (such as NewRoute.ref) by inserting the following 2 x 6 lines (try to place them near similar items labeled as Transfer and Static):

```
Transfer (
      FileName ( terrain.ace )
      Class ( <Transfers> )
      Align ( None )
      Description ( Terrain )
)
Static (
      FileName ( JP1tunnel1t.s )
      Class ( "Track Objects" )
      Align ( None )
      Description ( JP1SingleTunnel )
      TunnelEntrance ( )
)
```

- save and close the *.ref file;

- restart RE and reload your route;

- check that the tunnel entrance is present in the Track Objects class of the Object selector (under the description JP1SingleTunnel), and that the transfer is present in the Transfer class of the Object selector (under the description Terrain); (WARNING: if one or both are missing, check the warning at the end of section 3.9.4)

- place the tunnel entrance at the proper place with the correct orientation, as shown at right in the above figure (that figure shows the tunnel entrance in its selected wire-frame state, which is convenient for seeing behind it); see section 3.10 for placing, moving and rotating objects;

- deselect the tunnel entrance to see more clearly where the hole remains uncovered (you will have to move the camera around to see all sides): now you can judge the number and size of transfers needed to cover up those remaining hole pieces; in this example, only two are strictly needed: one over the top and one on the right-hand side of the tunnel entrance structure; I have nevertheless placed four transfers, as shown in the left part of the figure below: each transfer forms a rectangle pushed against the two sides, front and top of the black tunnel (you see them best when the terrain is shown as wire frame);

- move the tunnel entrance out of the way (don't rotate it!), to clear the terrain for laying transfers;

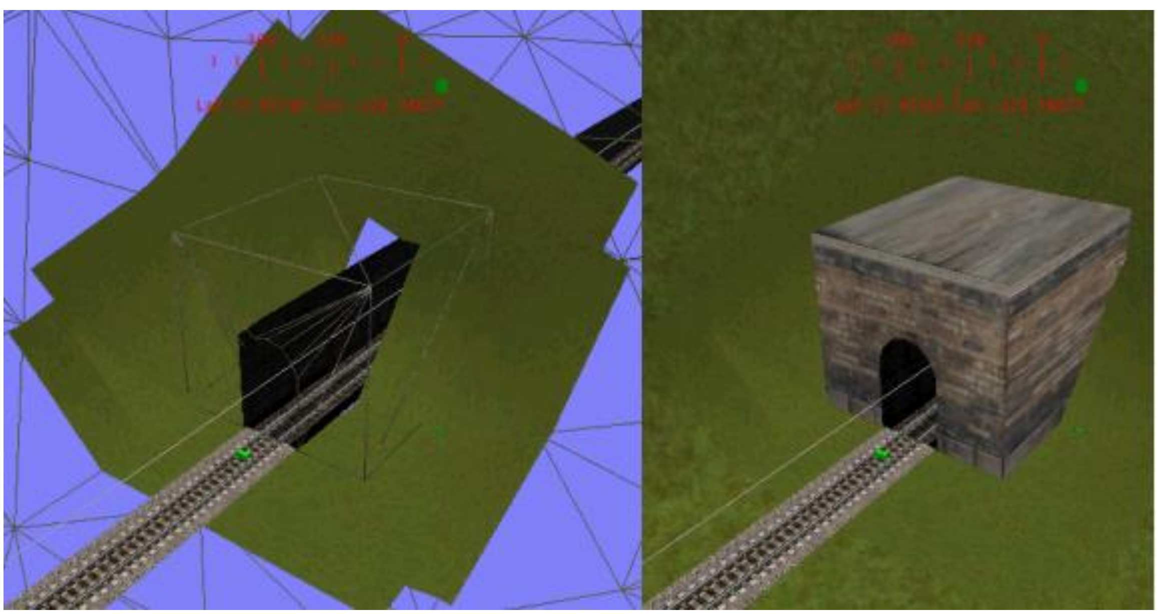
- place your transfers; move and rotate them with the green handles (all 4 handles are visible in the left figure below); I exaggerated the size of my transfers: they need not get so close to the black tunnel, and they need not reach so far away (I used dimensions of 30 x 15m and 15 x 30m);

- check that no transfers enter within the black tunnel (the top transfer especially can easily fall below the tunnel roof and become visible to the camera and the train driver);

- now place the tunnel entrance back where it should be;

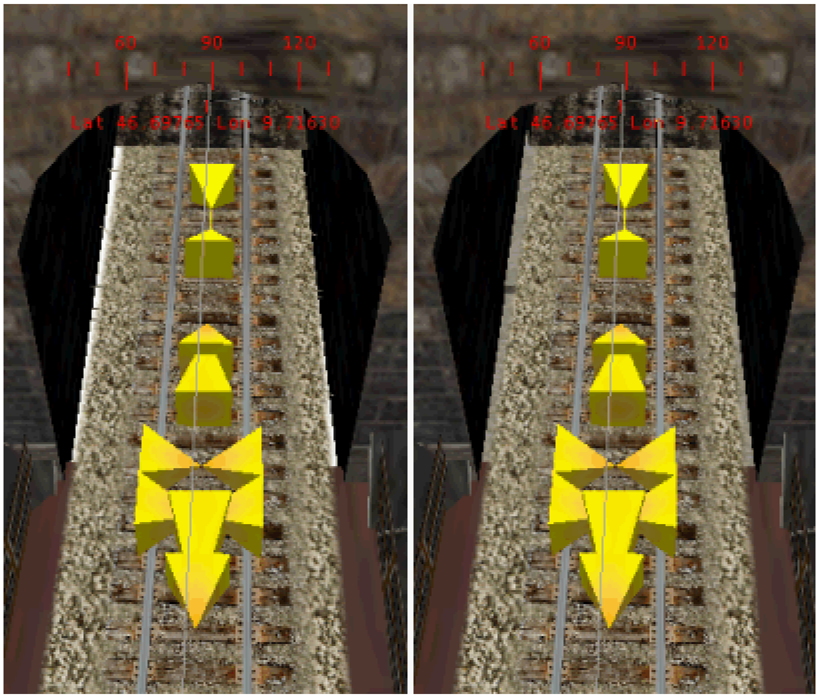
- check that everything fits.

The result is shown at right in the figure below.



NOTE: We used in this example a "texture transfer" instead of a "true" transfer. The only difference is that the true transfer would have three semi-transparent edges: those edges should be turned away from the tunnel entrance, so as to merge more smoothly with the underlying patch texture; the fourth, sharp edge of a true transfer should be turned toward the tunnel entrance.

[NEW SINCE V2] Also, pay attention to the area where the track enters the tunnel: it is easy to find gaps there through which the driver can look into the white underworld! The next pair of screenshots shows, at left, typical gaps (white) that can occur when the tunnel track section starts farther inside the tunnel than the hole in the ground (ignore the yellow handles): the trackbed is not wide enough to fill the gap between the rails and the tunnel entrance walls, while you may not be able to place transfers to fill this gap. One solution in this situation is to place solid objects under the gaps, to block the view of the underworld: I often place a gray concrete slab of type JP2CSlab10m.S under that section of track, as shown in the right part of the figure. (Remember that you can move the camera underground or into a tunnel after pressing /. You may need that to explore the situation near a tunnel entrance.)




### 3.11.3 UNDERGROUND STRUCTURES

Here we discuss how to put **switches underground**, as well as how to place **objects within tunnels**, such as for underground stations.

MSTS provides simple track sections with built-in tunnel walls and ceilings. However, no switches are equipped for use in tunnels. Also, while objects may be placed inside tunnels, the tunnel walls allow space for only very small objects (like signals and mileposts).

To build anything below the MSTS terrain is very difficult: it is necessary to lower the terrain and **build on the terrain surface**. Later, **when building is complete, the terrain can be raised above the tunnel**.

WARNING: **There is danger in burying tracks and objects that interact with tracks** (such as mileposts and signals). When pulling terrain up above tracks and objects, you can easily cause errors in the track database, etc. **Do this at your own risk!** It is highly recommended to **do this as late as possible** in the process of building a route: especially the raising of the terrain should be delayed as much as possible. Also, frequent saves, back ups and reboots are even more essential here!

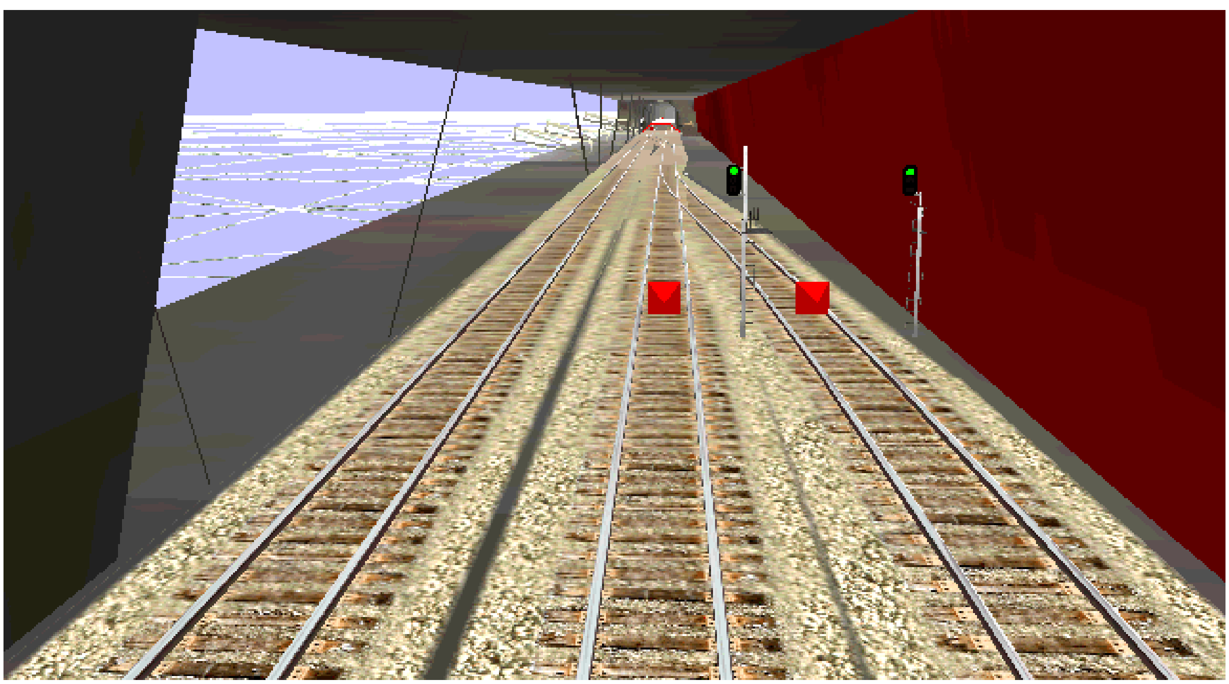
**A safe approach** that avoids burying anything is the following: place tracks at the bottom of a trench in the terrain, and surround the tracks with walls and ceiling to form the tunnel. Add any objects you wish within that trench, and leave the terrain as is: the terrain will always stay below the tracks. To do this, you may want to find and import objects than can serve as walls and ceiling. For the ceiling, one (imaginative but not very flexible) idea is to use the water surface, which is opaque and brown as seen from below: this essentially means building your route submerged at the bottom of a canal; note that rain and snow will fall inside this "tunnel"; (the freeware Port Ogden and Northern route of Jim Ward uses this approach very effectively to create a complete underground coal mine). If you want to build other tracks above this tunnel, you need to hide the trench with other objects.

For the walls you could use the terrain itself. Or you could look for the largest possible objects that have a suitable surface texture. These can also serve as ceiling. Large bridge sections and some wall segments are good candidates. My favorite is a huge slab that covers part of the Tokyo Shinjuku station: it is Jp1ShinjukuTunnel.s and Jp1ShinjukuTunnel.sd, with the texture JP1ShinjukuSt.ace. It is slightly curved and has many pillars, but if you turn it so the pillars point away from the tunnel, you get fine long straight walls and ceilings. Note that curved tunnels are more complex to make than straight tunnels, because curved walls are rare in MSTS.

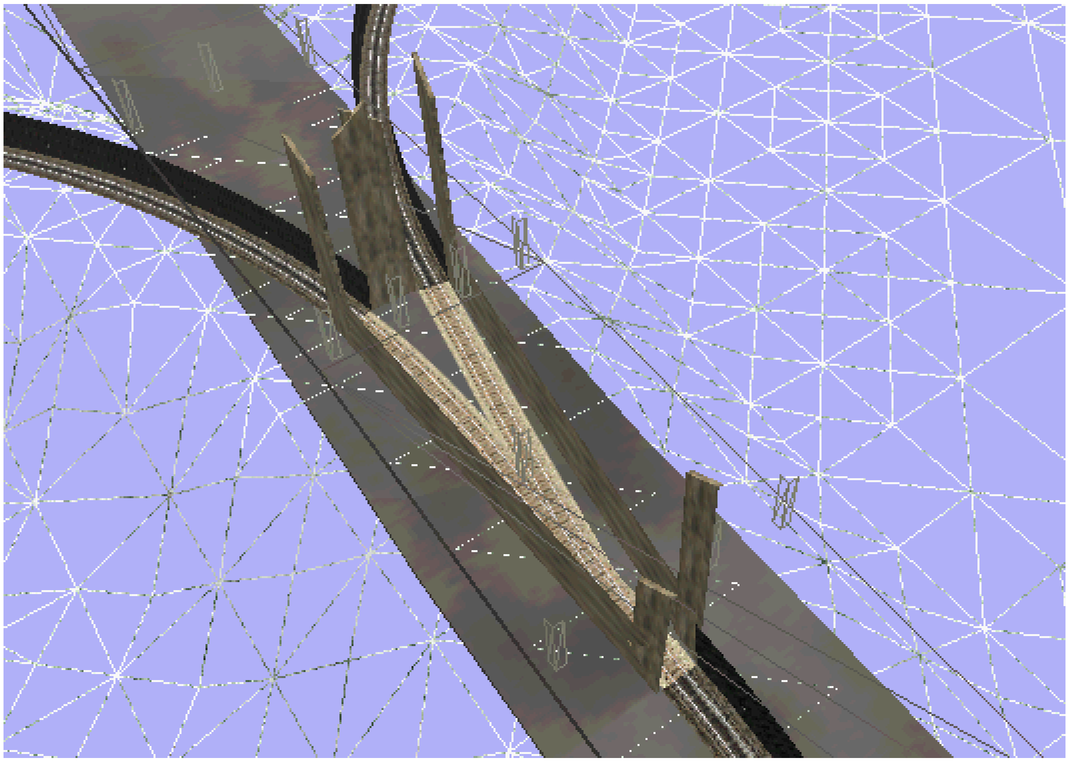
To place switches and larger objects, you need to create **larger underground spaces: caves**. You can build caves by assembling walls and ceiling over a foundation. The same large objects discussed above can be used for this. For example, you can put a horizontal slab as foundation, just below the tracks (which are made of regular non-tunnel track sections); then you add walls to form the sides of the tunnel, and place a slab overhead as ceiling.

An example of a cave in a very long mountain tunnel is illustrated in the next figure (this tunnel is not part of First Route). A standard dual-track tunnel arrives from the left and opens up into a cave (through a tunnel entrance to fill gaps between standard tunnel and cave). The Shinjuku slab is used as foundation (its pillars are sticking down, out of sight of the driver): it is necessary to change its Properties to Terrain object, and it helps to set no shadows in its Properties. Another copy of this slab is used as ceiling (upside down, with pillars sticking up, also out of sight). The back wall is shown in red: it is yet another copy of the same slab, tilted to form an inclined wall (its pillars are pointing away from the tracks, again out of sight). A similar wall exists on the near side of the tracks, and is here made transparent by being selected. The terrain is still below the tracks, but will be raised later.

The next picture shows another section of the same cave (whose purpose is to provide a third track for slower trains to let pass faster trains, and to create space for tunnel maintenance): again one wall section is selected and transparent. This cave includes interactive track objects (signals, speed limits, mileposts, even a crazy hiker), as well as other objects. With a large enough cave, you could place a station inside.



The next picture illustrates a smaller cave designed for a single switch.  The foundation is the same Shinjuku slab, and so is the ceiling (transparent here, but you can guess some of its pillars pointing up).  For walls, I have used OE60MWall.s from the Innsbruck-St. Anton route:  some of these are placed horizontally, others vertically to fill gaps.  Filling gaps is the most time-consuming part of building these caves!



You can also place **small objects inside a default MSTS tunnel section**:  in long tunnels especially, it may be good for the drivers to have mileposts and signals.  Such interactive objects can be placed in the normal way, when the tunnel-covered track is not buried underground.  You can let these objects stick outside the tunnel if necessary:  some signals are still too wide so you can push them partly out of sight outside the tunnel walls (this can even make them appear to be realistically attached to the tunnel wall instead of to a pylon).

When you have finished placing everything you want in the tunnel, you are ready to raise the terrain to bury the tunnel (but do it as late as possible!).  This is the **most dangerous part**:  make sure you save and back up your route before doing that.  And after raising the terrain, thoroughly test your route, before doing any further building.

## 3.12 Some special objects

### 3.12.1 PICKUP OBJECTS

Pickup objects along the tracks provide supplies for locomotives: coal, water, and diesel fuel. The next picture shows a coal tower at left, a water column in the center (servicing the tender), and a diesel pump in front.

A **water column** is available by default in a new route. To place it, select "Water Column" in the Pickup Objects class. Then place it between the rails of your track, and rotate it to be parallel to the track. To operate it in MSTS, press T (keep pressing) when a tender is next to it.

A **coal tower** must first be imported and declared. Copy the files coaltower.s and coaltower.sd from EUROPE1\SHAPES to your route's SHAPES folder, and copy coaltower.ace from EUROPE1\TEXTURES and from EUROPE1\TEXTURES\SNOW to the corresponding folders in your route. Also add the following lines to your route's *.ref file (using WordPad):

```
Pickup (
      FileName ( coaltower.s )
      Shadow ( DYNAMIC )
      Class ( "Track Objects" )
      PickupType ( _FUEL_COAL_ )
      Description ( "Interactive Coal Tower" )
)
```

To place a coal tower, select "Interactive Coal Tower" in the Track Objects class (that class is where MSTS has it in the EUROPE1 route: it would be more logical to put it in the Pickup Objects class; you could make that choice by writing Pickup Objects in the Class (…) line of the *.ref file). Then place it between the rails of your track, and rotate it to be parallel to the track. To operate it in MSTS, press T (keep pressing) when a tender is below it.

A **diesel pump** must also first be imported and declared. Copy the files US2fuelpump.s and US2fuelpump.sd from USA2\SHAPES to your route's SHAPES folder, and copy US2FuelStation.ace from USA2\TEXTURES and from USA2\TEXTURES\SNOW to the corresponding folders in your route. Also add the following lines to your route's *.ref file:

```
Pickup (
      FileName ( us2fuelpump.s )
      Class ( "Pickup Objects" )
      PickupType ( _FUEL_DIESEL_ )
      Description ( "Diesel pump" )
)
```

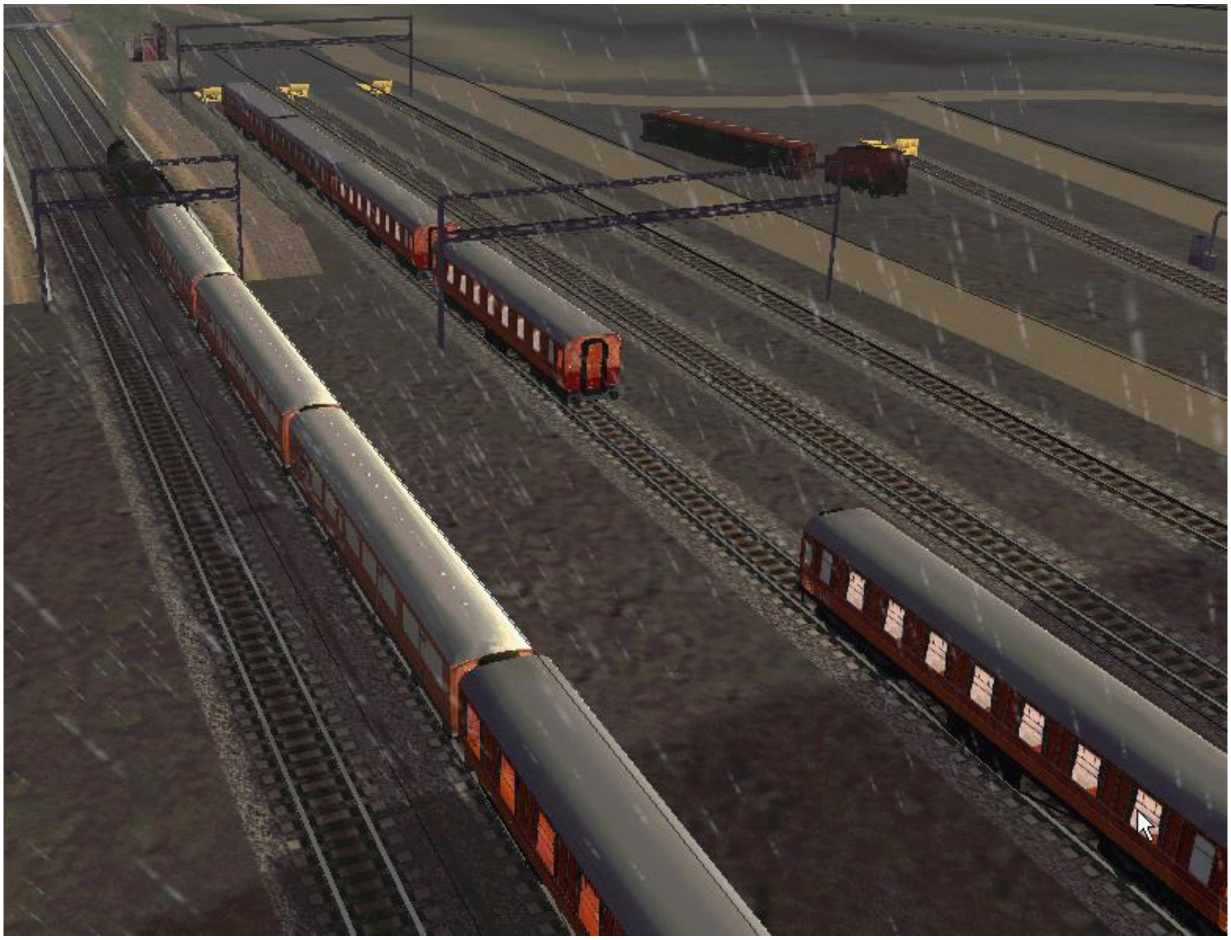
To place a diesel pump, select "Diesel pump" in the Pickup Objects class. Then place it between the rails of your track, and rotate it to be parallel to the track. To operate it in MSTS, press T (keep pressing) when a diesel locomotive is next to it.

## 3.12.2 STATIC LOCOMOTIVES AND RAILWAY CARS

There are two ways to place static locomotives and railway cars on the tracks of your route. The first is by defining them within activities: such rolling stock can be picked up by your engine, and thus become mobile.

The second way is to place rolling stock as objects. This requires importing and declaring the rolling stock like any ordinary objects. The main advantages of this method are: MSTS can run faster because it does not have to worry about whether and how this rolling stock is supposed to move; and you can place such rolling stock off the tracks and in any orientation, as in an accident.

In the next picture, only the train at left is mobile: the cars on the next track and those lying on their side further to the right are static objects (see also the picture in section 3.12.4).



As an example, let's import the Royal Scotsman steam locomotive, its tender, and a third-class Royal Scotsman car (wagon). You will need to copy the files ukroyalscot.s, ukroyalscot.sd, ukroyalscottender.s, ukroyalscottender.sd, scotsthirdclass.s and scotsthirdclass.sd from the TRAINS\TRAINSET\ROYALSCOTCLASS folder to your route's SHAPES folder. Also you must copy ukroyalscot.ace, ukroyalscottender.ace, scotcar1.ace, scotcar2.ace and scotcar3.ace from the same TRAINS\TRAINSET\ROYALSCOTCLASS folder to your route's TEXTURES folder. Then you must declare these three objects in your route's *.ref file, by inserting the following lines:

```
Static (
        Class           ( TrackObjects )
        Filename        ( ukroyalscot.s )
        Align           ( None )
        Description     ( Royal_Scotsman )
)
Static (
        Class           ( TrackObjects )
        Filename        ( ukroyalscottender.s )
        Align           ( None )
        Description     ( Royal_Scotsman_Tender )
)
Static (
        Class           ( TrackObjects )
```

```
        Filename        ( scotsthirdclass.s )
        Align           ( None )
        Description     ( Royal_Scotsman_Third_Class )
)
```

Now you can place any of these objects by selecting them in the TrackObjects class. They can be positioned anywhere you like, including on the tracks, and turned in any direction you wish.

### 3.12.3 OTHER MOVING ROAD VEHICLES

As explained in section 3.8, road traffic is controlled in part by the carspawner.dat file: this file lists the vehicles that travel along the roads.

Below is an example of the carspawner.dat file, as I prepared it for the First Route project (see the illustration in section S). The 7 lines labeled CarSpawnerItem contain 6 different vehicles: these will travel along the roads. One of the vehicles (car2.s) shows up twice, so it will appear twice as often as the other cars on the roads in MSTS. By repeating particular vehicles in this manner, you can make them appear more often (if you change the number of lines, adjust the line showing the number 7 to reflect that change).

```
SIMISA@@@@@@@@@@JINX0v1t______

7
CarSpawnerItem( "car1.s" 5 )
CarSpawnerItem( "c_van.s" 6 )
CarSpawnerItem( "car2.s" 7 )
CarSpawnerItem( "car4.s" 6 )
CarSpawnerItem( "c_car.s" 5 )
CarSpawnerItem( "Jp1Car01.s" 10 )
CarSpawnerItem( "car2.s" 13 )
)
```

Of the listed vehicles, Jp1Car01 is already present by default in a new route.

The other 5 vehicles must be imported: copy c_van.s, c_van.sd, c_car.s and c_car.sd from the EUROPE1\SHAPES folder to your route's SHAPES folder; copy c_van.ace, c_car0.ace and light.ace from the EUROPE1\TEXTURES, EUROPE1\TEXTURES\NIGHT and EUROPE1\TEXTURES\SNOW folders to the corresponding folders of your route; copy car1.s, car1.sd, car2.s, car2.sd, car4.s and car4.sd from the USA1\SHAPES folder to your route's SHAPES folder; copy car1.ace, cartwo.ace and car4.ace from the USA1\TEXTURES, USA1\TEXTURES\NIGHT and USA1\TEXTURES\SNOW folders to the corresponding folders of your route.

There is no need to declare these vehicles in the route's *.ref file.

The order in which the vehicles appear on the road is a randomized version of the list in the file: so the order in which they are listed is not important. The number to the right of each vehicle seems to specify the minimum gap to the next vehicle (in meters), to avoid collisions.

## 3.12.4 STATIC ROAD VEHICLES

It is possible to place non-moving cars (road vehicles) anywhere you wish, for example parked off the side of your roads: the same cars that are used by the car spawner can also be treated like any static object.

In the following picture, the off-road cars are static, while those on the road are mobile (also, the red locomotive and tender are static objects, as is the hiker):


To achieve this, you must import and declare the cars. Importing them has been described in section 3.12.3: look there for further instructions.

Next declare the 6 cars used by the car spawner described in section 3.12.3, by adding the following lines to your route's *.ref file:

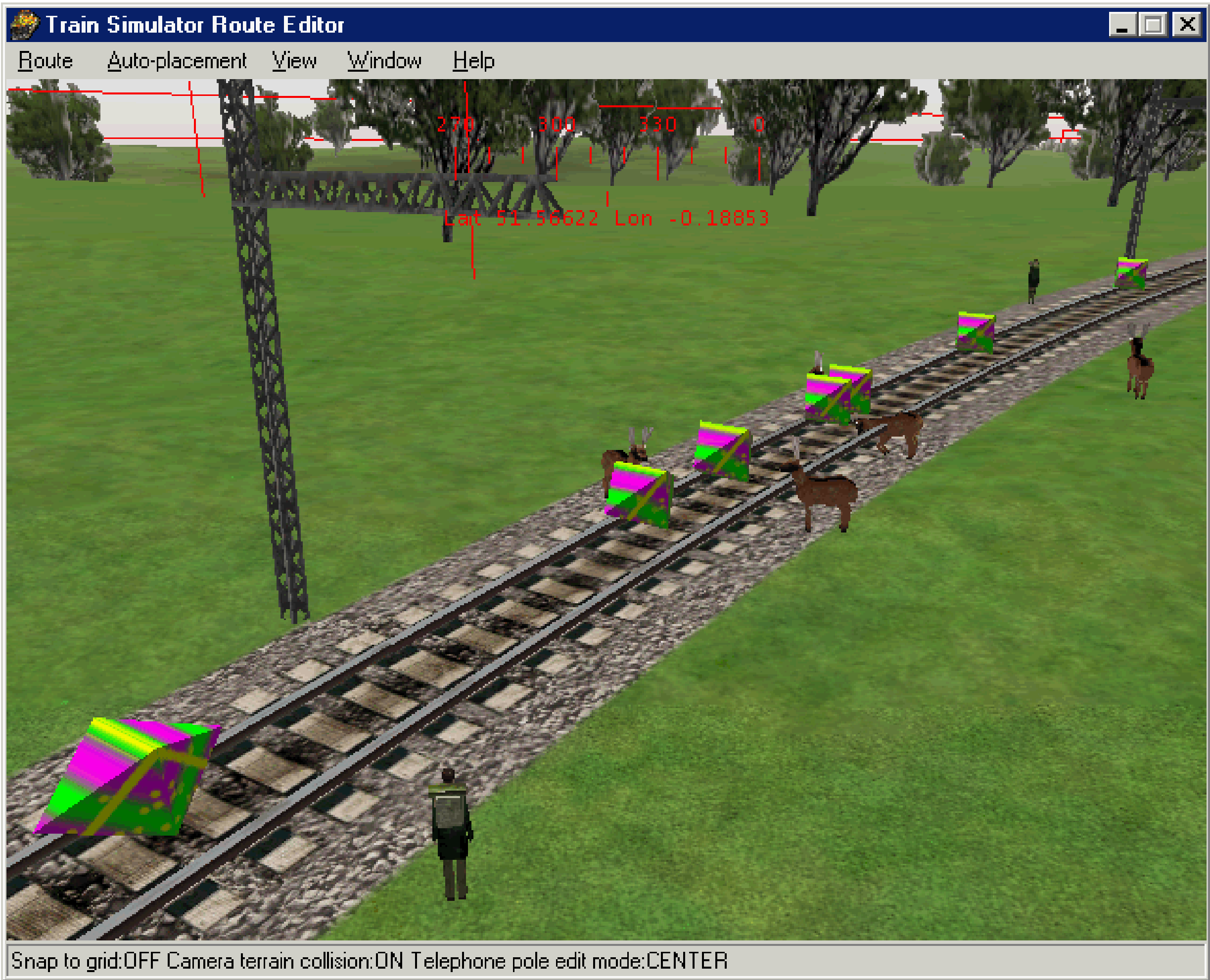
```
Static (
        Class           ( TrackObjects )
        Filename        ( Jp1Car01.s )
        Align           ( None )
        Description     ( JP_Car_01 )
)
Static (
        Class           ( TrackObjects )
        Filename        ( car1.s )
        Align           ( None )
        Description     ( JP_Car_1 )
)
Static (
        Class           ( TrackObjects )
        Filename        ( car2.s )
        Align           ( None )
        Description     ( JP_Car_2 )
)
Static (
        Class           ( TrackObjects )
        Filename        ( car4.s )
        Align           ( None )
        Description     ( JP_Car_4 )
)
Static (
        Class           ( TrackObjects )
        Filename        ( c_van.s )
        Align           ( None )
        Description     ( UK_Van )
)
Static (
        Class           ( TrackObjects )
        Filename        ( c_car.s )
        Align           ( None )
        Description     ( UK_Car )
)
```

Now you can select any of these cars in the TrackObjects class, and place them wherever you wish. They can be turned any way you like.

### 3.12.5 PEOPLE AND DEER

The default MSTS routes show very few people on this Earth. However, with the default objects you can populate the world with people, as long as they are either "female hikers" or "spotters"! In fact, the hikers and the spotters look exactly alike. The only difference is that hikers are permanently visible objects, while spotters are seen in MSTS only when running an Activity in which they have been activated (similar to animated deer, discussed in section 3.7.6).

The next picture from the RE shows two interactive spotters near a herd of interactive deer (each has a handle on the track).




We have already discussed deer as interactive track objects: they are available by default in a new route (see section 3.7.6). It is also possible to add permanently visible deer, as normal objects: see below.

To include spotters, hikers and static deer in your route, you must first import and declare them, as follows.

To import the **spotter**, copy the file spotter.haz from the EUROPE1 main folder to your route's main folder, and add the following lines to your route's *.ref file:

```
Hazard (
      FileName ( spotter.haz )
      Class ( <Hazards> )
      Align ( None )
      Description ( Person )
)
```

To place a spotter, select "Person" in the Hazards class: then place her between the rails along your track; move the spotter (hidden under the colorful handle) off the track, and orient her as you like.

[NEW SINCE V1.106] The default spotter is not animated:  she does not react to passing trains. But you can animate the spotters in a way similar to deer:  the spotters will then simply slide forward (rather than walk or jump) by a certain distance at a certain speed, especially when the train's horn or bell is sounded.  (Note that a spotter will slide along the ground, not above the ground on an elevated platform, for example, unless you raise the ground to make the spotter appear to slide on the platform.)

You can make this animation happen by changing the spotter.haz file of your route.  In file spotter.haz, replace the 5 lines

```
Idle_Key                  ( 0 9000 )
Idle_Key2                 ( 0 9000 )
Surprise_Key_Left         ( 0 9000 )
Surprise_Key_Right        ( 0 9000 )
Success_Scarper_Key       ( 0 9000 )
```

by the following 5 lines (you can copy them from any deer.haz file)

```
Idle_Key                  ( 59 178 )
Idle_Key2                 ( 46 59 )
Surprise_Key_Left         ( 36 46 )
Surprise_Key_Right        ( 25 35 )
Success_Scarper_Key       ( 0 24 )
```

Then change the Distance and Speed from 10 and 0 (no motion!) to something like 5 meter and 1 meter/sec (= 3.6 km/h ~ 2 mph).

One thing that you can do with this option is to make people move into or out of buildings at the sound of the train's horn or bell.

In the spotter.haz file, you can see the line

```
Workers ( workman.s )
```

According to the MSTS Tech Docs, this line makes a workman appear instead of a spotter when the train is in a zone of temporary reduced speed.

To activate spotters, move the People frequency hazard slider to the right in the Conditions and Hazards box of the Activity Editor.  The farther toward 100% you move the slider, the more likely will the spotters appear in an MSTS Activity:  at 50% only half the spotters will show up.

To import the (static) **female hiker** to your route, copy the files femalehiker.s and femalehiker.sd from EUROPE1\SHAPES to your route's SHAPES folder, and copy femalehiker.ace from EUROPE1\TEXTURES to your route's TEXTURES folder.  Also add the following lines to your route's *.ref file (using WordPad):

```
Static (
      FileName ( femalehiker.s )
```

```
      Shadow ( ROUND )
      Class ( "Clutter" )
      Align ( None )
      Description ( Female_Hiker )
)
```

To place a hiker, select "Female_Hiker" in the Clutter class:  then place her anywhere you wish. Make a group of hikers for more "static life".

To import the **deer**, copy the files US1deer.s and US1deer.sd from the GLOBAL\SHAPES folder to your route's \SHAPES folder, and copy the file US1deer.ace from the GLOBAL\TEXTURES folder to your route's TEXTURES folder;  then add the following lines to your route's *.ref file (using WordPad):

```
Static (
      FileName ( US1deer.s )
      Shadow ( ROUND )
      Class ( "Clutter" )
      Align ( None )
      Description ( Deer )
)
```

To place a static deer, select "Deer" in the Clutter class:  then place it anywhere you wish, and orient it as desired.

[NEW SINCE V1.106]
### 3.12.6 FORESTS

**MSTS allows placing forests of trees as objects.**  Somewhat surprisingly, they are classified as Track Objects, although they are in no way connected with tracks.

Forests are convenient in at least three ways:  first, they allow placing a large number of trees in relatively few steps;  second, they reduce the number of objects placed on a tile, since a complete forest counts as only one object;  and third, they improve frame rates, because the same tree is reproduced quickly many times.

By default in a new route, MSTS offers only one type of forest, namely a forest with trees of type "JP1Tree1".  **To plant** such **a forest**, you select the object "Forest - JP1 Tree" in the Track Objects, and place its center where desired.  Before unselecting this object, you can adjust its tree density:  right-click to open its Properties box, then select Forest and change its density:  a density in the range 500 to 5000 trees per square kilometer is more realistic than the small default value of 50.  Next you can adjust the size and orientation of the forest by selecting its two green side handles and dragging them after pressing F3.  (Warning:  making an enormous forest can crash RE!)  You may also shift the entire forest by selecting its central green handle, pressing F3 and dragging it (press H to put that handle at ground level, especially if you have accidentally moved it underground).  Note that all trees will stick to the ground as you move or rotate a forest, even on sloping terrain.

**It is possible to vary the size of the trees from one forest to another forest, or also within one forest.** Open the forests.dat file in the root folder of a new route (with WordPad). It looks like this:

```
SIMISA@@@@@@@@@@JINX0F0t______

1
Forest ( "JP1Tree1" "JP2AutoTree1.ace" 16.0f 20.0f 0.9f 1.1f )
```

This file defines one forest, named "JP1Tree1", using the file JP2AutoTree1.ace (and its seasonal variations). The tree width and height are 16m and 20m, respectively. And the size of the trees within the forest varies by a factor ranging randomly between 0.9 and 1.1: so the trees will vary in size up to 10% from their default size of 16m x 20m. **You can increase the variation of the tree size in a given forest by increasing the range 0.9 to 1.1** in the file forests.dat, for example by choosing the range 0.7 to 1.3.

**Note that you must change the forests.dat file <u>before</u> loading the route and placing a forest.** If you want to change an already created forest, you have to change its properties in the world files, as described further below.

You may also **create separate forests with differently sized trees**. For that you add new lines to the forests.dat file, for example as follows (note the number 4, which must indicate the number of lines that follow):

```
SIMISA@@@@@@@@@@JINX0F0t______

4
Forest ( "JP1Tree1-default" "JP2AutoTree1.ace" 16.0f 20.0f 0.9f 1.1f )
Forest ( "JP1Tree1-wide" "JP2AutoTree1.ace" 24.0f 20.0f 0.8f 1.2f )
Forest ( "JP1Tree1-tall" "JP2AutoTree1.ace" 16.0f 30.0f 0.7f 1.3f )
Forest ( "JP1Tree1-varied size" "JP2AutoTree1.ace" 16.0f 20.0f 0.5f 1.5f )
```

It is also possible **to create a variety of forests with different types of tree**. In the last example, you could replace JP2AutoTree1.ace with the names of other trees (you will have to import other trees from default routes, including the seasonal variations of the *.ace files). You can look at the forests.dat files of other routes for inspiration, for example the following one from the EUROPE1 route (here the size range is always 0.9 to 1.1, but you can change that as you wish, as well as other values):

```
SIMISA@@@@@@@@@@JINX0F0t______

17
Forest ( "JP1Bush1" "JP2AutoBush1.ace" 3.50f 2.5f 0.9f 1.1f )
Forest ( "JP1Bush2" "JP2AutoBush2.ace" 4.5f 3.2f 0.9f 1.1f )
Forest ( "JP1Bush3" "JP2AutoBush3.ace" 4.6f 3.4f 0.9f 1.1f)
Forest ( "JP1Tree1" "JP2AutoTree1.ace" 16.0f 20.0f 0.9f 1.1f )
Forest ( "JP1Tree2" "JP2AutoTree2.ace" 14.0f 18.0f 0.9f 1.1f )
Forest ( "JP1Tree3" "JP2AutoTree3.ace" 10.0f 14.0f 0.9f 1.1f)
Forest ( "UKTree1" "tree8.ace" 20.0f 18.0f 0.9f 1.1f)
Forest ( "UKTree2" "tree4.ace" 19.0f 17.0f 0.9f 1.1f)
Forest ( "UKTree3" "treemd.ace" 16.0f 15.0f 0.9f 1.1f)
```

```
Forest ( "UKTree4" "tree5.ace" 18.0f 18.0f 0.9f 1.1f)
Forest ( "UKTree5" "tree11.ace" 16.0f 16.0f 0.9f 1.1f)
Forest ( "US1Decidtree1" "usdecidtree1.ace" 16.0f 18.0f 0.9f 1.1f)
Forest ( "US1Decidtree2" "usdecidtree2.ace" 16.0f 18.0f 0.9f 1.1f)
Forest ( "US1Decidtree3" "usdecidtree3.ace" 16.0f 18.0f 0.9f 1.1f)
Forest ( "US2Fir1" "US2autofir1.ace" 12.0f 22.0f 0.9f 1.1f)
Forest ( "US2Fir2" "US2autofir2.ace" 12.0f 23.0f 0.9f 1.1f)
Forest ( "UKTree7" "tree7.ace" 16.0f 15.0f 0.9f 1.1f)
```

You can also **vary trees in a forest that you have already placed previously**. You can change the forest properties in the world file of the tile where you put that forest. In the world file, the forest definition will look something like this by default (to find these lines in the world file, see section 3.10.9):

```
      Forest (
            UiD ( 374 )
            TreeTexture ( JP2AutoTree1.ace )
            ScaleRange ( 0.9 1.1 )
            Area ( 50 163.25 )
            Population ( 24 )
            TreeSize ( 16 20 )
            StaticFlags ( 00100000 )
            Position ( -588.977 1 -233.875 )
            QDirection ( 0 -0.0103107 0 0.999947 )
            VDbId ( 0 )
            StaticDetailLevel ( 0 )
      )
```

Here you can change the `TreeTexture` name and the `TreeSize` width and height (and perhaps other quantities as well).

As you see, each forest only contains one type of tree. **To mix different types of trees in a forest**, you can place two or more forests with different trees so that they overlap each other (but avoid placing forests exactly on top of each other to prevent trees from coinciding).

## V. Building "First Route": importing and using sounds

In this near-final stage of building your "First Route", you will learn to import and use sounds. This will add environmental sounds ("Sound Sources") as well as variable track noises ("Sound Regions") to your route. See section 3.13 for more details.

By default, a new route only includes two sounds selectable through the Object selector: one is called "Town - small - modern - Sound Sources", available in the Sound Sources Object class; and the other is called "Default" in the Sound Regions Object class. Unfortunately, neither of these is really useful. The first is too faint, while the second only duplicates the "default" track sounds that your route already has.

So you will need to **import sounds** from one or more default MSTS routes.

Since the First Route is located in the UK, let's choose British sounds. We will import all British sounds for simplicity (see section 3.13.1a if you want to only import individual sounds).

**To import these British sounds, do this:**

- exit from RE;
- copy ssource.dat and ttype.dat from the EUROPE1 main folder to the FirstRoute main folder;
- copy all the files from the EUROPE1\Sound folder to the FirstRoute\Sound folder.

Now you may restart RE to **place sound sources**, as follows (you will find these sounds in the Sound Sources Object class of the Object selector):

- select "City - larger - modern - Sound Sources" in the Sound Sources;
- press F5;
- point near the buffers of the Central London station, then left-click: a yellow pillar appears (an example is shown in the figure further below).

If you approach the camera to that yellow pillar, you may be able to hear its sound, but it is rather faint.

You could place a station sound source near the station buildings (it will also be faint), as follows:

- select "Station - medium - Sound Sources";
- press F5;
- point near the station buildings, and left-click: another yellow pillar appears.

Next add some road traffic noise near the level crossing just North of the Central London station, in the same way: select "Traffic - parked or at crossing - modern - Sound Sources" and place it close to the level crossing (off the tracks and off the road), and left-click. This sound is much stronger: you should have no difficulty hearing it.

If you have placed a road near your tracks, you could add moving traffic noise: select "Traffic - in motion - modern - Sound Sources", and place it near the road. This is also easily heard.

Next place some nature sounds on the hill North of Central London: select "Natural 1 - Medium (550m) - UK - Sound Sources", and place it near the tracks. You should be able to hear birds!

Another possibility is to place flowing water sounds near the river over which you built a bridge: select "Water - large - running (river) - Sound Sources", and place it near both ends of the bridge.

NOTE: Unfortunately, once you have placed a sound source, there is no way to tell which type you installed, except by listening to it (if you can hear it!). The yellow pillars are all identical, and selecting and right-clicking a pillar gives no distinguishing information. So if you are uncertain what type a sound source is, the only way to be sure seems to be to remove it and place the type that you want to be there.

So far you have installed "environmental" sounds: the train driver will normally not hear these (the cab windows are closed), but they are only heard in MSTS from outside the cab (by pressing 2, 3 ,…).

**Next you will install "track" sounds**: these are the wheel noises. You can make the wheel noises change as the train passes over a level crossing, through a tunnel, under a bridge, or over various kinds of bridges. They are heard in MSTS from inside the cab. However, they cannot be heard in RE.

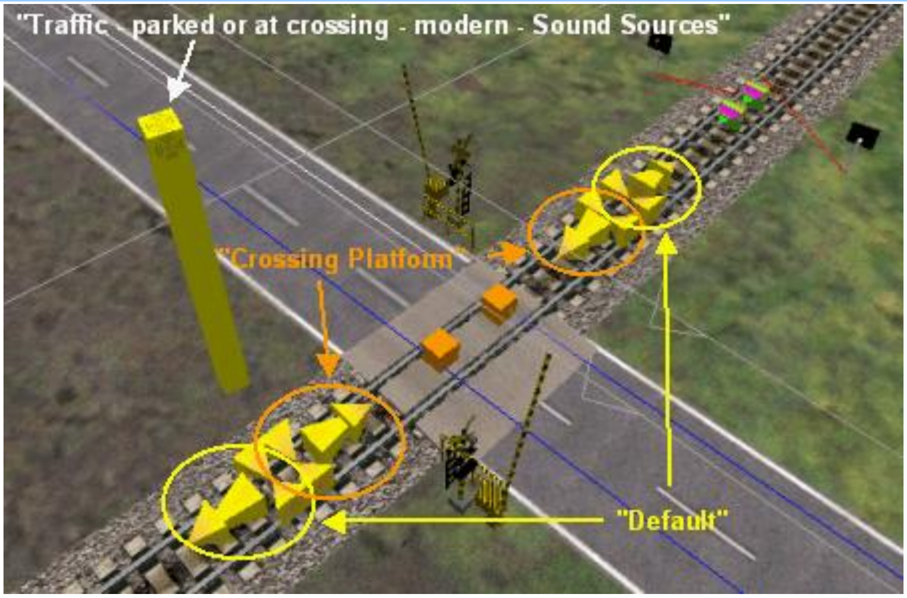
The wheel sounds are imported by copying the file ttype.dat, as you already did above: no other files need to be copied. You will find these sounds in the Sound Regions Object class of the Object selector.

Start by adding wheel sounds across the level crossing just North of the Central London station, as follows.

- select "Crossing Platform" in "Sound Regions";
- press F5;
- point between the rails on one side of the level crossing, and left-click: as shown outlined in orange in the next figure, you will get 4 yellow handles (2 of these point along the track and will actually appear as wire frame);
- make sure the two wire-frame handles point toward the level crossing, as shown in the figure: if they don't, press F3, move the camera to the side (over the road) and drag those two handles toward the level crossing;
- repeat this at the other side of the level crossing (also outlined in orange in the figure).

For MSTS to work properly, you now need to define what the wheel noises are away from the level crossing. For that we add two more sound regions of the "Default" type, as follows:

- select "Default" in "Sound Regions";
- press F5;
- point between the rails away from the level crossing, just beyond where you put the "Crossing platform" sound, as shown outlined in yellow in the next figure: you will again see 4 yellow handles appear;
- make sure the two wire-frame handles point away from the level crossing, as shown in the figure: if they don't, press F3, and drag those two handles away from the level crossing;
- repeat this at the other side of the level crossing.




**Save your work, after deselecting any selected track or objects!**

As you may have guessed, Sound Region objects operate in pairs of the same type (as in the example of the Crossing Platform objects shown in the figure above): as a train passes the first object of a pair, its particular track sound starts; and when the train passes the second object of that pair, this sound stops. The handles of a pair point at each other.

And when one Sound Region object stops a track sound, you need another Sound Region object to start a different track sound; and then one more Sound Region object is needed to stop that different sound further along the route. As you see, you will need to cover your entire route with Sound Regions.

Thus, your whole route can be broken up into different Sound Regions.

In particular, you need to stop the sounds near the ends of the route, by placing a suitable Sound Region object near each free track end. Also, you should leave no gaps between Sound Regions.

So now **you should complete the Sound Regions in your First Route**.

First, you can terminate the Default Sound Region that points back to the Central London station: place a Default Sound Region object near the buffers on one of the two tracks; it will automatically span the width of both tracks, so you don't have to worry about terminating the sound on both tracks separately.

Next, you can move North of the level crossing and place track-on-bridge sounds over the river. Since we installed a steel bridge, choose the Steel Bridge Sound Region object, and place one at each end of the bridge. And add a Default Sound Region object outside of each of those, just like the situation at the level crossing. This way, the Default track sound will be heard all the way from the level crossing to the bridge, and again after the bridge.

It is allowed to have one or more switches within a Sound Region. For example, between one Default sound object and the next - both pointing their handles at each other - you can install as many switches, spurs, sidings, etc., as you wish. It is also allowed to branch off onto another line: just make sure that, wherever the tracks go, each track sound is properly stopped before another starts.

North of the bridge, you can place an In Tunnel Sound Region within each of the two tunnels (again surrounded by Default Sound Region objects to cover the normal tracks). And you should terminate the last Default Sound Region at North London (you may terminate it just before the first switch of North London, instead of terminating it at every platform, yard track and wye end).

WARNING: Do not put a Sound Region object right on top of a switch. Otherwise, the sounds will be confused.

**Save your work, after deselecting any selected track or objects! It would also be good to back up your FirstRoute folder again. And you might as well reboot your PC after that.**

Next we will add electrification and gantries to the route. Then the route will be "finished"!

**To continue building our First Route, jump to the next blue box.**

## 3.13 Sounds

By default, when you run a train on a new route in MSTS, you can hear several types of sound: track sounds (due to the wheels) from inside the train; engine sounds from inside and outside the train; and various control sounds (such as clicking of the throttle, brake lever, and switches in the cab, and braking sounds).

**You can add two kinds of sounds to a route:**

- **Sound Sources, which are different "environmental sounds"** like city sounds, road traffic noise, natural sounds and industrial noise; these sounds are placed near your tracks and are heard in MSTS from outside the train; they are also heard in RE;

- **Sound Regions, which are various "track sounds"** due to the wheels rolling over the track on or under bridges, on a level crossing, and in tunnels; these sounds are placed on the tracks from one point to another (such as from one end of a bridge to another) and are heard from inside the train; they are not heard in RE.

[NEW SINCE V2] Sound sources are declared in the route's ssource.dat file, while Sound Regions are declared in the route's ttype.dat file.

By default, a new route only includes two sounds selectable through the Object selector: one is called "Town - small - modern - Sound Sources", available in the Sound Sources Object class; the other is called "Default" in the Sound Regions Object class. Neither is useful: the Sound Source available by default is too faint to be heard (but maybe it works for you!), while the "Default" Sound Region is the same as the track sounds that you hear anyway when running a train, so adding it changes nothing (we will need it later, however).

**To add other sounds, it is therefore necessary to import sound files from one or more default MSTS routes.**

Sounds need two types of files: a *.sms file and a *.wav file. Sound files of the *.wav type are quite large (a full set from a default route takes from 8 to 13 Mb), so you may want to select only those that you will use. But, as with importing object files, it is much easier to import the complete set of sounds from one default route, rather than try to figure out exactly which files you will need.

To complicate matters, sound files can be found in two different types of locations within the MSTS folders:

- sound files located in the Train Simulator's Sound folder; these sound files are "global", common to all routes: they need not be imported to a new route, but they must be declared in the new route's main folder within the ssource.dat or ttype.dat file;

- sound files located in an MSTS default route's Sound folder; these sound files are accessible only to that particular route: they must be imported to a new route, and they must also be declared in the new route's main folder within the ssource.dat or ttype.dat file.

## 3.13.1 IMPORTING SOUND SOURCES AND SOUND REGIONS

We will describe two cases: importing individual sounds, and importing all sounds from a default MSTS route.

NOTE: The Sound Region files are all "global", so you don't need to import them. But they are defined in the ttype.dat file, so you must copy the ttype.dat file from the desired default route. Although there are 6 default routes, there are only 3 different ttype.dat files, one for European routes, one for Japanese routes and one for American routes.

### 3.13.1a Importing individual sounds from a default MSTS route

First you need to **determine which files you must import**.

To be specific, let's assume that you want to use the Sound Source called "Natural 1 - Medium (550m) - UK" from the Settle and Carlisle route (EUROPE1).

This sound is listed in the ssource.dat file that is in the main EUROPE1 folder (a sound could also be listed in the file ttype.dat, so check both of these files):  in ssource.dat you can see that this sound uses the file gen_uk_nat1.sms.  If you now look in the EUROPE1\Sound folder you will see this file gen_uk_nat1.sms, as well as the file gen_uk_nat1.wav:  these are the files you need.

If you can't find the desired file in the route's Sound folder, it must be in the MSTS Sound folder: in that case, the sound is global, and you do not need to import its files, but you do need to import the ssource.dat file, and delete unused sounds from it, as discussed next.

You also see a file gen_uk_nat1_s.sms in the EUROPE1\Sound folder:  it corresponds to the sound called "Natural 1 - Small (300m) - UK" (as ssource.dat tells you), but it also uses the same file gen_uk_nat1.wav.  So you might as well also import the small file gen_uk_nat1_s.sms and get a second version of this sound with little cost in disk space.

**To import this pair of sounds from the default EUROPE1 route, do the following:**

- exit from RE;
- copy ssource.dat and ttype.dat from the EUROPE1 main folder to the NewRoute main folder;
- copy the files gen_uk_nat1.sms, gen_uk_nat1_s.sms and gen_uk_nat1.wav from the EUROPE1\Sound folder to the NewRoute\Sound folder;
- in ssource.dat, delete all the lines that correspond to the UK sounds that you do not want to use and did not import:  the UK sounds are those whose names end in "UK";  don't delete the other non-UK sounds from ssource.dat, such as those whose names start with "Town", "City", etc.

By this process, you have at the same time "imported" many more sounds that are global:  now all the sounds listed in the files ssource.dat and ttype.dat are available in your new route.

### 3.13.1b Importing all sounds from a default MSTS route

**To import all sounds from a default MSTS route, do the following:**

- exit from RE;
- copy ssource.dat and ttype.dat from the default route's main folder to the NewRoute main folder;
- copy all the files from the default route's Sound folder to the NewRoute\Sound folder.

By this process, you have at the same time "imported" many more sounds that are global:  now all the sounds listed in the files ssource.dat and ttype.dat are available in your new route.

### 3.13.2 PLACING SOUND SOURCES

Sound sources are placed near the tracks (not on the tracks): remember that they give environmental sounds.

**You place a sound source like any object**: select it in the Sound Sources Object class of the Object selector; press F5; point where you want it placed, then left-click: a yellow pillar appears.

If you approach the camera to that yellow pillar, you should be able to hear its sound. But note that such sounds can be rather faint, so they may be difficult to hear! Louder sounds include the traffic and industrial sounds.

NOTE: One difficulty with sound sources is that they all show up in RE as yellow pillars that are indistinguishable. Worse: if you select such a pillar and press F6 (or right-click) to get information about it, you will also not be able to find out which sound type it represents. With loud sounds, you can often tell by listening what kind of sound it is, but with faint sounds that may be impossible! So if you want to make sure that a given pillar represents a desired sound, the best way is to delete it and place the sound type that you want.

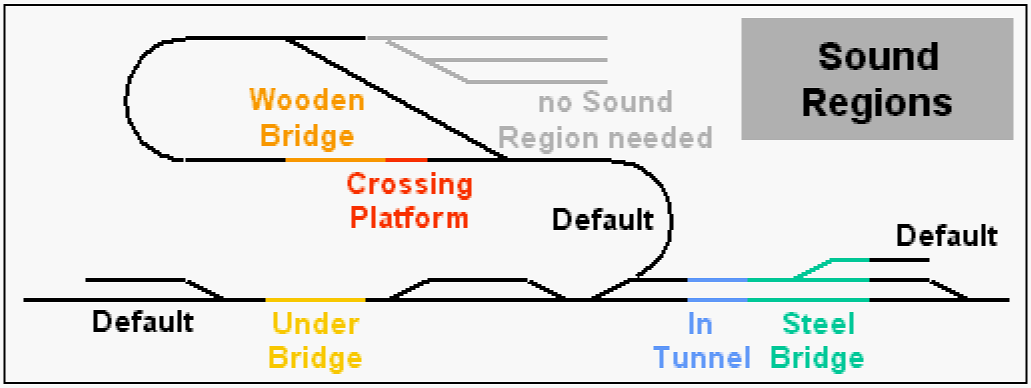
### 3.13.3 PLACING SOUND REGIONS

Sound regions are wheel noises (and its echoes, etc.) that a train makes as it passes over certain stretches of track (such as in tunnels, over or under bridges, etc). You can define exactly where those stretches are: for example, you can define where a tunnel-like track sound will start and where it will stop, by placing two markers at those locations: these markers are the Sound Region objects that we discuss here. To see what they look like, look at the illustration in the Building "First Route" project section dealing with importing and using sounds.

**To place Sound Region objects, do the following:**

- select the desired Sound Region type in "Sound Regions";
- press F5;
- point between the rails where you want that sound to start or stop, and left-click: you will get an object with 4 yellow handles (2 of these point along the track and will actually appear as wire frame);
- make sure the two wire-frame handles point toward the stretch where you want that sound to be heard: if they don't, press F3, move the camera to the side of the track and drag those two handles toward that stretch;
- repeat this to place an identical Sound Region object at the other end of the stretch where you want that sound to be heard (there may be more than one other end, if the track splits up): its handles should be pointing toward the first Sound Region object that you placed.

It is recommended to **assign a particular Sound Region to every part of your route**. In other words, every piece of your route will be in one or another Sound Region. One Sound Region will thus start where another Sound Region stops, as in the example of the next figure, where Sound Regions are color-coded (the "Crossing Platform" sound is used at level crossings). Two or more identical Sound Region objects terminate each Sound Region.




The parts where you don't want special sounds are labeled "Default" and colored black in the above figure: you should place "Default Sound Regions" to cover those stretches, so they are treated the same way as stretches with special sounds. For example, at bottom left is a track that branches into two tracks, labeled "Default": it will require 3 "Default Sound Region" objects, one at each end of the black-colored tracks.

Whenever you use dual- or multiple-tracks (for example in the "In Tunnel" and "Steel Bridge" regions), one Sound Region object will automatically span all the parallel tracks. So two "In Tunnel Sound Region" objects will take care of the tunnel sounds, and two "Steel Bridge Sound Region" objects will produce the desired steel-bridge sounds, including on the spur. And the large complicated "Default Sound Region" in the center of the figure will require 5 "Default Sound Region" objects.

NOTE: It appears that you can simplify those "Default Sound Regions" that are at the ends of a route, as shown in gray in the figure. There we put a "Default Sound Region" object before the switch leading into the three yard tracks, terminating the Default sound there: although there are no Sound Region objects in that yard, the Default sound will still be heard throughout the gray region.

WARNING: Do not put a Sound Region object right on top of a switch. Otherwise, the sounds will be confused.

## W. Building "First Route": Electrification

You will next electrify First Route: the route will be electrified in its entirety (you cannot leave parts non-electrified). This involves **two main steps**: the first step is simple and declares the route to be electrified (which allows electric trains to run and causes overhead cables to show); the second is more complex and adds gantries to support the overhead cables. See section 3.14 for more detail.

## Declare electrification

This simple first step can be done in three ways.

[NEW SINCE V1.106] The first way is to load the route in the RE and enable electrification in the Route Properties, as follows:
- open RE and load First Route;
- select Route | Properties;
- under Electrification, click to enable Overhead wires (Third rail apparently does not work);
- set the Wire height to 5.4 (meters);
- adjust the Voltage if desired.

The second way is to load the route in the RGE and enable electrification there, as follows:
- open RGE and load First Route;
- select Edit | Route values;
- enable Electrified track;
- set the Electrified cable height to 5.4 meters;
- exit from RGE.

The third way is to edit (with WordPad) the route's *.trk file:
- replace the lines

```
Electrified ( 00000000 )
OverheadWireHeight ( 0 )
```

with the lines

```
Electrified ( 00000001 )
OverheadWireHeight ( 5.4 )
MaxLineVoltage ( 25000 )
```

(so you only change `00000000` to `00000001` and `0` to `5.4` and add the Voltage line).

Electrification applies to your entire route: you cannot declare part of the route to remain non-electrified. This means that you will be able to use electric trains everywhere on the route, but also that you will see electric cables over all tracks. These electric cables will hang in the air without support, an unrealistic sight (but MSTS will work this way), unless you place gantries to support them.

In RE you will only see those pieces of overhead cable that hang over switches.

You can later easily delete electrification of the route, similarly to the ways described above. But note that any gantries that you place will stay there (without overhead cables), unless you delete them one by one from the entire route.

## Place gantries

Gantries are only needed for their visual appearance: they are not needed to run electric trains, and they need not touch the electric overhead cables. They can be handled as normal objects.

The RE provides a tool for the auto-placement of gantries along the route: the file gantry.dat can be prepared to place gantries automatically at fixed intervals along all tracks, with proper orientation and often proper height. This saves a lot of hard labor. However, you will probably still have to fine-tune some of the automatically placed gantries manually one by one (especially in yards).

The **gantry.dat file** allows you to specify different kinds of gantry depending on the width of your single and multiple tracks: you can arrange for single-pylon gantries to be placed on one side of single-track sections, while two-pylon gantries of different widths can be automatically placed across dual, triple or quadruple track sections. That is what we will do here, by preparing the following gantry.dat file (you can get this file from my accompanying First Route, so you don't have to type it in):

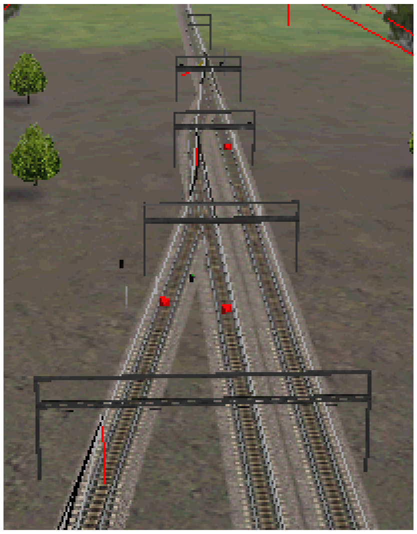
```
SIMISA@@@@@@@@@@JINX0g0t______

Tr_GantryFile (
  GantrySets ( 1
    GantrySet(
      Name ( "First Route" )
      Style ( 00000002 )
      Separation ( 40 )
      GantryTable ( 4
        GantryTableEntry (
          Filename ( "JP1gantry1tall.s" )
          Distance ( 5.000 )
        )
        GantryTableEntry (
          Filename ( "JP1gantry2.s" )
          Distance ( 11.373 )
        )
        GantryTableEntry (
          Filename ( "JP1gantry3tall.s" )
          Distance ( 17.000 )
        )
        GantryTableEntry (
          Filename ( "JP1gantry4lan.s" )
          Distance ( 30.000 )
        )
      )
    )
  )
)
```

Here the Name is the route name, and Separation gives the interval between gantries (40m in this case). The number after GantryTable (4 here) specifies how many gantry types will be listed and used: each is specified as a GantryTableEntry, together with a Filename and a Distance (to be explained in section 3.14).

The gantry.dat file lists four types of gantry that I selected from the JAPAN1 route: only one of these (the two-track gantry JP1gantry2.s) is available by default in a new route. The other three gantries need to be imported from the JAPAN1 route, as follows (unfortunately, only "tall" versions of JP1gantry1.s and JP1gantry3.s are available):
- copy JP1gantry1tall.s and JP1gantry1tall.sd from JAPAN1\SHAPES to FirstRoute\SHAPES;
- copy JP1gantry3tall.s and JP1gantry3tall.sd from JAPAN1\SHAPES to FirstRoute\SHAPES;
- copy JP1gantry4lanl.s and JP1gantry4lan.sd from JAPAN1\SHAPES to FirstRoute\SHAPES.

(These all require JP1gantry1.ace, which is already available by default in a new route.)

The next figure shows the four types of gantries placed at the entrance to the North London station and yard.



If you want to be able to add individual gantries yourself with the normal RE Placement tool (I recommend this especially for yards), you should define the gantries in the *.ref file of the route, by adding the lines:

```
Static (
        Class           ( TrackObjects )
        Filename        ( JP1gantry1tall.s )
        Align           ( None )
        Description     ( JP1_Gantry1_Tall )
)
Static (
        Class           ( TrackObjects )
        Filename        ( JP1gantry2.s )
        Align           ( None )
        Description     ( JP1_Gantry2 )
)
Static (
        Class           ( TrackObjects )
        Filename        ( JP1gantry3tall.s )
        Align           ( None )
```

```
        Description     ( JP1_Gantry3_Tall )
)
Static (
        Class           ( TrackObjects )
        Filename        ( JP1gantry4lan.s )
        Align           ( None )
        Description     ( JP1_Gantry4_Lan )
)
```

Once the gantry.dat file is ready and placed in the main folder of the route, you can start RE, load the route and **start placing gantries, as follows**:
- move the camera to a tile that has tracks (the auto-placement will only occur within the current tile);
- select Auto-placement | Add gantries;
- gantries should show up along the tracks in the current tile (including "tall" ones!);
- you can delete excessive gantries: for example fewer gantries are needed along straight tracks than in curves, so you might delete every other gantry along straight tracks;
- adjust the remaining gantries: the "tall" gantries have to be sunk into the ground (as described next), and the single-pylon gantries may be moved closer to the tracks (they can be moved like any objects);
- move to the next tile that needs gantries and repeat the process of auto-placing and then fine-tuning gantries.

Remember to save frequently!

**Set the height of gantries**

The 2- and 4-track Japanese gantries are automatically placed to fit the Japanese cable height (5.4 m). So, if you set the cable height to 5.4 m, those gantries will have the correct height. For a higher cable height (as in the USA or Europe), these gantries are too low. For a lower cable height, you will have to lower the gantries one by one (see next).

The single-pylon and 3-track Japanese gantries are "tall" (perhaps to be used on bridges), and have to be lowered to fit the cable height.

Adjusting the height of "tall" gantries can be labor-intensive, especially along sloping track. For accuracy, you need to fix each gantry's numerical altitude (y value), as described below.

If you prefer, you can try to adjust the gantry heights visually rather than through calculation, especially along sloped tracks: that will be faster, although less accurate. The white line above the tracks gives a visual guide to set the gantry height.

For horizontal track, the easiest exact way to set the height of a "tall" gantry is to place a "tall" gantry right next to a 2- or 4-track gantry that already has the proper height. Then lower the tall gantry until it visually fits the properly placed gantry, and record the tall gantry's height (y value in its Properties, seen by pressing F6 or right-clicking). Now copy that y value (with Ctrl-C) and apply it to all similar gantries that stand on the same terrain level (with Ctrl-V change the y value in their Properties).

For sloping track, you could use the following exact approach (a slightly simpler approach is described in section 3.14.4, which works best on level ground). When you perform the above gantry height fitting, also record the local terrain height where you made the fit (that is 1 m by default; otherwise press F9, point nearby, right-click, and read the content of Set height). Now subtract that terrain height from the y value to get the gantry's "depth below ground" (it will be negative): for example, the JP1gantry1tall.s placed on default terrain fits well if lowered to y = -28.74m; therefore, its best "depth below ground" is -28.74 - 1 = -29.74m. Now go to a gantry that needs to be lowered, measure the terrain height near it (as just described), and add to it the negative "depth below ground" found earlier (adding a negative number is the same as subtracting the positive value): that gives you the proper y value for this gantry. For example, if the terrain is at 2.96m, you should lower a JP1gantry1tall.s to y = 2.96 + (-29.74) = 2.96 - 29.74 = -26.78m.

Likewise, I found that JP1gantry3tall.s fits best when its "depth below ground" is -30.105m.

Save and back up the route.

**Now you could produce one or more simple Activities** for yourself, or for another user: see section 5.

You could also prepare to distribute your route to other users (but First Route is not the ideal choice for that!): see sections 6 and 7.

**The First Route is "finished"!**

However, no route is truly ever finished: you can continue adding to it forever... To illustrate some of the possibilities, my version of First Route includes the following further additions (which are explained in more details in other sections):
- "pickup objects", like water columns, coal towers and diesel pumps (section 3.12.1);
- static locomotives and railway cars (section 3.12.2);
- more cars for the car spawners (section 3.12.3);
- static road vehicles (sections 3.12.4);
- people and static deer (section 3.12.5);
- "environments", namely variable fog, skies, water, rain, snow and wind (see section 3.15).

Beyond that, you can add a myriad of other objects: miscellaneous trackside equipment, houses, buildings (after all, First Route is supposed to be in London!), walls, fences, cows, graveyards, radio towers, piles of rubble, etc. Just look at the default MSTS routes, or outside your window, for inspiration. In fact, adding such objects can take much more time than what you have done so far! You can also learn to build your own add-on objects, spending even more time!

## 3.14 Electrification

**A route can be electrified at any time.** I recommend that it be done after all tracks and all track-related objects have been placed, because gantries are objects.

Electrification permits the selection of electric locomotives in MSTS, and makes MSTS display the overhead electric cable (catenary).

**The route will be electrified in its entirety** (you cannot leave parts non-electrified). Thus, you will be able to use electric trains everywhere on the route, and you will see electric cables over all tracks.

MSTS does not require that the cables be supported, but unsupported cables are obviously unrealistic: it is thus desirable that you place gantries to support the cables.

Electrification therefore involves **two main steps**: first, **declare the route to be electrified**; second, **add gantries** to support the overhead cables.

### 3.14.1 DECLARING ELECTRIFICATION

When enabling electrification on a route, you also need to **specify the height of the electric cables**, because that varies across the world: for example, it is generally 7.2 m in the USA, something like 6.0 m in Western Europe, and 5.4 m in Japan.

This simple first step can be done in **three ways**.

[NEW SINCE V1.106] The **first way is to load the route in the RE**, select Route | Properties, then enable electrification, as well as specify the height of the electric cables, and the voltage (if different from the default value).

The **second way is to load the route in the RGE** and enable electrification, as well as specify the height of the electric cables. To declare electrification, **open RGE, load your NewRoute, select Edit, then Route values: now enable Electrified track, and set the Electrified cable height; exit from RGE.**

**The third way is to edit (with WordPad) the route's *.trk file**: replace `Electrified ( 00000000 )` by `Electrified ( 00000001 )`, and `OverheadWireHeight ( 0 )` by `OverheadWireHeight ( 5.4 )` (for 5.4m). Also add a line saying `MaxLineVoltage ( 25000 )` (for 25000 Volts) after that.

After this first electrification step, you will in RE only see those pieces of overhead cable that hang over switches. In MSTS the entire overhead cable will be visible, over all tracks. You cannot have overhead cable on one part of the route, and not on another part.

You can later easily delete electrification of the route, reversing the two ways described above. But note that any gantries that you place will stay there (without overhead cables), unless you delete them one by one from the entire route.

### 3.14.2 PLACING GANTRIES

**Gantries are only needed for their visual appearance**: they are not needed to run electric trains, and they need not touch the electric overhead cables. They can be handled as normal objects.

The default MSTS routes provide two sets of gantries: one for the Northeast Corridor route (USA1), and one for the Tokyo-Hakone route (JAPAN1). The USA1 set covers 2-, 3- and 4-track widths, and includes tall poles for telephone wires and other high overhead cables: this rather complete set of gantries is somewhat unique in style and would not fit European or Japanese routes.

The JAPAN1 set of gantries covers widths from 1 to 4 tracks and would also fit European routes. However, the 1- and 3-track width gantries have amazingly tall pylons and are by default positioned far too high: those must be lowered to a reasonable height one at a time, a very tedious and laborious job (described in section 3.14.4). For long single-track routes, especially on sloping terrain, this is far too time-consuming.

For general use, it is more practical to use add-on gantries downloaded from the web. A nice free set that covers all widths from 1 to 6 tracks is the Swedish gantry set (available at http://msts.e-buzz.net/ - look for the "Swedish Objects 1" by Mats Strid). This set fits European routes well and is easy to install in any route.

The RE provides a **tool for the auto-placement of gantries along the route: the file gantry.dat** can be prepared to place gantries automatically at fixed intervals along all tracks, with proper orientation and often proper height. This saves a lot of hard labor, but still requires fine-tuning some of the automatically placed gantries to adjust their height or their distance from the tracks: this is done manually one gantry at a time (especially in yards, because the auto-placement works poorly there).

**To place gantries automatically** (after you have created a proper gantry.dat file, as discussed below), do the following:

- **move the camera into a tile where you want to place gantries;**
- **select Auto-placement | Add gantries**: this places gantries in the current tile only.

To add gantries in another tile, move the camera there, and repeat the same operation (but it is recommended that you first fine-tune all gantries in the previous tile, save your work, etc.).

You may **delete all the RECENTLY auto-placed gantries** in one step for the current tile, as follows (this does not work for gantries placed earlier):

- select Auto-placement | Remove gantries.

### 3.14.3 THE GANTRY.DAT FILE

**The gantry.dat file controls the auto-placement of gantries.** You can **adjust gantry.dat to your needs**.

In particular, gantry.dat allows you to specify different kinds of gantry depending on the width of your single and multiple tracks: see section W for an example (the number following GantryTable specifies how many gantries you want to list in the file). You need to identify which types of gantry you want, and import them into your route like any other objects (including *.ace files as well as their Snow and other versions, if present).

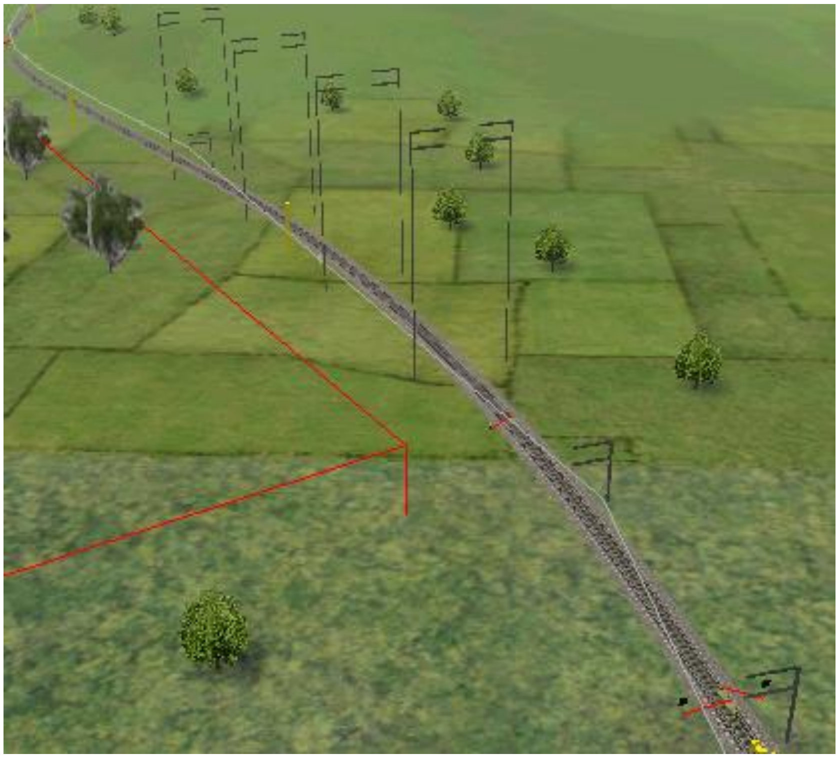
**If you add the imported gantries to the route's *.ref file, you can also place them individually with the RE Placement tool**: this will be especially convenient in yards. (Note that copying and pasting unfortunately does not work for some gantries.)

The way the automatic gantry selection works is that the first listed gantry is chosen where the track width is smaller than the Distance ("gantry width limit") that you specified for this gantry. Where the track width is larger than this limit, the second listed gantry is chosen, assuming that the track width is smaller than the Distance ("gantry width limit") that you specified for this second gantry. And so on for the wider tracks and gantries. So you should list gantries in the gantry.dat file in order of increasing width.

It is not clear what RE takes as the track width, so a trial-and-error approach is best: set your gantry width limits (you can use my values shown in section W, or contained in the file gantry1234.dat mentioned below); place gantries (select Auto-placement | Add gantries) in a tile that has a variety of track widths to check whether the result is as desired; then delete all those gantries (select Auto-placement | remove gantries); change your gantry width limits in the gantry.dat file (you don't need to exit from RE); and try again. The freeware Swedish gantry set (mentioned above) comes with its own gantry.dat file that specifies track widths of 5, 10, 15, 20, 25 and 30 m for 1-, 2-, 3-, 4-, 5- and 6-track lines: you can try these values on your route.

You may change the gantry.dat file when you move to another tile, so as to create other sets of gantries there.

Another option in the gantry.dat file is to have pairs of gantries placed opposite each other across the tracks: this is specified as `Style ( 00000002 )` in the gantry.dat file. This can be useful when placing single-pylon gantries along single-track lines, in which case you may want to have the pylons standing always on the OUTSIDE of curves (on the left of the track in a right curve, and on the right of the track in a left curve). You can then simply delete those gantries that you don't want, instead of painfully moving a gantry across the track and turning it around. This is shown in the next figure (here the two nearest gantries have been adjusted, while the more distant "tall" pairs must still be adjusted).



Unfortunately, if you apply this method to two-pylon gantries for multi-track lines, you will get useless off-centered pairs of gantries. So my recommendation, if you want outside gantries in turns, is that you first place pairs of single-track gantries (limited to narrow track widths), and delete all those you don't want, then change the gantry.dat file to place non-paired multi-track gantries, again deleting those that you don't want: this uses `Style ( 00000001 )` in the gantry.dat file.

I have provided with the First Route several corresponding gantry.dat files: gantry1.dat only places pairs of single-track gantries; gantry234.dat places only non-paired 2-, 3- and 4-track gantries; and gantry1234.dat places all four gantries, non-paired (the contents of gantry1234.dat are listed in section W). To use any of these files in RE, copy it to gantry.dat.

### 3.14.4 SETTING THE HEIGHT OF GANTRIES

Some gantries (including the 2-track Japanese gantry provided by default in a new route) are automatically placed to fit the region's cable height (5.4 m for Japan). So, if you set the correct cable height in the Electrification step, those gantries will be placed at the correct height. But there are situations where you will have to adjust gantry heights to fit the cable height, and we discuss that now.

You can **try to adjust the gantry heights visually**: that will be approximate, but relatively fast (remember that you don't need to make the gantry touch the cable). The white line floating above the tracks is a visual guide to set the gantry height: the best is to first place one gantry accurately (with a method described below), and then to visually compare its height with the white line floating above the tracks; then adjust all the other gantries visually so they have a comparable height compared to that white line.

Adjusting the height of gantries accurately can be labor-intensive, especially along sloping track: you need to fix each gantry's numerical altitude (y value relative to sea level), as described next.

For horizontal track, the easiest exact way to set the height of a gantry is to place it right next to a gantry that already has the proper height, or next to an overhead cable over a switch. Then lower the gantry until it visually fits the properly placed gantry or the cable, and record the adjusted gantry's height (y value in its Properties). Now copy that y value (with Ctrl-C) and apply it to all similar gantries that stand on the same terrain level (with Ctrl-V change the y value in their Properties).

For sloping track, you could use the following exact approach. When you perform the above gantry height fitting, also record the gantry's "Altitude" (which is measured relative to the local ground level). Now go to a gantry that needs adjusting, and raise or lower it until its "Altitude" is equal to that recorded value: if the local terrain height fits the track level exactly, your gantry will have exactly the proper height; if the terrain does not fit the track, you may correct the gantry height by guessing or by visual comparison as described above.

Another approach, even more labor-intensive, is to record the local terrain height where you made the fit (that is 1 m by default; otherwise press F9, point nearby, right-click, and read the content of Set height). Next subtract that terrain height from the gantry's y value to get the gantry's "depth below ground" (it will be negative): see section W for an example. Now go to a gantry that needs to be adjusted, measure the terrain height near it (as just described), and add to it the negative "depth below ground" found earlier (adding a negative number is the same as subtracting the positive value): that gives you the proper y value for this gantry.

A time-saving approach is to adjust several gantries together: select them as you would a group of objects, then move them together. This works best on level terrain and along straight tracks, but can be applied in other situations also. (Note that the Altitude for a group of gantries on a slope does not indicate the Altitude of any of the selected gantries, but perhaps some average value.)

[NEW SINCE V2]
## 3.14.5 DE-ELECTRIFICATION AND DELETING GANTRIES

You can **cancel a route's electrification** by reversing any of the three methods of electrification given in Section 3.14.1.

In Explore mode, MSTS will then only offer the use of non-electric locomotives. (However, activities that used electric locomotives before de-electrification will still work with those electric locomotives after de-electrification, even though the overhead cable is no longer present.)

NOTE: The user of MSTS can prevent the display of the electric cable in Options - Advanced Display, even when the route is electrified (but this does not remove the gantries, and makes the cable invisible in all routes).

De-electrification does not automatically remove the gantries. You must do that separately. Here are three ways of **deleting gantries**:

1) Slowest method: Load the route into the RE, move along the tracks, select each gantry in turn, and delete it; save the route frequently.

2) Faster, but more dangerous method: Open each *.w file in the route's WORLD folder in turn (with WordPad), and delete each Gantry section, such as this one:

```
Gantry (
      UiD ( 343 )
      FileName ( JP1gantry2.s )
      Position ( -522.048 2.95939 -992.82 )
      QDirection ( 0 -0.0990618 0 0.995081 )
      VDbId ( 4294967295 )
      StaticDetailLevel ( 0 )
)
```

(be very careful to also remove the closing parenthesis on the last line!).

3) Quick and dirty method: Delete all the gantry shape files (*.s and *.sd, such as `JP1gantry2.s` and `JP1gantry2.sd`) in the route's SHAPES folder; then load the route into the RE; this will cause many error messages (one for each gantry in the current tile); answer Yes to each error message (by pressing Enter continuously); save the route after each string of error messages; then move along the track until you get a new string of error messages for the next tile; repeat the process, saving the route at each stage. Make sure that you visit the entire route, so all gantries will be removed.

After removing gantries, you should remove the entries for gantries in the route's *.ref file, if present. You may also clean up (if you wish) by removing all gantry shape files from the route's SHAPES folder, as well as their textures from the route's TEXTURES folder.

[NEW SINCE V1.106]

## 3.15 Environment

**The environment in MSTS includes the appearance of the fog, sky, water, precipitation (rain, snow), and wind.**

The fog, sky and precipitation are what varies most in the default MSTS routes. By contrast, the water looks the same in all default routes, in all seasons, and in all weather conditions. And the wind plays a very secondary role.

This section will allow you to change many of these aspects of MSTS. Especially the sky and water allow much flexibility and creativity. Because of the many variable factors involved and their frequent interconnections, this is a relatively complex subject. Also, this subject is relatively unexplored, with many uncertainties, and many other discoveries to be made in the way MSTS handles the environment.

In my version of First Route (accompanying this Guide), you can see 12 different environments, prepared according to the methods described in this section: just choose different combinations of season and weather.

The freeware add-on Sky! Conductor (available from http://www.howintheworld.com/downloads.shtml) modifies a few aspects of the weather of a route. It mainly allows the user to change the sky appearance, by providing more varied and realistic sky images. Also, Sky! Conductor changes the density of fog, rainfall or snowfall. However, you can make many more environmental changes than Sky! Conductor offers.

**IMPORTANT: The freeware add-on Sky! Conductor is incompatible with the environmental changes that are discussed in this section!** The reason is that Sky! Conductor modifies the same files that you will modify to change the environment. See section 3.15.7 for a method to get around this incompatibility.

**IMPORTANT: Changing the environment involves changing the contents of files: make sure that you use a Unicode-capable word processo**r like Word or WordPad (older versions of WordPad may not be Unicode-capable). **Also, you may need to create or change graphics files**: this is discussed in Appendix H.

CREDITS: Ron Spalding appears to have discovered a number of the techniques described below many months before I worked them out myself: he published some of them as a new water system for Rich Garber's Ohio RR and worked on the water system for Sea View 2 with Bill Burnett (using his Active World Environments! or AWE! approach), which saved me some re-discovery time. I also thank Jeff Bush for very useful discussions on the MSTS environment.

[NEW SINCE V1.106]

## 3.15.1 MANAGING THE ENVIRONMENT

**IMPORTANT: To see environmental changes in the Route Editor, you must change the route's editor.env file** (not the sun.env, rain.env or snow.env files, which are used only by MSTS). If you want **to see the same environmental effects in RE that you see in MSTS**, simply **copy the *.env file used in MSTS** (for instance rain.env) **over the editor.env file** (after backing up the original editor.env file). All these *.env files are contained in your route's ENVFILES folder. **Conversely, you may copy the editor.env file over any of the other *.env files to see the same environment in MSTS that you see in RE.**

You must **understand how the environment is controlled in MSTS**.

If you open a route's *.trk file, you will a section which looks like this by default:

```
Environment (
  SpringClear ( sun.env )
  SpringRain ( rain.env )
  SpringSnow ( snow.env )
  SummerClear ( sun.env )
  SummerRain ( rain.env )
  SummerSnow ( snow.env )
  AutumnClear ( sun.env )
  AutumnRain ( rain.env )
  AutumnSnow ( snow.env )
  WinterClear ( sun.env )
  WinterRain ( rain.env )
  WinterSnow ( snow.env )
)
```

**This sets the environmental conditions** (fog, sky, water, precipitation, wind) **for every combination of season** (spring, summer, autumn, winter) **and weather** (clear, rain, snow): there are 12 such combinations.

The environmental conditions are defined by default in the files sun.env, rain.env and snow.env (these are contained in the route's ENVFILES folder). As you see, the same three *.env files are used by default in all seasons: in other words, there is no seasonal dependence of the environment. So, **by default in MSTS, the user can only choose among three "environments": sunny, rainy, and snowy, as given by the 3 files sun.env, rain.env and snow.env**. The seasonal dependence that you do see by default in MSTS comes mostly from the seasonal change of the terrain and object textures (such as snow on the ground in winter, and brown leaves in autumn).

The setup of the route's *.trk file, with its 12 Environment lines shown above, potentially allows 12 different season/weather combinations of fog, sky, water, precipitation and wind: this permits much variety. And this does not even count the additional variations due to time of day: sunrise, daylight, sunset, night. Those variations come on top of the 12 environmental combinations; some of these time-of-day effects can also be controlled by the environmental settings, such as variations in the tint of fog, sky and terrain between sunrise, noon and sunset, or variations in the times of sunrise, sunset, moonrise and moonset. Remember that you change the time of day in RE by pressing + or -, thereby moving the sun and moon across the sky (RE does not tell you the clock time, however!).

**It is thus possible to define 12 different kinds of environment in a route.** The user of the route can then easily choose any of these 12 environments by selecting one of the 4 seasons and one of the 3 weather types before starting the route. (In addition, the user can also choose a time of day.)

To allow seasonal variations, you might change the relevant lines in the route's *.trk file to, for example:

```
Environment (
  SpringClear ( springclear.env )
  SpringRain ( springrain.env )
  SpringSnow ( springsnow.env )
  SummerClear ( summerclear.env )
  SummerRain ( summerrain.env )
  SummerSnow ( summersnow.env )
  AutumnClear (autumnclear.env )
  AutumnRain ( autumnrain.env )
  AutumnSnow ( autumnsnow.env )
  WinterClear ( winterclear.env )
  WinterRain ( winterrain.env )
  WinterSnow ( wintersnow.env )
)
```

If you do this, your route would need 12 (instead of 3) separate files of type *.env, defining the 12 kinds of environment. You would have to prepare those 12 *.env files, and place them in the route's ENVFILES folder. The rest of this section will discuss how to do that.

Of course, you don't have to make 12 separate kinds of environment and *.env files: the point is that you can create a maximum of 12 kinds with 12 different *.env files. In fact you might want the opposite extreme: just one type of environment, which you can achieve by defining only one file of type *.env, say average.env, and using only that file in the route's *.trk file.

**The Route Editor does not use the sun.env, rain.env or snow.env files: it uses the editor.env file.** Note that the editor always uses "summer" textures for terrain and objects, so that even if you generate snowy weather through the editor.env, you will still see summer-like terrain and objects.

NOTE: Sky! Conductor replaces the sun.env, rain.env and snow.env files of a route: this destroys your environmental modifications that are within those files. If you use alternate file names, such as springrain.env, etc., Sky! Conductor will have no effect on your route's environment. See section 3.15.7 for a method to get around this incompatibility.

The *.env files use texture files found in the ENVFILES\TEXTURES folder: these **texture files, of type *.ace, contain all the required graphics to draw the sun and moon, the sky layers, the water layers, and the raindrops or snowflakes**.

## 3.15.2 SKY

### 3.15.2a Quick how to

This section will give short answers to common questions, with few explanations. To better understand those answers and to produce other effects, you will need to read the subsequent sections.

The changes indicated in the answers are to be made inside the appropriate *.env files, unless noted otherwise.

## HOW DO I MAKE THE SKY SEASON-DEPENDENT?

- create additional files of type *.env for different seasons (see 3.15.1);
- adjust the sky properties in each of those *.env files;
- refer to the new *.env files in the route's *.trk file.

## HOW DO I CREATE FOG?

To create "ground fog" (which only obscures terrain and objects, but not the sky), do this:

- in a *.env file, add the lines

```
world_fog_distance ( 450 )
world_fog_day_colour ( ffffffff )
world_fog_night_colour ( ff000000 )
```

after the lines

```
SIMISA@@@@@@@@@@JINX0w0t______

world
(
```

- then adjust the value (in meters) of the `fog_distance`.

To create "sky fog" (which only obscures the sky, but not terrain and objects), do this:

- create a new sky layer whose color matches that of the ground fog, as follows:
- create an opaque white texture, called something like fog.ace;
- add a new sky layer section in the *.env file (as discussed further below);
- adjust the colors of the sun to be

```
world_sky_satellite_dir_rise_colour ( ff000000 )
world_sky_satellite_dir_high_colour ( ffffffff )
world_sky_satellite_dir_set_colour ( ff000000 )
```

(so the sun does not turn the foggy sky reddish while the ground fog remains gray).

(See Appendix I for the meaning of the color codes.)

## HOW DO I CHANGE THE FOG DENSITY?

- adjust the value (in meters) in `world_fog_distance ( 450 )`.

## HOW DO I CHANGE THE FOG COLOR?

- adjust the fog's day and night colors in

```
world_fog_day_colour ( ffffffff )
world_fog_night_colour ( ff000000 )
```

(See Appendix I for the meaning of the color codes.)

## HOW DO I ADD A SKY LAYER?

- increase by 1 (for example from 3 to 4) the value in the following lines

```
world_sky_layers
( 3
```

- add a new section like

```
world_sky_layer
(
  world_sky_layer_top
  (
    world_sky_layer_top_nfaces ( 8 )
    world_sky_layer_top_radius ( 800 )
    world_sky_layer_top_height ( 800 )
  )
  world_sky_layer_edge
  (
    world_sky_layer_edge_steps
    (
      2
      world_sky_layer_edge_step_height ( 500 )
      world_sky_layer_edge_step_radius ( 1400 )
      world_sky_layer_edge_step_height ( 240 )
      world_sky_layer_edge_step_radius ( 1650 )
    )
  )
  world_anim_shader
  (
    world_anim_shader_frames
    (
      1
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvtiles ( 2 2 )
        world_anim_shader_frame_uvscroll ( 0 0.01 )
      )
    )
    world_shader ( BlendATexDiff
      terrain_texslots ( 1
        terrain_texslot ( snowskylo.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
```

```
        )
      )
```

in the correct sequence before or after similar sections in a *.env file (the correct sequence is such that sky layers appear in the *.env file in the order from high to low altitude);
- adjust all the numbers and names shown in red above (see the next questions for specific changes).

## HOW DO I CHANGE A SKY LAYER'S HEIGHT?

- adjust the red numbers (given in meters) in the lines

```
world_sky_layer_top_radius ( 800 )
world_sky_layer_top_height ( 800 )
```

and

```
2
world_sky_layer_edge_step_height ( 500 )
world_sky_layer_edge_step_radius ( 1400 )
world_sky_layer_edge_step_height ( 240 )
world_sky_layer_edge_step_radius ( 1650 )
```

(the number 2 is the number of "edges" that are defined; each edge needs 2 lines, one for its height and one for its radius; I recommend making the outer edge have a height 0, or even negative values like -100 in mountains).

## HOW DO I CHANGE THE TEXTURE OF A SKY LAYER?

- in the next line, change the texture file name to another containing a different texture:

```
terrain_texslot ( snowskylo.ace 1 0 )
```

(make sure it has the right transparency).

## HOW DO I CHANGE A SKY LAYER TEXTURE'S SIZE AND REPETITION?

- adjust the red numbers in the line

```
world_anim_shader_frame_uvtiles ( 2 2 )
```

(larger/smaller values create more/fewer smaller/larger copies of the texture; these numbers need not be whole numbers).

- you can create a mirrored ("seamless") repetition by using the value 2 in the line

```
terrain_texslot ( snowskylo.ace 2 0 )
```

## HOW DO I CHANGE A SKY LAYER TEXTURE'S SPEED AND DIRECTION?

- adjust the red numbers in the line

```
world_anim_shader_frame_uvscroll ( 0 0.01 )
```

(the first number gives the speed component to the west, the second to the north; negative values reverse the direction of motion).

## HOW DO I REDUCE THE "SEAMS" BETWEEN REPEATED TEXTURES IN A SKY LAYER?

- in the next line, use the value 2

```
terrain_texslot ( snowskylo.ace 2 0 )
```

(this will mirror neighboring copies of the texture, so that the textures will flow into each other).

## HOW DO I ADD A FLYING OBJECT?

- create a texture file (*.ace) with the 2D image of a flying object;
- place that file in the route's ENVFILES\TEXTURES folder;
- create a new sky layer (see above);
- adjust its properties (see above).

## HOW DO I CHANGE THE SUN'S OR MOON'S SIZE?

- adjust the red numbers in the following 2 lines (separately for the sun and moon)

```
world_sky_satellite_low_scale ( 800 )
world_sky_satellite_high_scale ( 400 )
```

(halving these values halves the apparent sizes of the sun or moon).

## HOW DO I CHANGE WHERE THE SUN OR MOON RISES OVER THE HORIZON?

- adjust the red number in the following line (separately for the sun and moon)

```
world_sky_satellite_rise_position ( 95 )
```

(the number gives in degrees the rise position clockwise from north, so 90 = east; by default, the moon is made to rise incorrectly at 0 degrees = north: change that to a value near 90).

## HOW DO I CHANGE THE SUNRISE AND SUNSET TIMES, OR THE MOONRISE AND MOONSET TIMES?

- adjust the red numbers in the following 2 lines (separately for the sun and moon)

```
world_sky_satellite_rise_time ( 06:00:00 )
world_sky_satellite_set_time ( 19:00:00 )
```

(the numbers measure the time of day from midnight, from 00:00:00 to 23:59:59).

## HOW DO I CHANGE THE COLOR OF THE SKY AT SUNRISE/NOON/SUNSET?

- adjust the red quantities in the following 3 lines (only for the sun!  not for the moon!)

```
world_sky_satellite_dir_rise_colour ( ffffee8b )
world_sky_satellite_dir_high_colour ( ffffffff )
world_sky_satellite_dir_set_colour ( ffffb573 )
```

(these colors do not affect the color of the fog;  to make the fog and sky have similar colors, make the `dir_rise_colour` and `dir_set_colour` the same as the `fog_day_colour`;  it may not be possible to obtain exactly the same colors;  see Appendix I for the meaning of the color codes).

## HOW DO I CHANGE THE COLOR OF THE TERRAIN AND OBJECTS AT SUNRISE/NOON/SUNSET?

- adjust the red quantities in the following 3 lines (only for the sun!  not for the moon!)

```
world_sky_satellite_amb_rise_colour ( ff101010 )
world_sky_satellite_amb_high_colour ( ff707070 )
world_sky_satellite_amb_set_colour ( ff101010 )
```

(the shown default values are shades of gray;  see Appendix I for the meaning of the color codes).

## HOW DO I CHANGE THE SUN OR MOON TEXTURE?

- change the contents of the files sun.ace or moon.ace;

OR

- create new texture files for sun or moon (with new names);
- place it/them in the route's ENVFILES\TEXTURES folder;
- change the file name in the line

```
terrain_texslot ( sun.ace 1 0 )
```

and/or

```
terrain_texslot ( moon.ace 1 0 )
```

## HOW DO I CHANGE THE MOON'S INTENSITY?

- change the intensity of the graphic in the moon.ace texture file (or create a new texture file, as in the last question/answer);
- delete the lines

```
world_sky_satellite_light ( 0 )
world_sky_satellite_fog ( 96 )
```

(the first of these two lines links the fog brilliance to the height of the moon, which is incorrect if the moon is visible in daytime; the second line links the moon's intensity to the height of the sun, which is astronomically incorrect).

## HOW DO I ADD A "SATELLITE"?

- create a texture file with a 2D image of a satellite;
- save it in the route's ENVFILES\TEXTURES folder;
- change from 2 to 3 the value in the lines

```
world_sky_satellites
(
  2
```

- add a new section like the following after the section for the moon:

```
world_sky_satellite
(
  world_sky_satellite_low_scale ( 400 )
  world_sky_satellite_high_scale ( 350 )
  world_sky_satellite_rise_position ( 0 )
  world_sky_satellite_rise_time ( 18:30:00 )
  world_sky_satellite_set_time ( 08:00:00 )
  world_sky_satellite_dir_rise_colour ( ff151530 )
  world_sky_satellite_dir_high_colour ( ff151530 )
  world_sky_satellite_dir_set_colour ( ff151530 )
  world_sky_satellite_amb_rise_colour ( ff101020 )
  world_sky_satellite_amb_high_colour ( ff101020 )
  world_sky_satellite_amb_set_colour ( ff101020 )
  world_sky_satellite_light ( 0 )
  world_sky_satellite_fog ( 96 )
  world_anim_shader
  (
    world_anim_shader_frames
    (
      1
      world_anim_shader_frame ( )
    )
    world_shader ( BlendATex
      terrain_texslots ( 1
        terrain_texslot ( moon.ace 1 0 )
      )
      terrain_uvcalcs ( 1
```

```
                terrain_uvcalc ( 1 0 0 0 )
              )
            )
          )
        )
```

- delete the lines

```
          world_sky_satellite_light ( 0 )
          world_sky_satellite_fog ( 96 )
```

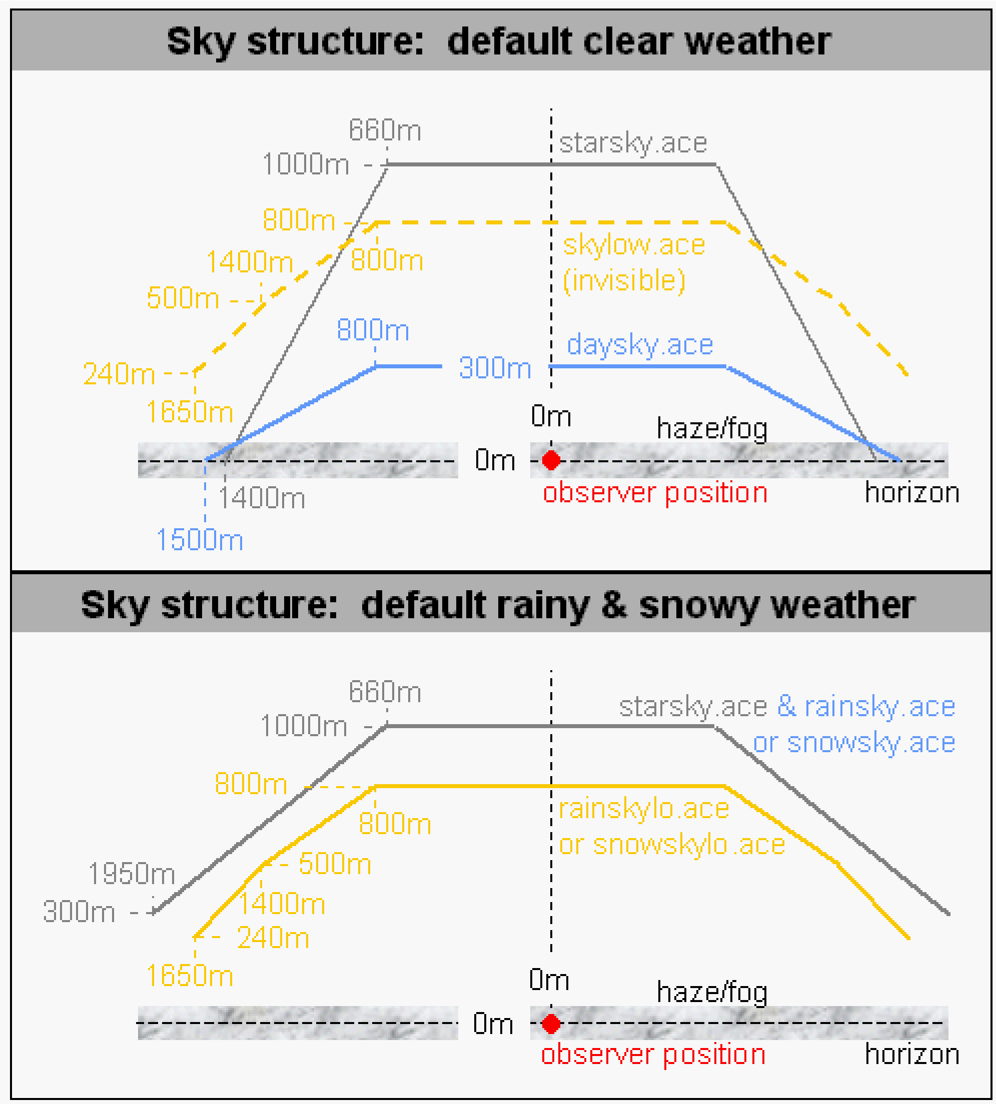
- adjust the quantities in red, as discussed above.

(Note that this satellite will only rise and set once a day, at the chosen times.)

3.15.2b Structure of the sky

We need to understand how the sky is structured in MSTS. **You can think of the MSTS sky as being several tents on top of each other**. The observer (whether the train driver in MSTS or the camera in RE) is in the common center of the tents, looking up at those tents. Some of the tents are opaque, while others are semi-transparent (you can control that). The tents will follow the observer, so he/she/it remains in their center, including when the observer rises or drops in altitude, even if he/she/it goes above the terrain, such as on a bridge. **The sky textures (graphics) will be "projected" onto those tent surfaces**, and will slide along them as controlled in the *.env files.

This sky structure is sketched in the next figure, which shows two cross-sections of the sky above the observer.




Let's first look at the sky structure defined by the default clear weather (as given by a default file sun.env): that is the topmost of the two sketches. It consists of 3 sky layers (3 tents). The **layer labeled as "daysky.ace"** (the innermost tent in this case) gives the blue sky that you see on clear days in MSTS: it goes all around the horizon and gets its graphics from the file daysky.ace; most importantly, it is opaque. The **top layer, labeled "starsky.ace"**, carries the starry sky at night. It uses an opaque graphic contained in starsky.ace. What is supposed to happen in MSTS at sunset is that the starry sky slowly fades in as the blue day sky slowly fades out, while the reverse should happen at sunrise; however, MSTS appears to make the fading in occur suddenly at the start of the evening fade-in period, while the fading out occurs at the end of the morning fade-out period.

The **middle layer, labeled "skylow.ace"**, is light blue, but perfectly transparent by default, so it is invisible and it does not matter where it is in this case; we will see in a moment that this layer is often used to paint low clouds against an opaque sky layer.

**The numbers in the sketch indicate the "tent dimensions" (in meters)**, as far as I understand them: you will find them all in the sun.env file, and you can change them all; they include a height (above the observer), and a radius (from the observer). They appear, for example, as follows in a sun.env file, defining the top and outer edge of the "daysky" layer:

```
world_sky_layer_top_radius ( 800 )
world_sky_layer_top_height ( 300 )
```

and

```
world_sky_layer_edge_step_height ( 0 )
world_sky_layer_edge_step_radius ( 1500 )
```

We will discuss these dimensions individually further below, in section 3.15.2g. Of particular importance is the outer lower edge of the "tents": they often become visible, thereby artificially cutting off your clouds above the horizon; I find it usually necessary to pull those outer edges down to the horizon (height 0) or even below in mountainous terrain (negative height).

It is possible to turn on **fog** in MSTS and RE, as we will see shortly. However, fog will NOT affect your visibility of the sky: so I prefer to call it "ground fog"! **MSTS fog only obscures terrain and objects**. If you want the fog to obscure the sky (I call this "sky fog"), you have to make the skylow layer have a foggy appearance, as illustrated further below. This is slightly tricky, because you have to try to make the fog that obscures terrain and objects have the same color as the fog that obscures the sky: that is not always possible, so you will probably see a sharp line where they meet.

The sun and moon are drawn against the opaque high sky layer, either daysky or starsky, depending on the time of day or night. This means that the **sun and moon show up BELOW the sky painted on daysky or starsky**, even if that sky has clouds! But **you can make other clouds pass below the sun and moon, by including those clouds in the skylow layer**. The result can be quite realistic.

**IMPORTANT: MSTS draws the sky layers in the order in which they are defined in the *.env file**, NOT in the order from highest to lowest (as given by their height). **So the sky layers must be defined in the *.env file in the order needed for correct perspective** (so that lower clouds hide higher clouds, for instance). However, it seems that it is not necessary for the successive "tents" to fit inside each other. In other words, the "height" of a sky layer (tent) need not be related to the actual height of a cloud layer, for example. If you first define a low tent, and then a high tent in the *.env file, the low tent will not hide the high tent: while this may seem illogical, it actually gives you more freedom to shape each tent independently of the others for optimum appearance. (Note that the sun and moon don't count as sky layers.)

NOTES:
- the sun and moon are drawn after the high cloud layers are drawn, and before the lower cloud layers are drawn;
- a sky "tent" may dip below the horizon (with a negative height with respect to the observer); this may be useful in mountainous terrain;

- however, no matter how small and how low below the horizon you make a sky "tent", it will not hide the terrain or objects;  this is because the terrain and objects are drawn after the sky is drawn, against that sky;  in particular, a small tent will not hide distant mountains;  this is important for "sky fog" (very low clouds):  it will never obscure mountains;
- remember that fog only affects the terrain and objects, not the sky.

Let's now look at the situation with the default rainy and snowy weather, shown in the second sketch of the last image.

In these two cases, the day and star skies have the same shapes:  their two tents are identical.  But the daytime graphics come from the files rainsky.ace and snowsky.ace, both being opaque. (Confusingly, you will find rainsky.ace in snow.env, and snowsky.ace in rain.env, but the file names are less important than the graphics they contain, I suppose!)

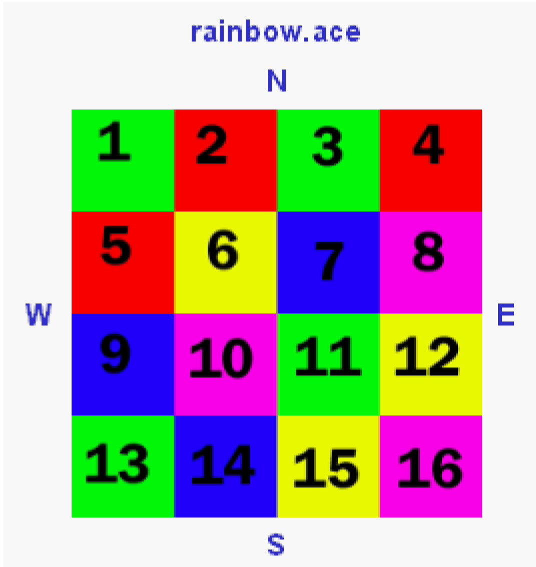
Below the top layer(s), we find the low sky layer, labeled as "rainskylo.ace" and "snowskylo.ace", respectively (by default, they are again interchanged between rain.env and snow.env).  This layer carries images of dense clouds, but it is semi-transparent, so it allows seeing some higher clouds and occasional glimpses of the sun and moon.

The sky layers (tents) in the second sketch do not reach down to the ground:  I find that this sometimes gives a visible lower edge of a sky layer, as if clouds suddenly stopped existing near the horizon (the default rain or snow situation relies on the rain or snow to obscure this effect, but if you remove rain or snow, you will see it).  In general, **I recommend dropping the tent edges down to ground level (height 0)**, or even lower in mountainous terrain (for example, -100 meter).

[You may have noticed that the low sky tent shape in the clear-weather case (sun.env) is the same as in the rainy (rain.env) and snowy (snow.env) cases.  Probably this layer was designed for the rainy and snowy weather, and left unchanged but made transparent for the case of clear weather.]

### 3.15.2c Sky shape and textures

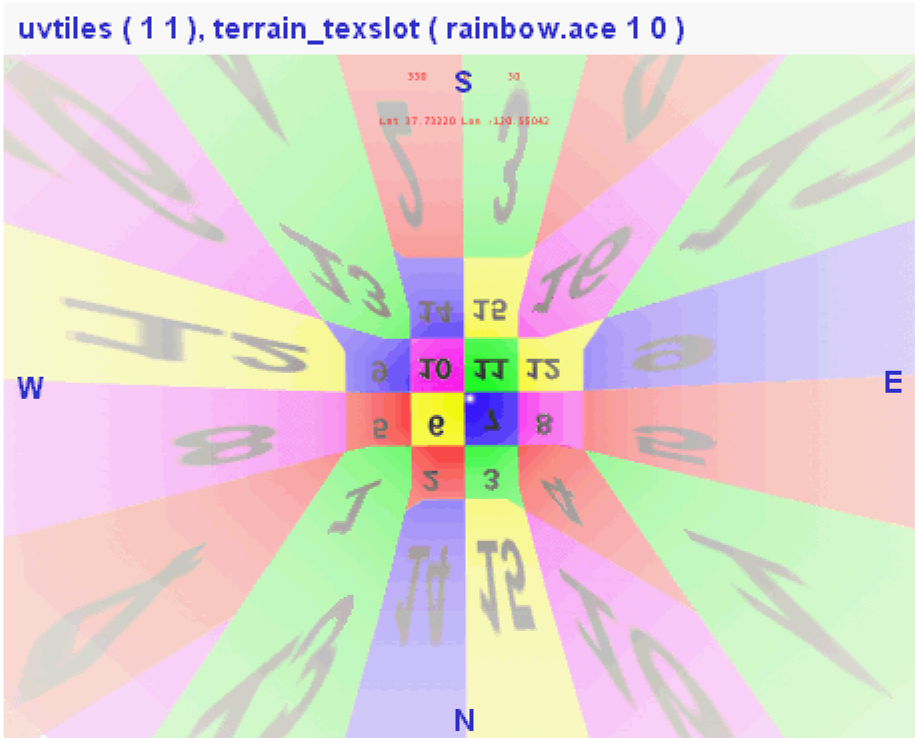
Let's look a little more closely at the **shape of the sky layers, and how the textures are projected on them**.  The next figures show you the sky in the RE, after replacement of the opaque layer by a test pattern (which you can find as rainbow.ace in the ENVFILES\TEXTURES folder of the default EUROPE1 route).  The texture rainbow.ace consists of a single big square containing 4x4 smaller squares with different colors and numbered 1 through 16 (it is not a rainbow!), arranged in the following pattern:



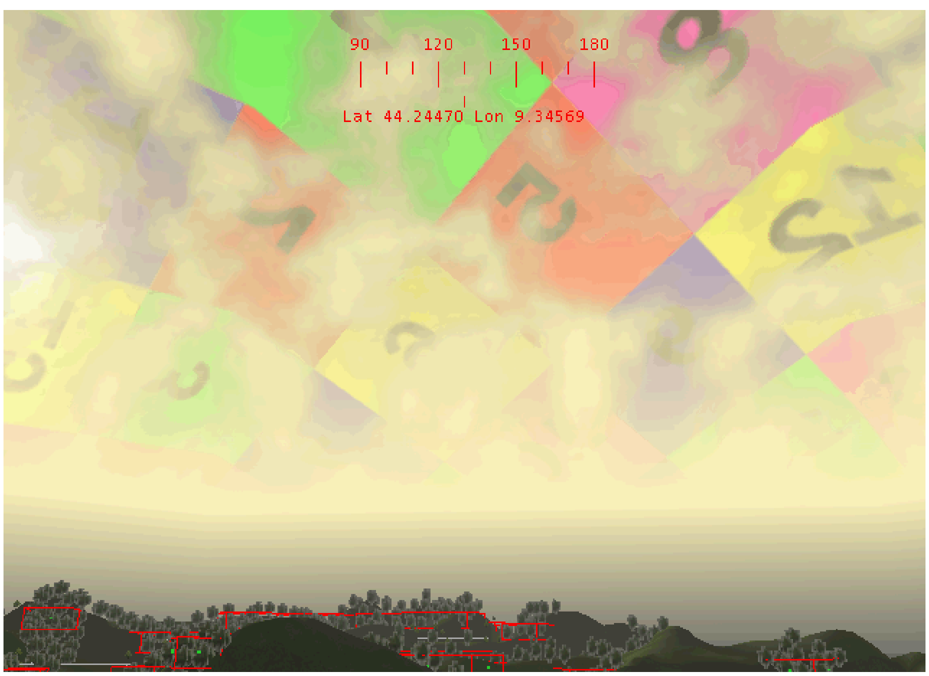

The next pictures show this pattern used as the top (opaque) layer in the sky. First is shown a "fish-eye lens" view straight up, spanning nearly to the horizon in all corners, with clear weather (my small sun is visible between the numbers 7 and 10). The compass directions are shown. In this view the default settings are used for the sky layers (except for the texture name).

**The sky is covered by repeating copies of the texture.** By default, it takes almost 3 copies of the texture pattern (each having 4x4 colored squares) to cover the distance from one horizon across the sky to the opposite horizon: so, **by default, it takes almost 3x3 copies of a normal sky texture to cover the whole sky from north to south and east to west**.

You can see that this "rainbow" **pattern is drawn as if projected from above the sky ceiling down onto the outermost tent**; that is why the numbers are inverted - you read them from the back. From the compass directions, you can also see that the top of this pattern (the side with the numbers 1 2 3 4) points to the north, while the right of the pattern (where the numbers 4 8 12 16 are) points to the east, as labeled in the picture above. (This can get very confusing!)

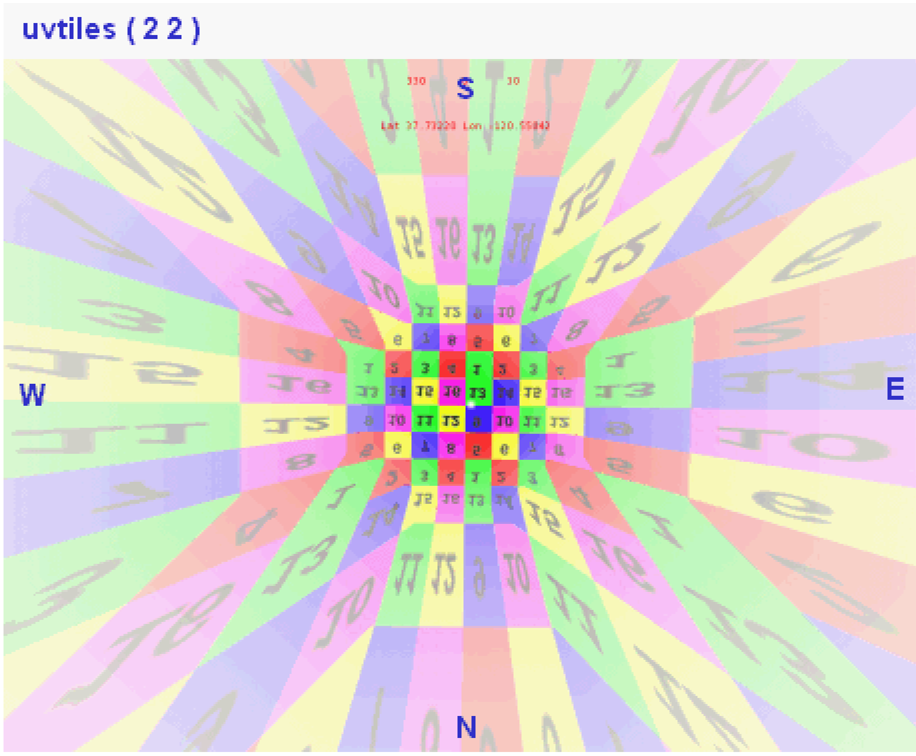



The next picture gives a more "normal" view, with the default foggy horizon.  It also contains a cloudy low sky layer (its outer edge is clearly visible where its yellow color ends and shades of gray start), and you can just guess the white sun partially hidden by it at the far left edge.




Less obvious in these pictures is that the edges of the squares have "kinks" in them:  they are not perfectly straight lines.  This is much more obvious in the next fish-eye picture, in which the squares are half as large [this is achieved with the command `uvtiles ( 2 2 )` in the *env files, which replaces each copy of the texture with 2x2 copies].

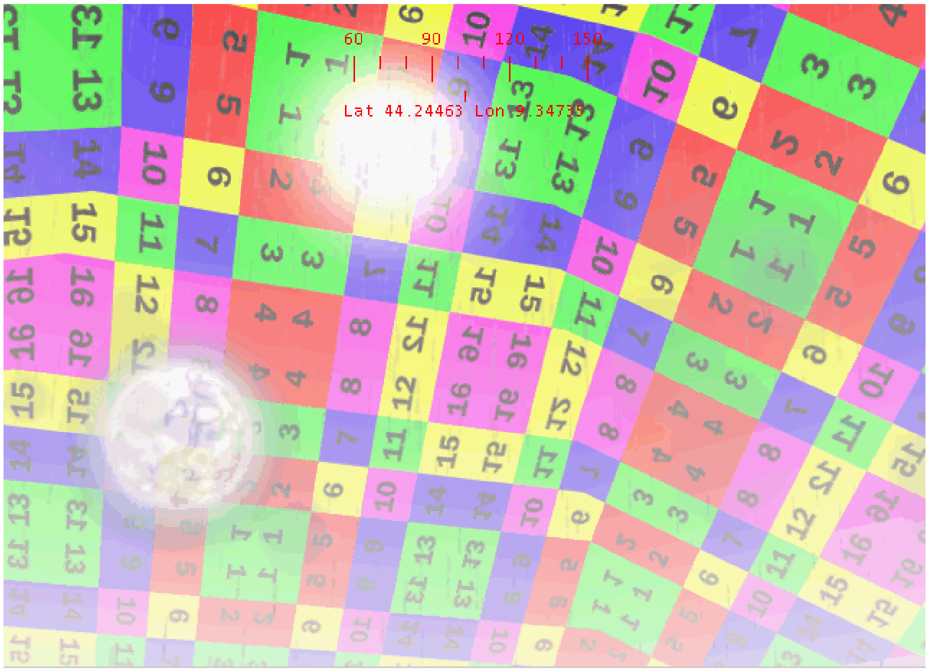
What is happening is this: the flat top of the sky is not a piece of a sphere or a flat circular disk, but an 8-sided flat polygon (an octagon). From each of its 8 edges, a flat surface slopes down toward the ground (near the horizon). So **a sky layer is an 8-sided tent composed only of flat surfaces**.




The sides of this tent may be broken into two (or more) levels, as shown in the above sketches for the low sky layer. This gives some more flexibility for handling the sky near the horizon: by adding "edges" (as they are called in the *.env files), you can make the tent look a bit more like a sphere.

### 3.15.2d Sky texture design

The next picture helps understand an important point about the design of textures to simulate a cloudy sky. When I took that screenshot, the "high-level wind" was blowing from north to south (from left to right in the picture). This is achieved simply as a constant flow of the texture along the layer. This means that the texture rises from the north horizon and dips below the south horizon, as it should. However, this also means that a cloud pattern will be upside down when it dips below the southern horizon, compared to when it rises from the northern horizon. This could look very strange if your cloud image has obvious "up" and "down" directions (as with clouds illuminated from above by the sun). Thus, **avoiding upside-down clouds requires some care in designing a sky texture that has no obvious up or down directions!**

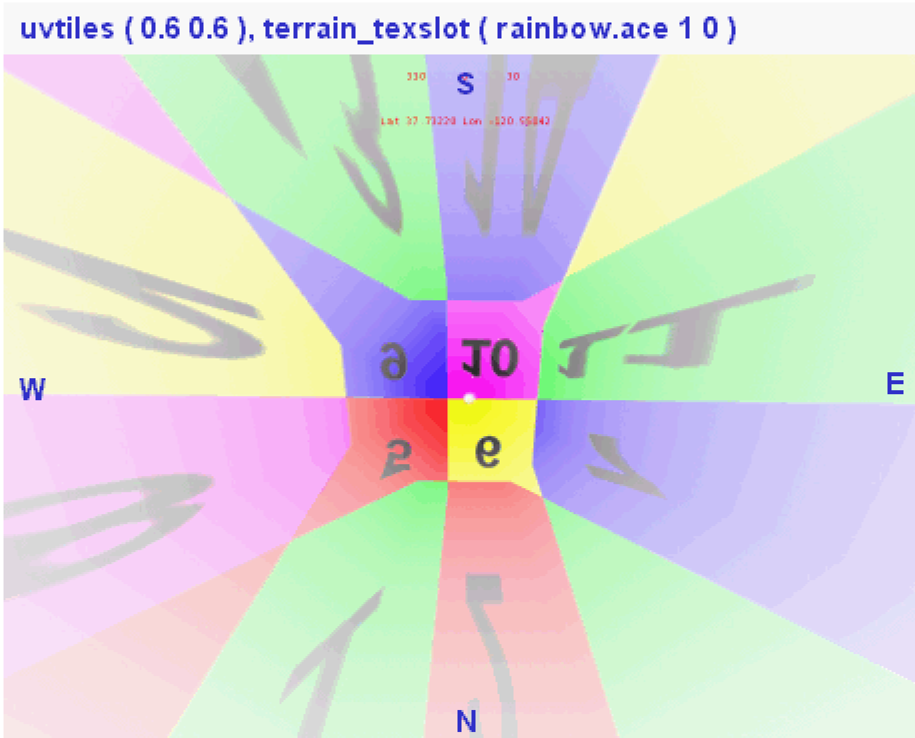



Also illustrated in the last picture is a solution to another problem (it's a quite general problem with repeating textures): **how to avoid mismatches at the seams between repeated textures**. Here the basic 4x4 set of squares is mirrored in a special form of tiling to minimize the seam mismatch problem. You can see how the basic 4x4 texture is mirrored so that the seams always occur between squares of the same color and with the same number: that way, the color and texture pattern continues smoothly across these seams. This mirroring is achieved by using the number 2 in the command `terrain_texslot ( rainbow.ace 2 0 )` in the *env files.

**You may want to produce a sky such that only one copy of the texture covers the entire sky**, as in the next picture. To do this, you can stretch the texture so it reaches the horizon all around. This is done with the command `uvtiles ( 0.6 0.6 )` in the *env files, which replaces each copy of the texture with (0.6x0.6) copies (you can vary the value 0.6 as you wish).

One reason why you may want to do this is **to draw a single object in the sky, like an airplane, balloon or UFO**. This may be done by defining a lower sky layer that uses a texture composed not of clouds but of a flying object (and a suitable transparency, so you still see the sky above it). You can make the object fly in any direction and at any speed you wish, by giving the texture the right direction of motion and speed. You can also draw **lightning** this way, but you need to make it "fly" at a very high speed so it jumps around the sky rather than flow smoothly.

NOTE: It is tricky to place a stationary object this way (using zero speed). The difficulty is knowing where on the texture to draw the object so it appears in the correct place in the sky. My recommendation is to use the next picture as a guide: decide where in that picture you want the object to appear; note where that is with respect to the "rainbow" test pattern; and draw it then in the right spot on your texture, with the right orientation (reversed since the observer will be looking from the back of the texture).




**You may choose different texture resolutions, depending on the detail needed**: textures with 128x128, 256x256, 512x512 and 1024x1024 pixels are all acceptable (smaller sizes may work as well, but remember that a texture gets magnified enormously as it is stretched across the sky, so you will see the individual pixels magnified). However, I find that with a 1024x1024 size, you can't use a semi-transparent alpha channel: the alpha channel may only contain black and white pixels, at least with TGATool2.

In designing a texture, you will also have to pay attention to the color resolution: the 24-bit color resolution used in the *.ace texture files can give **noticeable steps in colors**, as you can see in the last picture when you move out from its center to its edge (particularly in the blue "square" numbered 7). This is most noticeable for very slowly varying colors, and may become distracting and unsatisfactory. To reduce this effect, you could choose quickly varying colors.

You can find sky pictures on the web that can be converted to sky textures for MSTS. For example, you will find real-sky pictures at http://www.aaabackgrounds.com/textures/sky/ and http://www.art.net/~jeremy/photo/public_texture/. But remember that you will have to "wrap" these to fill a complete sky!

### 3.15.2e Sun and moon

The last several pictures show very clearly that **the sun and moon are drawn against the opaque top sky layer**, which normally has clouds painted on it! This means that, **to avoid giving the impression that the sun and moon are lower than the clouds, you will need to have another cloudy layer below sun and moon**.

It also shows something else: by default, both **sun and moon are drawn far too big**. Exactly how big is not clear: the compass bar indicates that the sun and moon images are about 50-60° in diameter, but the 170° fish-eye view suggests about 20°. In reality sun and moon both have a diameter of just over 0.5°, as seen from the Earth: so the sun and moon are drawn anywhere from 40 to 100 times too large; it's as if sun, earth and moon were 40 to 100 times closer together than they actually are! As we will see later, that can be easily fixed, but probable reasons that sun and moon are drawn so large are first to simulate the blinding brilliance of the sun, and secondly to show some detail on the moon's surface. A compromise is about 10 times smaller than the default size, as illustrated in the last picture. Note also that you don't get to see the phases of the moon: a single image of the moon is all you can use.

### 3.15.2f Fog

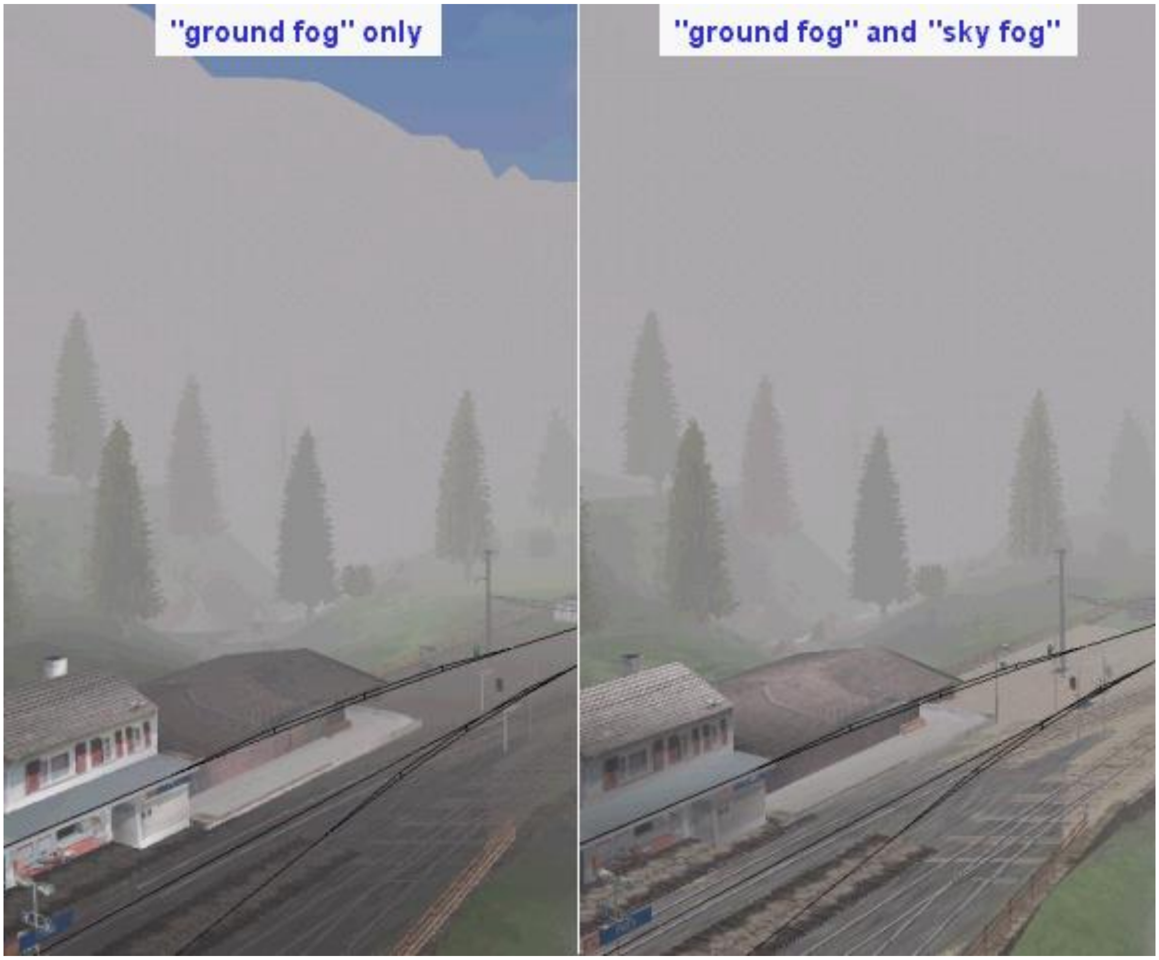
With the help of an example I will here give an impression of what you need to do to get proper fog in your route.

As mentioned earlier, fog can be turned on (in the *.env files, including the sun.env file). But fog has a problem, as you can see at left in the next picture (from my Albula Line 2): while the entire mountain turns uniform gray (meaning invisible behind the fog), the sky is not affected by the fog at all! The same is true with buildings, trees, etc.: they become gray, while the sky behind them remains perfectly clear, as you can see in the title image of this Guide (see title page). So, **this fog only obscures terrain and objects: it does not obscure the sky**. Therefore **I call this "ground fog"**.

This may or may not satisfy you. To obtain what I consider a more realistic fog appearance, **it is necessary to make the low sky layer equally "foggy"**, giving the result at right in the picture: **this adds what I call "sky fog"**. For that, I used a pure white texture with rather low uniform transparency, expecting to get a gray sky. But the resulting sky at first turned out somewhat pink, because MSTS realized it was near sunset; however, the fog in front of the mountain stayed a non-pink gray, still forming an illogical contrast (imagine the left picture with a pinkish rather than blue sky). To get the proper result, **I also had to remove the reddish tint radiated by the setting (or rising) sun and turn it gray**.

There is one more serious problem: if you now change the time of day (for example, by pressing + or - in RE), you will see the "horizon" reappear. The reason is that the gray levels of "ground fog" and "sky fog" do not stay equal to each other away from noon. I have not found a fix for that.

So you see that it takes a number of steps to achieve some degree of realism here.




CAUTION: Fog can seriously slow down your frame rate!

WARNING: **The MSTS treatment of how the sky and textures change appearance with time of day is quite complex and somewhat unpredictable**. You may get exactly what you want at noon, but if you change the time to sunrise, sunset or night, you may find a result that you did not expect or want. It can be a challenge to produce exactly what you want at all times of day and night!

### 3.15.2g The sky details in the *.env files

To explain how to modify the appearance of the sky, we will **look in detail at a default version of the file rain.env**. Much the same can be done with the sun.env and snow.env files, since the sky sections of these files are nearly identical.

CAUTION: Before changing these files in your route, **make sure to back up the original versions**, because it is easy to make mistakes with all those opening and closing parentheses: if you mismatch even one parenthesis, the RE and MSTS will crash!

Below, **the numbers in red are those that you may change**; I don't know the meaning of the others, so I recommend great caution with those.

NOTE: If you precede a line by a % symbol, that line will be ignored by MSTS and RE. However, be careful not to break the sequence of parentheses that way! Remember to use a Unicode-capable word processor like WordPad.

**The default rain.env file starts like this:**

```
SIMISA@@@@@@@@@@JINX0w0t______

world
(
      world_fog_distance ( 450 )
      world_fog_day_colour ( ffffffff )
      world_fog_night_colour ( ff000000 )
```

The lines containing `world_fog` control the "**ground fog**" that you see in MSTS rain (or snow).

If you look at the sun.env file, you will find that these 3 lines are absent: there is no fog on a clear day, by default. But you may insert these lines into sun.env and create fog without rain or snow (as I did for the pair of foggy pictures above).

The `fog_distance` determines the visibility range: you can change it from the default 450 m to whatever you desire (the pair of foggy pictures above uses 150 m).

The next two lines control the color of the fog at noon (`day_colour`) and at night (`night_colour`). By default these are white (`ffffffff`) and black (`ff000000`). I recommend increasing the night color to something like `ff404020` (dark yellowish): that prevents the terrain and objects from turning very dark before sunset (and right after sunrise). At other times between noon and sunrise or sunset, the fog changes its color gradually between these two colors; so in mid-morning and mid-afternoon, you should expect to get a fog that is mid-gray by default.

NOTE: See Appendix I for an explanation of the color codes used here. A very brief summary of color codes:
- the color code has the form "ttrrggbb", where "tt" is transparency (which is apparently not used in MSTS and RE);
- the color intensities "rr" (red), "gg" (green) and "bb" (blue) are given in hexadecimal notation, with "00" = zero intensity, and "ff" = maximum intensity;
- each hexadecimal character can range from 0 (smallest or darkest), via 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d and e to f (highest or brightest).

Continuing in the rain.env file, we read (you will find minor differences between the rain.env files present in different default routes):

```
world_sky
(
  world_sky_nlayers_behind_satellites ( 2 )
```

The above line specifies "nlayers behind satellites": the satellites, as we will see later, are the sun and the moon. Perhaps this line implies that the first two layers defined below (the daytime sky and the night sky) will be drawn BEHIND the sun and moon, which is indeed what happens.

Next we start defining the **3 sky layers: two top sky layers (a daytime sky and a night sky) and one low sky layer**:

```
  world_sky_layers
  ( 3
```

Note that if you change the 3 to 2, the third sky layer will be ignored: this is a simple way to remove the third layer (the low sky layer by default), without removing all the lines defining that layer below.

CAUTION: If you reduce that 3 to 2, DO NOT remove the lines defining the third layer. RE and MSTS will crash if you remove it. Leave the third layer's lines: they will be ignored, but they must be present.

We now define the **first of the 3 sky layers**, starting with its top part (the horizontal 8-sided section overhead the observer):

```
    world_sky_layer
    (
      world_sky_layer_top
      (
        world_sky_layer_top_nfaces ( 8 )
        world_sky_layer_top_radius ( 660 )
        world_sky_layer_top_height ( 1000 )
      )
```

The 8 may refer to the 8 sides of the top, but changing it to 4 or 16 has no effect, so its meaning is uncertain. The 660 and 1000 are the radius and height of that section (in meters), as drawn in the second sketch in section 3.15.2b.

Next is defined the outer edge of this layer, again as illustrated in the earlier sketch:

```
      world_sky_layer_edge
      (
        world_sky_layer_edge_steps
        (
          1
          world_sky_layer_edge_step_height ( 300 )
          world_sky_layer_edge_step_radius ( 1950 )
        )
      )
```

I recommend setting the height 300 to zero (or even negative values like -100 in mountains): otherwise, you may see that sky layer artificially cut off near the horizon.

This layer's animation is specified next.

```
world_anim_shader
(
  world_anim_shader_frames
  (
    1
    world_anim_shader_frame
    (
      world_anim_shader_frame_uvscroll ( 0 0.02 )
    )
  )
```

The `uvscroll ( 0 0.02 )` controls the speed and direction of motion of this layer. The first number (0 here) gives the speed component to the west, while the second number (0.02 here) gives the speed component to the north. Negative values reverse the direction of motion. I am not sure of the speed unit, but speed values in the range 0.001 to 0.2 are realistic (this depends also on how high you place your sky!).

[You can multiply and shrink the textures by adding a line before the line containing `uvscroll`. If you add the line

```
world_anim_shader_frame_uvtiles ( 2 2 )
```

you will replace each copy of the texture by a set of 2x2 copies at half size. This was used to generate the picture with many small squares in the sky in section 3.15.2c. You may use unequal and even non-whole positive numbers here, such as ( 4 3 ) or ( 1.5 0.75 ).]

Following this, the texture for this layer is given as `snowsky.ace`:

```
    world_shader ( TexDiff
      terrain_texslots ( 1
        terrain_texslot ( snowsky.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
```

This texture is opaque. Why `snowsky.ace` in rain.env? You will conversely find that `rainsky.ace` is used in snow.env! Possibly, the designers changed their minds about which textures to use for rainy and snowy weather, without adjusting the file names.

The number 1 following `snowsky.ace` can be changed to create the special tiling effect mentioned earlier: if you change this 1 to 2, you get mirrored versions of the texture, as illustrated for the sky in section 3.15.2c. This reduces the seams that may occur at the texture edges with simple repetition.

The command `terrain_uvcalc ( 1 0 0 0 )` must contain some secret effects that remain to be discovered… (one guess is that it allows defining different weather in different tiles, but how?).

Next comes the **second of 3 sky layers**: it defines the **starry night sky**, replacing the previous daytime sky layer when needed. Its definition is very similar to the last layer, with three differences noted below:

```
world_sky_layer
(
  world_sky_layer_fadein ( 18:00:00 20:00:00 )
  world_sky_layer_fadeout ( 06:00:00 08:00:00 )
  world_sky_layer_top
  (
    world_sky_layer_top_nfaces ( 8 )
    world_sky_layer_top_radius ( 660 )
    world_sky_layer_top_height ( 1000 )
  )
  world_sky_layer_edge
  (
    world_sky_layer_edge_steps
    (
      1
      world_sky_layer_edge_step_height ( 300 )
      world_sky_layer_edge_step_radius ( 1950 )
    )
  )
  world_anim_shader
  (
    world_anim_shader_frames
    (
      1
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvtiles ( 2 2 )
      )
    )
    world_shader ( BlendATex
      terrain_texslots ( 1
        terrain_texslot ( starsky.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
```

Notice the two lines containing `world_sky_layer_fadein` and `world_sky_layer_fadeout`: these are supposed to make the starry sky fade in and out over a 2-hour period, replacing the previously defined sky layer, but this actually happens suddenly in both RE and MSTS. You can change the timing as desired.

This layer does not move (no `uvscroll` command): you could make it move by adding a line with the `uvscroll` command. But it will not rotate around the North and South Poles the way it should in reality; instead it will move like a cloud layer (but who will notice that, given the low speed of the stars across the sky?).

This layer uses the texture in `starsky.ace`, which is also opaque.

Next we define the **third of the 3 sky layers**. It will be the **low sky layer** and will need a partly transparent texture to see the high sky layer as well as the sun and moon.

```
  world_sky_layer
  (
    world_sky_layer_top
    (
      world_sky_layer_top_nfaces ( 8 )
      world_sky_layer_top_radius ( 800 )
      world_sky_layer_top_height ( 800 )
    )
    world_sky_layer_edge
    (
      world_sky_layer_edge_steps
      (
        2
        world_sky_layer_edge_step_height ( 500 )
        world_sky_layer_edge_step_radius ( 1400 )
        world_sky_layer_edge_step_height ( 240 )
        world_sky_layer_edge_step_radius ( 1650 )
      )
    )
    world_anim_shader
    (
      world_anim_shader_frames
      (
        1
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvtiles ( 2 2 )
          world_anim_shader_frame_uvscroll ( 0 0.01 )
        )
      )
      world_shader ( BlendATexDiff
        terrain_texslots ( 1
          terrain_texslot ( snowskylo.ace 1 0 )
        )
        terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
    )
  )
)
```

CAUTION: Do not remove this third layer's section, even if you reduce the number of layers defined earlier from 3 to 2 (RE and MSTS will crash if you remove this section).

You notice two main differences above compared to the first two sky layers. First, this layer contains two edges, as illustrated in the sketch in section 3.15.2b. Second, this layer uses the `uvtiles` command to shrink and multiply the `snowskylo.ace` texture across the sky. You may change this to stretch or further shrink your texture (if you delete the line with `uvtiles`, it will stretch by a factor 2).

Notice that this layer has a different speed (`uvscroll`) than the first layer defined above: it will therefore be seen to slide against the higher cloud layer.

You can use this layer to create fog in the sky. I made the foggy sky illustrated above by using for the top radius and height the values 200 and 100 m, and for the edges the values 80, 240, -100, 300 instead of 500, 1400, 240, 1650 (the negative value makes the foggy layer penetrate down into valleys); but remember that no matter how small you make this tent, distant mountains will not be obscured by it, so its size is not critical! I used a white texture that is opaque to hide the sun and moon; but you could use a semi-transparent color to see a bit of the sky (but also sharp edges around distant terrain and objects, as discussed before).

Next comes a specification for the treatment of the horizon: it probably creates the hazy horizon that we see even in the absence of fog; I don't know whether there are other choices here ("Sharp" instead of "Diffuse" does not work):

```
world_sky_horizon
(
  world_shader ( Diffuse )
)
```

Now that the sky layers have been defined, it is time to create the sun and the moon.

```
world_sky_satellites
(
  2
```

They are called **satellites**, suggesting that you could add more satellites, such as man-made ones (you would increase the value 2 appearing below to 3 or 4 or whatever you need, and add sections defining other satellites; this has not yet been tested, to my knowledge). Perhaps you can also use this method to fly airplanes (drawn on a texture), choosing appropriate times to make them fly at a reasonable speed!

First we define the **sun**:

```
world_sky_satellite
(
  world_sky_satellite_low_scale ( 800 )
  world_sky_satellite_high_scale ( 400 )
  world_sky_satellite_rise_position ( 95 )
```

```
world_sky_satellite_rise_time ( 06:00:00 )
world_sky_satellite_set_time ( 19:00:00 )
world_sky_satellite_dir_rise_colour ( ffffee8b )
world_sky_satellite_dir_high_colour ( ffffffff )
world_sky_satellite_dir_set_colour ( ffffb573 )
world_sky_satellite_amb_rise_colour ( ff101010 )
world_sky_satellite_amb_high_colour ( ff707070 )
world_sky_satellite_amb_set_colour ( ff101010 )
world_sky_satellite_light ( 1 )
```

The `low_scale` and `high_scale` control the apparent size of the sun, varying between the horizon and straight above the observer (in reality the size does not vary, except for a small flattening near the horizon, but our human vision makes it look like the sun and moon are larger when close to the horizon). As mentioned earlier, the sun (and moon) size is grossly exaggerated (by as much as a factor of 100): you can reduce the apparent size here to something you like. I prefer 80 for both sun and moon, even though that removes the "splash" of the sun across the sky, due to its blinding brilliance.

The `rise_position ( 95 )` sets the point on the horizon where the sun will rise, in degrees clockwise from north. A value of 95 degrees is near the east. The `rise_time` and `set_time` specify when the sun must rise and set. (The sun, and the moon also, follow a simple arc through the zenith, the point directly above the observer, so it is not possible, as far as I know, to make the sun and moon rise and set farther north in the northern summer, and farther south in the northern winter, or vice versa in the Southern Hemisphere. Also you can't make the sun and moon stay low in the sky, as they would in polar regions. But you can still lengthen or shorten the days simply by adjusting the times of rising and setting.)

Next come several color definitions for "direct" vs. "ambient" lighting.

The first 3 of these definitions (containing `dir`) tint the color of the sky at sunrise, noon and sunset, respectively. By default, you get a weakly yellowish sky at sunrise (`ffffee8b`), an untinted sky at noon (`ffffffff` = white), and a pinkish sky at sunset (`ffffb573`).

CAUTION: These colors (containing `dir`) do not affect the "ground fog". So, if you want to match the tint of the sky layers to that of the fog (to avoid the contrast problem illustrated for fog in the last section), you should give `dir_rise_colour`, `dir_set_colour` and `fog_day_colour` the same color (such as "`ffffffff`"), and you should also give `dir_high_colour` and `fog_night_colour` the same color (such as "`ff000000`").

The next 3 color definitions (containing `amb`) tint the terrain and objects, again depending on time of day. These 3 tints by default are simply shades of gray: dark at sunrise and sunset (`ff101010`) and mid-gray at noon (`ff707070`).

The line with `satellite_light` appears to link the fog brilliance to the sun height above the horizon. This line is needed for the sun: without it MSTS and RE crash (but not for the moon, as we shall see). Changing the value 1 to something else had no effect for me.

We still need to specify the sun texture (don't change that 1 after sun.ace or you will get multiple suns!):

```
      world_anim_shader
      (
        world_anim_shader_frames
        (
          1
          world_anim_shader_frame ( )
        )
        world_shader ( AddATexDiff
          terrain_texslots ( 1
            terrain_texslot ( sun.ace 1 0 )
          )
          terrain_uvcalcs ( 1
            terrain_uvcalc ( 1 0 0 0 )
          )
        )
      )
    )
```

Now we can repeat all this with slight modifications for the **moon**:

```
    world_sky_satellite
    (
      world_sky_satellite_low_scale ( 400 )
      world_sky_satellite_high_scale ( 350 )
      world_sky_satellite_rise_position ( 0 )
      world_sky_satellite_rise_time ( 18:30:00 )
      world_sky_satellite_set_time ( 08:00:00 )
      world_sky_satellite_dir_rise_colour ( ff151530 )
      world_sky_satellite_dir_high_colour ( ff151530 )
      world_sky_satellite_dir_set_colour ( ff151530 )
      world_sky_satellite_amb_rise_colour ( ff101020 )
      world_sky_satellite_amb_high_colour ( ff101020 )
      world_sky_satellite_amb_set_colour ( ff101020 )
      world_sky_satellite_light ( 0 )
      world_sky_satellite_fog ( 96 )
```

The only major difference here is the new command `satellite_fog ( 96 )`: it allows you to obscure the moon from 96% down to 0% or brighten it to 100%. More importantly and strangely, this line links the moon's intensity to the sun's height above the horizon! In reality the moon's intensity should not vary at all (except for the presence of clouds). So **I strongly recommend removing this line completely!** The reason it works by default is that the moon by default only appears at night, whereas in fact it can appear at any time of day or night. (Note that if you copy this line into the sun's section above, you get only minor changes in the appearance of the sun.)

The line with `satellite_light` has another strange effect: it links the fog brilliance to the height of the moon. That works to some extent if the moon only appears at night, but it is very disturbing when the moon appears in daytime, because it should be the sun that illuminates the fog: here also, **I strongly recommend removing this line completely!**

A moon `rise_position` of 0 degrees is unrealistic (it makes the moon rise in the north and set in the south): this should be something like 90 degrees.

After defining the moon's texture, the world sky is complete:

```
            world_anim_shader
            (
              world_anim_shader_frames
              (
                1
                world_anim_shader_frame ( )
              )
              world_shader ( BlendATex
                terrain_texslots ( 1
                  terrain_texslot ( moon.ace 1 0 )
                )
                terrain_uvcalcs ( 1
                  terrain_uvcalc ( 1 0 0 0 )
                )
              )
            )
          )
        )
      )
```

[NEW SINCE V1.106]

## 3.15.3 WATER

This section explains how to change the appearance of water in MSTS and RE. You can modify its apparent depth and its texture, you can cause wave motion as well as a constant flow, and you can make the water reflect the sky. Thereby you can also create weather-dependent water, as well as season-dependent water.

### 3.15.3a Quick how to

Here I give short answers to common questions, with few explanations. To better understand those answers and how to produce other effects, you will need to read the subsequent sections.

The changes indicated in the answers are to be made inside the appropriate *.env files, unless noted otherwise.

**HOW DO I MAKE WATER VISIBLE?**

- to make water visible in your route, see section 3.6.

[NEW SINCE V2]
## HOW DO I STOP WATER FROM FLASHING IN MSTS?

- to fix this bug, replace 2 or 3 occurrences of `BlendATexDiff` by `BlendATex` in the `world_water` section of each *.env file used by your route (to find out which *.env files are used, check in the route's *.trk file); do not make this change in the `world_sky` or `world_precipitation` sections of the *.env files (see section 3.15.3c for more details).

Unfortunately, this introduces another, less disturbing bug: water and terrain glow at night. And this does not fix another bug: sometimes, depending on the MSTS camera's view angle, you see a constant non-flashing change of the water color, usually to white.

## HOW DO I MAKE THE WATER SEASON-DEPENDENT?

- create additional files of type *.env for different seasons (see 3.15.1);
- adjust the water properties in each of those *.env files;
- refer to the new *.env files in the route's *.trk file.

## HOW DO I CREATE/CHANGE WAVES?

- adjust the red numbers in the lines

```
        world_water_wave_height ( 0 )
        world_water_wave_speed ( 0 )
```

(the wave height is in meters; the wave "speed" is its frequency in cycles per 2-second period: so a "speed" of 1 gives a frequency of one cycle every 2 seconds, while a "speed" of 0.1 gives a frequency of one cycle every 20 seconds).

## HOW DO I CREATE SKY REFLECTIONS ON WATER?

[NEW SINCE V2] WARNING: For some users, adding sky reflections to water can cause a "Train World Initialization Failure" when starting the route: their systems (perhaps older video cards) can't handle the reflecting water layers.

- create a partly transparent texture containing the image that will be seen reflected, typically an image of the current sky; call it something like skyrefl.ace;
- save that texture file in the route's ENVFILES\TEXTURES folder;
- increase by 1 (normally from 3 to 4) the number in the following lines (these lines already exist)

```
    world_water_layers
    (
      3
```

- make sure the following lines are included in the *.env file (they also already exist by default)

```
world_water_layer
(
  world_water_layer_height ( 0 )
  world_water_layer_sky_reflection ( 0 )
  world_anim_shader
  (
    world_anim_shader_frames
    (
      1
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvscroll ( 0 0.1 )
      )
    )
    world_shader ( BlendATexDiff
      terrain_texslots ( 1
        terrain_texslot ( skyrefl.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
```

as the last water layer (make sure these lines appear after the other water layer definitions);

- adjust the red numbers as needed: the height should be at or above the topmost water layer height (the normal choice is to make this height equal to that of the next highest water layer); the 0 after `sky_reflection` allows motion of the texture, as defined by `uvscroll`; setting `sky_reflection` to 1 freezes that motion; include your texture file's name (here `skyrefl.ace`);

- increase the daytime terrain and object brightness by changing the gray level in the line

```
world_sky_satellite_amb_high_colour ( ff808080 )
```

for the sun; here choose `ffffffff` for a clear sunny day, but a darker gray level for cloudy, rainy or snowy skies (for example `ffdddddd` for a light day, `ffaaaaaa` for a medium dark day, or `ff808080` for a very dark day).

## HOW DO I ADD A NORMAL WATER LAYER?

- add 1 to the number in the next lines (changing 3 to 4)

```
world_water_layers
(
  3
```

- add a section like this

```
world_water_layer
(
  world_water_layer_height ( -0.1 )
  world_anim_shader
  (
    world_anim_shader_frames
    (
      1
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvscroll ( 0 0.008 )
        world_anim_shader_frame_uvtiles ( 3 3 )
      )
    )
    world_shader ( BlendATexDiff
      terrain_texslots ( 1
        terrain_texslot ( watermid.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
```

in the desired order of water layers (they will be drawn from bottom up, so they must be ordered in the *.env file such that the bottom layer is defined first and the top layer last);

- adjust the quantities in red as needed (see the next questions and answers).

## HOW DO I CHANGE THE HEIGHT OF A WATER LAYER?

- adjust the number in the line

```
world_water_layer_height ( -1 )
```

(this number gives the depth in meters of this water layer, relative to the local water level defined in each tile, even if it is sloped:  a negative value puts the layer below that level, a positive value above that level).

## HOW DO I CHANGE THE SPEED AND DIRECTION OF A WATER LAYER?

- adjust the numbers in the line

```
world_anim_shader_frame_uvscroll ( 0 -0.008 )
```

(the first number gives the layer's speed component to the west;  the second number gives the layer's speed component to the south;  negative numbers reverse the direction of motion;  a speed of 1 corresponds to about 100 km/h).

## HOW DO I CHANGE THE TEXTURE OF A WATER LAYER?

- in the next line, change the texture file name to another containing a different texture:

```
terrain_texslot (watermid.ace 1 0 )
```

(make sure it has the right transparency).

## HOW DO I CHANGE THE SIZE AND REPETITION OF A WATER LAYER TEXTURE?

- adjust the numbers in the line

```
world_anim_shader_frame_uvtiles ( 8 8 )
```

(the pair of values gives the number of copies, 8x8 in this case, that cover one 128x128 m patch; you may use numbers that are not whole, but watch out for "seams" at patch edges).

## HOW DO I REDUCE THE "SEAMS" BETWEEN REPEATED TEXTURES IN A WATER LAYER?

- in the next line, change the 1 to 2

```
terrain_texslot (watermid.ace 1 0 )
```

(this will mirror neighboring copies of the texture, so that the textures will flow into each other).

## HOW DO I MAKE A WATER LAYER "WIGGLE"?

To make a water layer move back and forth, do the following:

- add the line

```
world_anim_shader_framelen ( 0.2 )
```

before the line

```
world_anim_shader_frames
```

- adjust the number 0.2 to give the time step (in seconds) between jumps of the water;

- replace the lines like these

```
1
world_anim_shader_frame
(
  world_anim_shader_frame_uvscroll ( 0 0 )
  world_anim_shader_frame_uvtiles ( 8 8 )
```

```
)
```

by lines like these

```
4
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0 0 4 4 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0.002 0 4.001 4 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0.003 0.002 4.002 4.002 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0.003 0.004 4.002 4.002 )
)
```

- adjust the red numbers: the number 4 gives the number of jumps before the motion repeats itself: it corresponds to the number of sections labeled `world_anim_shader_frame` (each taking 4 lines); keep the numbers like `0.004 -0.002 4.002 3.998` close to 0 or close to a whole number to get small jumps: if you let them deviate more, you will get larger jumps.

### 3.15.3b What aspects of water can be changed?

This section first describes the changes of the MSTS water that I know are possible; there most likely exist other possibilities. It also explains how water is "organized" in both MSTS and the RE. The next section will explain how these changes are made in practice.

**CAUTION: The water changes discussed below are incompatible with the use of Sky! Conductor** (which changes the sky appearance by replacing files that also contain the water definitions).

**The default water that appears in MSTS can be modified in several ways:**

- you can change the **level** of water (in addition to the tile water level setting);
- you can make **waves**;
- you can change the **water colors/textures and its depth perception**;
- you can make water **flow**;
- you can make water **reflect the sky**.

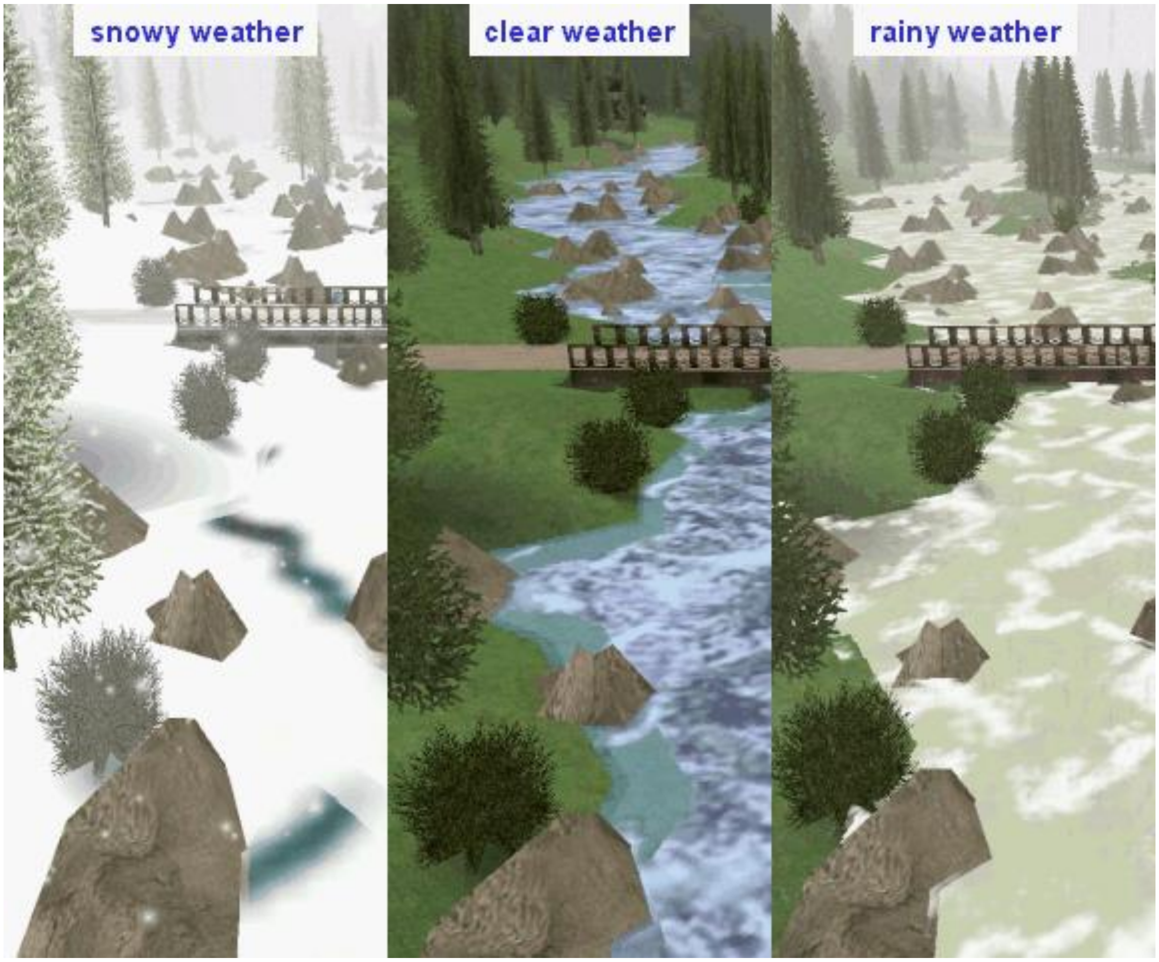
NOTE: There is a serious problem with modeling water flow in rivers within MSTS. Since water can only be made to flow in the same direction everywhere, it is not possible to make it follow the curves of rivers. And it is not possible to make it flow with the different directions of different rivers. I know of no solution to this problem.

The next figure illustrates the difference between the MSTS default water and a modified style of water.  The views are from my mountain route with rivers (Albula Line at left, and the same spot in its version 2 at right).  The default water shown at left was augmented with a few white/green transfers to give the impression of "white" water;  but the modified water at right includes white water much more realistically.  In addition, not visible in the figure, is motion that makes the stream come alive;  this is described in more detail below.


**All the above changes can be made to depend on the weather and season**, allowing a total of 12 different forms of water (see section 3.15.1).

**The weather dependence is illustrated in the next figure**:  the three views of the same spot were taken from the same route (Albula Line 2) simply by selecting a different weather/season combination before running a train (the user need not exchange files or do anything more than choose the weather and season when starting exploring;  in an activity the selection is even automatic).

These water appearances are also included in my version of First Route, accompanying this Guide, even though they may not fit the London climate well!




For clear weather (especially in the summer) I made the water rather transparent, with a rocky bottom, white water on the surface and a blue reflection of the sky;  to give some feeling of motion, I made the white water on the surface wiggle back and forth (remember that waves on a river don't move relative to the terrain:  they are "glued" to rocks, etc.);  I also made the rocky bottom wiggle very slightly (because that is what the eye perceives when you look through moving water);  and I added a small-height wave that makes the water move up and down a little bit.

For snowy weather, I lowered the water level (a river carries less water in winter), froze its surface and covered it with snow, leaving cracks to see the dark bottom and water motion;  I also stopped the waviness and motion of the frozen top surface, while removing the sky reflection.

And for rainy weather I prepared a higher water level, with a muddy color, more white water, shallower visibility, a faster motion, increased waviness, and again no reflection of the sky.

You can also **make these water changes visible in the RE**, but only for one type of weather at a time (see the illustrations further down, which also show how these water appearances were created).

**The water that you define will apply to the entire route.** As far as I know, you can't make it look different in different parts of the route. This means that you cannot make one type of water for lakes, another for rivers and a third for the sea.

In MSTS, **water is composed of 4 layers by default**; one of these, a sky-reflecting layer, is not active by default, but can be activated. The top two visible layers are semi-transparent, dark blue, featureless and (apparently) motionless, while the third is dark brown and opaque. You can make any layer opaque or semi-transparent or fully transparent. Each layer can move (flow) independently of the others in any direction and at any speed; it can also be made to "jump" stepwise (the purpose of this could be to move white-topped waves a short distance, but I use this to make water wiggle around). You can also change the number of water layers, if needed (for example, you could remove layers to reduce the computing load of MSTS).

**The water is controlled through the 4 files editor.env, sun.env, rain.env, and snow.env** contained in the route's ENVFILES folder.

**The 3 files sun.env, rain.env, and snow.env control what the user sees when running in MSTS**, depending on whether MSTS runs in clear, rainy or snowy weather. You will need to change these files to change the appearance of water in MSTS: by applying different changes to the 3 different files, you will get 3 different types of water depending on weather. If you also make season-dependent versions of these files, you can prepare up to 12 different types of water (see section 3.15.1).

**The file editor.env controls what you see in the Route Editor**, but does not affect what the user sees when running MSTS.

You may change the contents of editor.env in the same way that you change the files sun.env, rain.env, and snow.env. Then the RE will show the water type that you wish: water for clear, rainy or snowy weather, but only one of these at a time. This can be convenient for testing purposes: you can make quick changes to the editor.env file and view the result in the RE, just by reloading the route without exiting from the RE. (Note that editor.env does not affect the terrain or object textures in the RE; for example, if you specify "snowy" water in editor.env, the terrain and objects in the RE will not look snowy: only the water will look "snowy", as you can see in the figures below.)

The four *.env files point at texture files found in the ENVFILES\TEXTURES folder: these **texture files, of type *.ace, contain all the required graphics to draw the water**. The following 4 graphics files are important - they correspond to the 4 different water layers mentioned earlier (you may change their names as needed):

- **waterbot.ace** draws the bottom water layer: by default, this layer is opaque and lies at a depth of -1 m (the layer depth can be changed);
- **watermid.ace** draws the middle layer: by default, it is semi-transparent and lies at a depth of -0.5 m (this layer depth can also be changed);

- **watertop.ace** draws the top layer: by default, it is semi-transparent and lies at a depth of 0 m, namely at the local water level defined for each tile (this layer depth can also be changed);
- **test.ace** can draw the reflected sky on an additional layer (which can be made to coincide with the top layer), but does not do so by default: it can be modified to show the reflected sky, as discussed in the next section.

Note that you may change the number of layers used to define water to something else than 3 or 4; it can be as low as 1, and at least as high as 4 (I don't know if there is an upper limit).

Each *.ace file can contain two images: the water texture itself, and an "alpha" image that defines the transparency of the texture, pixel by pixel. Thus, a set of 4 water layers is defined by up to 8 different images, each of which can be made to vary with the weather: this gives a lot of freedom for creating water styles! For the deepest layer to be visible, all layers above the deepest one must be partially transparent. The deepest layer does not have to be opaque: if it is partially transparent, the user can see deeper into the water, down to the terrain, including any objects placed there (rocks, fish, sunken ships, etc.); but the visibility range into the water will be given by the "fog" in the air, because MSTS does not have a separate visibility range within water.

By default, the sun.env, rain.env, and snow.env files refer to the same four *.ace files for water: therefore, the water looks the same in all weather conditions. By creating different version of these *.ace files, you can produce different water for the 3 weather conditions. For example, for water in rainy weather you could create files waterbotr.ace, watermidr.ace, watertopr.ace and testr.ace (with the added letter "r" meaning rain), and, similarly, for water in snowy weather you could create files waterbots.ace, watermids.ace, watertops.ace and tests.ace (with the added letter "s" meaning snow). Then you would have to modify the rain.env, and snow.env files so that they refer to these new file names.

### 3.15.3c The water details in the *.env files

To explain how to modify the appearance of water, we will continue **looking at a default version of the file rain.env**. Exactly the same can be done with the sun.env and snow.env files, since the water sections of these files are identical by default.

CAUTION: Before changing these files in your route, **make sure to back up the original versions**, because it is easy to make mistakes with all those opening and closing parentheses: if you mismatch even one parenthesis, the RE and MSTS will crash!

Below, **the numbers in red are those that you may change**; I don't know the meaning of the others, so I recommend great caution with those.

The section that interests us here begins as follows:

```
world_water
(
  world_water_terrain_patch_map ( 256 Wsib-W.raw )
```

```
world_water_wave_height ( 0 )
world_water_wave_speed ( 0 )
```

**The last two lines define a wave**, with a height given in meters: that is the height of a wave crest above the average water level. The wave "speed" is really the wave frequency, given as number of oscillations per 2-second time interval. A "speed" of 1 makes the wave rise and then fall at a fixed point in 2 / 1 = 2 seconds before starting its next full oscillation cycle, while a "speed" of 0.1 makes it rise and fall in 2 / 0.1 = 20 sec; a speed of 0 gives an infinite result (2 / 0 = infinity), meaning no motion. Thus, the higher the "speed", the faster the water moves up and down. I estimate the "wave length" to be about 100-200 m, which cannot be changed, as far as I know: that is the distance between one wave crest and the next wave crest.

(The speed is not given as km/h or anything like that, because this wave is actually a bit unusual: it is not a "traveling" wave like that created by a ship, for instance , but a "standing" wave like that in a violin string. This means that you will find points where the water goes up and down as you expect, but other points where the water does not go up and down at all. Incidentally, the wave crests form straight lines that are oriented from NW to SE.)

NOTE: If a wave motion is defined, the different water levels given below will follow that motion together, as one single wave.

Next, the sun.env file contains the lines

```
world_water_layers
(
  3
```

**The 3 means that 3 water layers are defined next**: you should see three sections now, each starting with `world_water_layer`. Actually you will find 4 such sections: only the first 3 are used (they correspond to the 3 visible layers), while the 4th is not used (it is the layer that should reflect the sky). You can change this number 3, and add or delete whole sections starting and ending with the lines

```
world_water_layer
(
   ..
)
```

(Be very careful to catch the right closing parenthesis!)

NOTE: You may lower the number 3 without deleting any sections: these sections will simply be ignored, as with the 4th section mentioned here.

**The first** `world_water_layer` **section** starts like this and **defines the bottom water layer** (as you can tell from the file name `waterbot.ace` found a bit later in this section):

```
world_water_layer
(
  world_water_layer_height ( -1 )
```

**The layer height is set to -1 m here, and can be changed as desired.** This height is really what makes this layer the bottom layer: as we will see, the other layers are set higher than this one. The fact that the name `waterbot.ace` contains "`bot`" for "bottom" is not important: you could use any file names you wish, such as "`firstlayer.ace`", if you actually have a file called `firstlayer.ace` in the ENVFILES\TEXTURES folder.

The layer height is measured relative to the water level defined in each tile. If the water level was defined to be sloping, this layer (and the other water layers defined later) will slope accordingly, following the general sloping water level.

So this bottom water layer, at height -1 m, will be 1 m below the water level that you have defined in each tile.

Next come some mysterious-looking lines. We only need to be concerned with two of them:

```
world_anim_shader
(
  world_anim_shader_frames
  (
    1
    world_anim_shader_frame
    (
      world_anim_shader_frame_uvscroll ( 0 0 )
      world_anim_shader_frame_uvtiles ( 8 8 )
    )
  )
```

**The line with** `frame_uvscroll ( 0 0 )` **controls the speed of flow of the texture drawn on this (bottom) layer.** The first number (0 here) is the speed component in the westerly direction, while the second number (also 0 here) is the speed component in the southerly direction. A speed of 1 is about 100 km/h, much too fast! A more reasonable speed would be 0.02 to 0.1, meaning 2 to 10 km/h. Use a negative sign to reverse the speed component, for flow to the east and/or north.

**The line with** `frame_uvtiles ( 8 8 )` **tells** MSTS and RE **that each patch of 128x128 m should contain 8x8 copies of the texture** defined in the texture file given below (`waterbot.ace`). You can choose any two whole numbers here, and even non-whole numbers; they also do not have to be equal to each other, as in `( 4 3 )` or `(0.5 0.75 )`.

The following lines (see below) tell **which texture file to use in this layer:** `waterbot.ace` in this case. You may freely change this file name, as long as you provide a correspondingly named file in the ENVFILES\TEXTURES folder. In my experience, this file may have either 128x128 or 256x256 pixels; probably other sizes (powers of 2 like 512x512) are also allowed.

```
world_shader ( TexDiff
  terrain_texslots ( 1
  terrain_texslot ( waterbot.ace 1 0 )
           )
```

```
          terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
    )
  )
```

The number 1 following `waterbot.ace` can be changed to create a special tiling effect: if you change this 1 to 2, you get mirrored versions of the texture, as illustrated for the sky in section 3.15.2b. This reduces the seams that may occur at the texture edges with simple repetition.

The command `terrain_uvcalc ( 1 0 0 0 )` must contain some secret effects that remain to be discovered…

This completes the definition of the bottom layer. **Next comes the second, middle layer:**

```
  world_water_layer
  (
    world_water_layer_height ( -0.5 )
    world_anim_shader
    (
      world_anim_shader_frames
      (
        1
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvscroll ( 0 -0.008 )
          world_anim_shader_frame_uvtiles ( 8 8 )
        )
      )
      world_shader ( BlendATexDiff
        terrain_texslots ( 1
          terrain_texslot ( watermid.ace 1 0 )
        )
        terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
    )
  )
```

You see that its height is -0.5 m below the predefined water level. And it flows at about 0.008 x 100 = 0.8 km/h to the north (because of the negative sign). However, this flow is invisible, because the texture used by MSTS by default has no structure. This texture, `watermid.ace`, is copied 8x8 times in each patch. (You don't have to use the same number of copies in both layers: you could use 5x5 copies in this layer, for instance; this may even help break the periodic repetition of the water patterns that looks artificial on large water surfaces. However, each 128m x128m patch will always be identical to all the others, except for the sky reflection, of which one copy fills a complete tile.)

[NEW SINCE V2]
**FIX FOR FLASHING WATER:** You may have seen water flash in MSTS: it can oscillate in color between blue and white in certain camera views, depending in particular on view direction (this may be a strong or weak effect, depending perhaps on hardware and/or graphics driver). Although this was thought to be related to dynamic track or complex objects, it has been found by Jeff Bush to result from the use of the `BlendATexDiff` shader appearing in the `world_shader` command listed above and in the next two water layers (if present). Replacing `BlendATexDiff` by `BlendATex` in these water layers stops the flashing (do <u>not</u> apply this change to the `world_sky` or `world_precipitation` sections of the *.env files). This fix must be applied to each *.env file used by your route; it can also be applied to the default MSTS routes (first check which *.env files a route uses by looking in the route's *.trk file). Unfortunately, this introduces another, less disturbing bug: water and terrain glow at night.

**The third layer** to be defined **is the top layer**, shown next:

```
world_water_layer
(
  world_water_layer_height ( 0 )
  world_anim_shader
  (
    world_anim_shader_framelen ( 0.6 )
    world_anim_shader_frames
    (
      4
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0 0 4 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.5 0 4.5 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.75 0 4.75 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 1 0 5 4 )
      )
    )
    world_shader ( BlendATexDiff
      terrain_texslots ( 1
        terrain_texslot ( WaterTop.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
```

This layer is at 0 m height, right at the predefined water level. And it uses the texture file `WaterTop.ace`. Note that by default `WaterTop.ace` is a uniform color with no structure, so any motion will be invisible. **If you want to see motion, you will have to supply a texture that has structure in it.**

However, there are some significant differences here compared to the first two layers. First, you have a line with `world_anim_shader_framelen ( 0.6 )`. Second we see 4 subsections of the type

```
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0 0 4 4 )
)
```

**These four subsections cause the texture to jump along the water surface (rather than flow smoothly)**, making two jumps to the west, and after a short pause a jump back to the start point. The time between jumps is given by `framelen ( 0.6 )`, where the unit is seconds (thus 0.6 second by default).

I guess this "water jumping" option was designed to allow more complex movements than a simple continuous flow. One possibility is to create crashing white wave tops: these would move a few steps forward and then start again from the initial point. I used this option in the water illustrated above and below to make white water wiggle around (see the example in section 3.15.3d): you can produce as many jumps as you wish (not just the 4 shown by default): increase that 4 to whatever number of positions you wish to produce, and include the corresponding number of sections of the type

```
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0 0 4 4 )
)
```

with different numbers in the parentheses `( 0 0 4 4 )`.

So what do those four numbers `( 0 0 4 4 )` mean?

The first 4 represents the number of copies of the texture (`WaterTop.ace`) within a 128x128 patch along the west/east direction; the second 4 is the same for the north/south direction. A special feature here is that these numbers need not be whole numbers, as you can see in the default file: they can be 4.5, 4.75, 4.01, etc., so that you can make small changes from one position to the next, giving small jumps. Thus, a larger value here will compress the texture.

By contrast, the first 0 is a stretch factor of the texture in the west/east direction; and the second 0 is a stretch factor in the north/south direction. What is a bit odd is that a value of 0 means a stretch factor of 1 + 0 = 1 (no stretch), while a value of 1 gives a stretch factor of 1 + 1 = 2 (doubling the texture length).

The exact function of these numbers is not clear to me, but what you need to do is follow the example given in the default file: if you do that, the texture of this layer will jump to the positions you define. In section 3.15.3d, you can see other examples of the use of this option to create jumpy motion.

One more point on this option: the default values (like 0, 0.5, 0.75, 1, and 4, 4.5, 4.75, 5) may create excessively large and unnatural jumps for water (this will depend on the kind of texture you use): for more continuous motion I recommend smaller differences, like 0, 0.01, 0.03, etc. and 4, 4.01, 4.03, etc. (you may also use 4, 4.02, 4.05, etc.: such irregular variations will create more "randomness" in the motion).

[**If you wish to replace the jumpy motion by a smooth "flow", do the following:** replace the whole section

```
4
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0 0 4 4 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0.5 0 4.5 4 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 0.75 0 4.75 4 )
)
world_anim_shader_frame
(
  world_anim_shader_frame_uvstamp ( 1 0 5 4 )
)
```

by a section like that of the second (middle) layer:

```
1
world_anim_shader_frame
(
  world_anim_shader_frame_uvscroll ( 0 -0.008 )
  world_anim_shader_frame_uvtiles ( 8 8 )
)
```

and adjust the red numbers as desired. Also delete the line containing `framelen ( 0.6 )`. Now this third (top) layer will flow smoothly.]

Continuing in the rain.env file, we find **another** `world_water_layer` **subsection**, the 4th:

```
world_water_layer
(
  world_water_layer_height ( 0 )
  world_water_layer_sky_reflection ( 0 )
  world_anim_shader
  (
```

```
          world_anim_shader_frames
          (
            1
            world_anim_shader_frame
            (
              world_anim_shader_frame_uvscroll ( 0 0.1 )
            )
          )
          world_shader ( BlendATexDiff
            terrain_texslots ( 1
              terrain_texslot ( test.ace 1 0 )
            )
            terrain_uvcalcs ( 1
              terrain_uvcalc ( 1 0 0 0 )
            )
          )
        )
      )
    )
  )
```

Because of the number 3 that appeared right before the first `world_water_layer` subsection, this 4th subsection is ignored. As the words `sky_reflection` indicate, **this layer can be made to reflect the sky, with some extra work**.

The layer height is 0 m, coinciding with the top visible layer (you can raise it above the top layer, but if you drop it below the top layer, it loses its reflection power). The 0 in `sky_reflection ( 0 )` allows motion of the texture, as defined by the `uvscroll` of the top sky layer; setting `sky_reflection` to 1 freezes that motion. The texture used as reflection is `test.ace`.

To make this layer reflect the sky, you must first activate this layer, by changing the number 3 mentioned above to 4, so this 4th section is not ignored. If you do this and then load the route into MSTS, you will see your water covered with a uniform and opaque blue color: it shows no water depth, and it shows no cloud reflections! Why? First, the default texture `test.ace` has zero transparency, so it does not allow seeing into the water. Second, that texture is solid and uniform blue: it includes no clouds or other structure, so its reflection is uniform and uninteresting; instead, this texture should be a carbon copy of the sky texture (rainsky.ace or snowsky.ace in this case), rather than show a uniform blue color. The principle here is that a "reflection" is actually another water layer onto which a copy of the sky texture is drawn. So, to obtain reflections, you will need to copy the rainsky.ace or snowsky.ace file over the test.ace file. But, in so doing you will discover that the rainsky.ace and snowsky.ace textures also have no transparency (since they are opaque), although they do show clouds: so you must add transparency for realism, but you need to do this in new files with new file names to avoid making the sky ceiling transparent! (Appendix H discusses manipulating *.ace files with alpha channels to control transparency.)

[NEW SINCE V2] WARNING: For some users, adding sky reflections to water can cause a "Train World Initialization Failure" when starting the route: their systems (perhaps older video cards) can't handle the reflecting water layers. If you distribute your route with reflecting water layers, you should be prepared for this possibility: you should tell those users to not use the reflective water layer, by using a water layer count of 3 instead of 4; or, better, you can supply replacement *.env files in which you have used a water layer count of 3 instead of 4.

When you add a reflecting layer, you will notice a serious drop in the daytime illumination intensity of the terrain and objects, which is unrealistic. To compensate for this, you can increase the "ambient" light intensity by changing the gray level in the line

```
world_sky_satellite_amb_high_colour ( ff808080 )
```

for the sun; you may choose `ffffffff` for a clear sunny day, and a darker gray level for cloudy, rainy or snowy skies (for example `ffdddddd` for a light day, `ffaaaaaa` for a medium dark day, or keep `ff808080` for a very dark day). You may similarly increase the "ambient" intensity at sunrise (`amb_rise_colour`) and sunset (`amb_set_colour`).

I find that a sky reflection can be difficult to see in RE or MSTS: the default clear-day sky of MSTS has only a few clouds, and they don't often come into view in reflection if you stand still. But if you move around (especially in RE), you should notice cloud reflections passing over your water surfaces. It also helps to use the terrain wireframe view if you have a lot of terrain above water (press W in RE). To see more varied reflections, you should have a sky that contains more clouds.

Note that the `test.ace` texture file will fill a complete tile: there will be only one copy of `test.ace` in one entire tile of 16x16 patches (2048 x 2048 m).

This completes the part of the sun.env file that defines water. The remainder of this file deals with winds.

For water textures suitable for wavy water on lakes and seas, I recommend the beautiful tropical textures by Ron Malicki, available at Train-Sim.com; other textures are available at [http://www.aaabackgrounds.com/textures/water/](http://www.aaabackgrounds.com/textures/water/) and other web sites. Otherwise, you can try make them yourself.

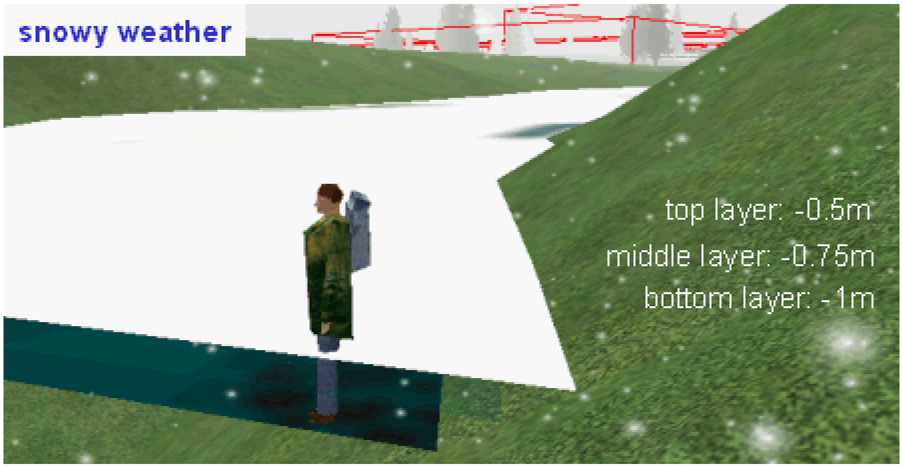
### 3.15.3d An example of changed water

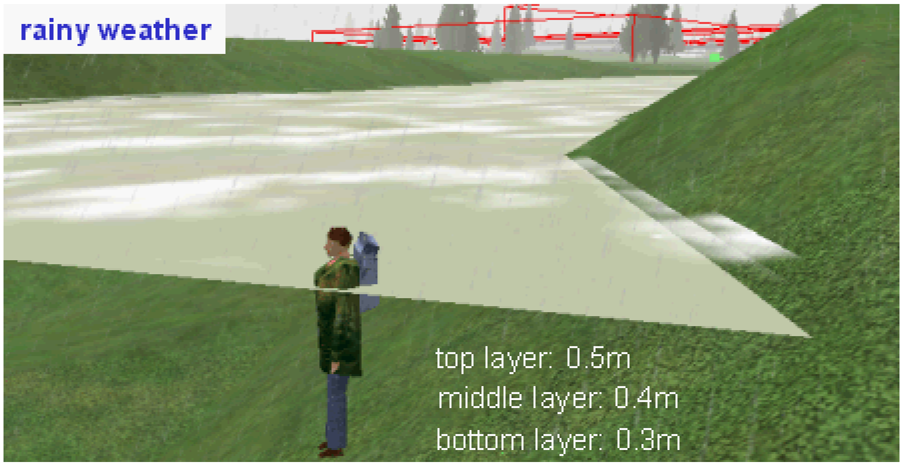
Let me explain **how to create the three water appearances shown in the above figures** for a mountain river (designed for my route Albula Line 2).

For a route with winding rivers, you face the problem of **modeling the down-slope flow** of water, because the flow direction will have to be the same everywhere in the route: whatever the flow direction that I choose, in some places such a flow would move upstream, and almost everywhere it will flow across the river: very unrealistic! As mentioned before, I don't know a good solution to this. The way I handled this for the illustrations given here is to give a "hint" of motion, a suggestion that "something is moving", rather than to model actual water rushing downstream. I did this in three ways: first I made the water move up and down a little bit by using the wave option for water; second, I allow white patches at the top of the water to wiggle around relatively fast, to suggest that this white water is constantly changing its position a little bit around a fixed point;  and third, I let another white water layer "flow" very slowly, so as to gradually change the overall combined pattern of the different water layers, and thereby "randomize" the motion.

Another issue that you face is the periodic repetition of textures, especially on larger water surfaces:  that can look very artificial.  One way to minimize that is to use different "unrelated" values in `world_anim_shader_frame_uvtiles ( 8 8 )` for the different layers:  for example, I used ( 8 8 ) for the top layer, ( 3 3 ) for the middle layer and ( 5 5 ) for the bottom layer to "confuse" the viewer and break up the periodic character of the textures:  this way, the overall repeating pattern is the 128x128 m patch rather than the individual textures;  if I had used ( 8 8 ) for all layers the repeat period would be only 1/8 of a patch, or 16x16 m.  But for large open water surfaces, you are stuck with the 128x128 m patch as a repeating pattern.

The next three illustrations (taken in the RE) show the cross-section of a river (at the edge of a "wet" patch which has water toggled on) under three different weather conditions:  they use the three types of weather-dependent water shown in the figures at the beginning of section 3.15.3b.  The female hitchhiker will help us get a feeling for heights:  her feet in each picture are at a constant level of -1 m, which is 1 m below the local water level defined in this tile (the water level is in fact gently sloping , but this is not visible in the figures).




Let's first look at the **clear-weather condition**, illustrated in the first of these three figures. Here I used 4 layers, all visible. The details are the following (you can also see how they are coded in the listing below, while the textures are shown further down):

- the water is given a wave motion of 0.1 m height (not visible in the figures, because of the long wavelength of about 200 m), with a speed of 0.05 (a 40 second oscillation period);

- the bottom layer has a rocky texture and is opaque; it wiggles a little bit (to suggest the changing bending of light rays through the wavy water surface), as defined by 3 small jumps, with a time step of 0.2 seconds between jumps;

- the middle layer, 0.1 m below the surface, contains white water that flows very slowly (0.008 is less than 1 km/h), to produce a gradual change of the overall white water pattern;

- for the top layer, at height 0 m, I used an irregular wiggling motion, defined by 8 small jumps, with a time step of 0.2 seconds between jumps; it also carries a white water pattern;

- the reflective layer is positioned 0.1 m above the top layer in the illustration above, to make it visible as a separate layer in the figure; but, in fact, I gave it a height of 0 in the route; it shows blue because it reflects the blue sky; its speed is set by the `uvscroll` statement of the top sky layer, not by the `uvscroll` statement of the reflective layer! I called the texture used for the reflective water layer dayskytransp.ace, because it includes a copy of the (opaque) daysky.ace texture, together with a transparency channel to let us see into the water.

Here is the water section of the sun.env file that produced this clear-weather water, with the important choices marked in red:

```
world_water
(
  world_water_terrain_patch_map ( 256 Wsib-W.raw )
  world_water_wave_height ( 0.1 )
  world_water_wave_speed ( 0.05 )
  world_water_layers
```

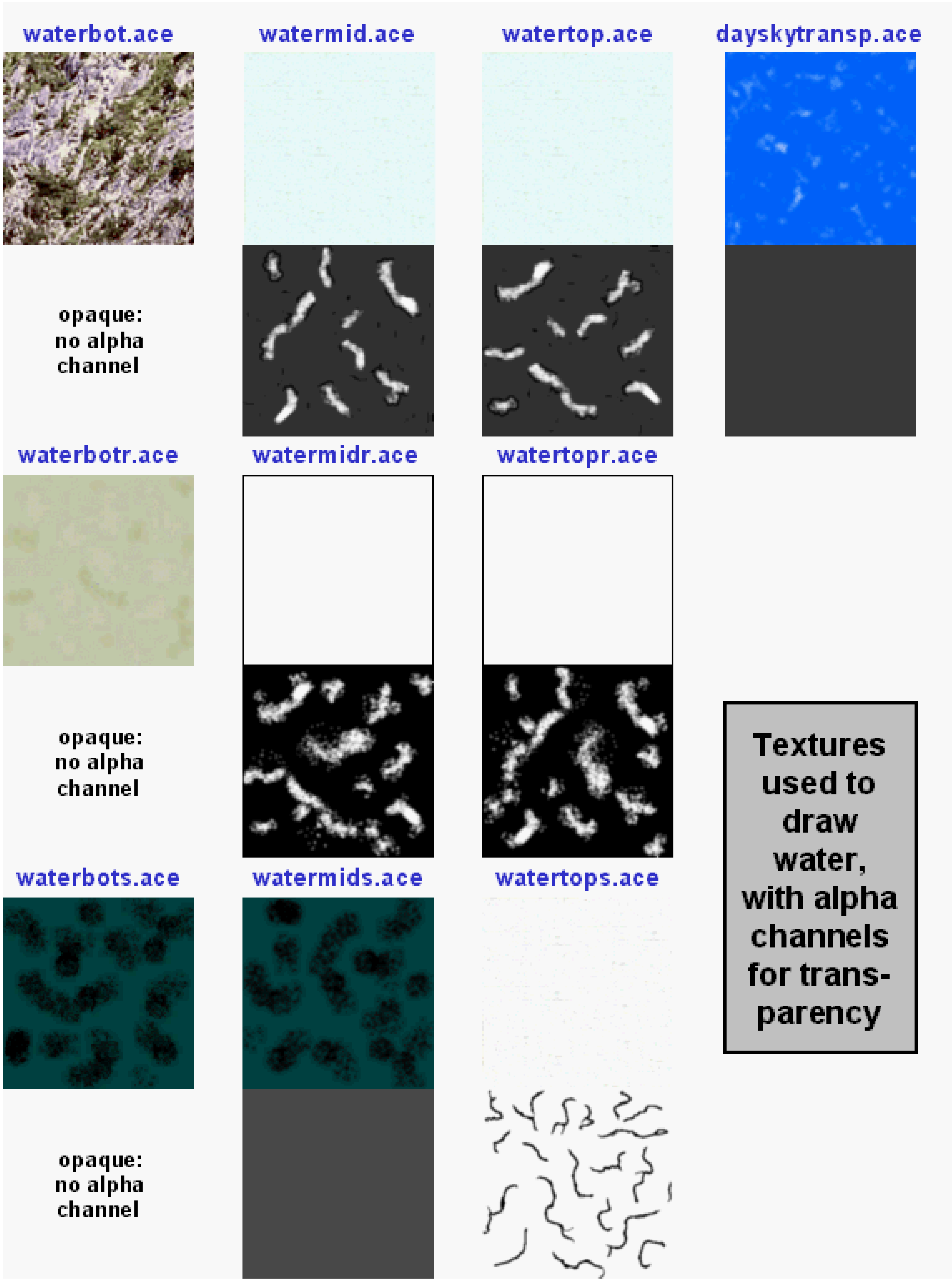
```
(
  4
  world_water_layer
  (
    world_water_layer_height ( -0.5 )
    world_anim_shader
    (
      world_anim_shader_framelen ( 0.2 )
      world_anim_shader_frames
      (
        3
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( 0 0 5 5 )
        )
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( -0.001 0 4.9995 5 )
        )
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( -0.001 -0.001 4.9995 5.0005 )
        )
      )
      world_shader ( TexDiff
        terrain_texslots ( 1
          terrain_texslot ( waterbot.ace 1 0 )
        )
        terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
    )
  )

  world_water_layer
  (
    world_water_layer_height ( -0.1 )
    world_anim_shader
    (
      world_anim_shader_frames
      (
        1
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvscroll ( 0 0.008 )
          world_anim_shader_frame_uvtiles ( 3 3 )
        )
      )
      world_shader ( BlendATexDiff
        terrain_texslots ( 1
          terrain_texslot ( watermid.ace 1 0 )
        )
        terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
```

```
  )
)

world_water_layer
(
  world_water_layer_height ( 0 )
  world_anim_shader
  (
    world_anim_shader_framelen ( 0.2 )
    world_anim_shader_frames
    (
      8
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0 0 4 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.002 0 4.001 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.003 0.002 4.002 4.002 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.003 0.004 4.002 4.002 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.002 0.001 4.001 4.001 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.004 -0.002 4.002 3.998 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.006 0 4.003 4 )
      )
      world_anim_shader_frame
      (
        world_anim_shader_frame_uvstamp ( 0.003 0.002 4.002 4.002 )
      )
    )
    world_shader ( BlendATexDiff
      terrain_texslots ( 1
        terrain_texslot ( watertop.ace 1 0 )
      )
      terrain_uvcalcs ( 1
        terrain_uvcalc ( 1 0 0 0 )
      )
    )
  )
)
world_water_layer
(
```

```
        world_water_layer_height ( 0 )
        world_water_layer_sky_reflection ( 0 )
        world_anim_shader
        (
          world_anim_shader_frames
          (
            1
            world_anim_shader_frame
            (
              world_anim_shader_frame_uvscroll ( 0 0.02 )
            )
          )
          world_shader ( BlendATexDiff
            terrain_texslots ( 1
              terrain_texslot ( dayskytransp.ace 1 0 )
            )
            terrain_uvcalcs ( 1
              terrain_uvcalc ( 1 0 0 0 )
            )
          )
        )
      )
    )
  )
```

The rainy and snowy version of my water are made in much the same way: they just use different variations of the choices made for the clear-weather version.

The next figure shows all the textures used to make the 3 versions of water illustrated here. The alpha channels that control transparency are shown below each texture. Remember that transparency is greatest when the alpha channel is darkest (black gives full transparency, and white is opaque).




The first row of textures and alpha channels corresponds to water designed for clear weather (described in detail earlier in this section). The texture for waterbot.ace was taken from a photograph of rocks and shrubs used as a transfer to be placed on terrain: it also works well as a river bottom, except that opposite edges don't match perfectly, giving abrupt seams. The textures for watermid.ace and watertop.ace are bluish versions of the texture of watertops.ace: they give bluish white water near the water surface. White water patches are obtained by means of their transparency: I drew their alpha channels with Microsoft Paint (smoothing them with Microsoft Photo Editor), and rotated one with respect to the other to create more variety when superposed on top of each other. The texture in dayskytransp.ace is the same as that in daysky.ace by MSTS, used to draw the default clear sky; I have only added a simple alpha channel to make it partly transparent.

The second row corresponds to water in rainy weather. I drew the texture for waterbotr.ace with Paint and Photo Editor to look like muddy water: it serves as the bottom layer. The textures for watermidr.ace and watertopr.ace are simply blank: they represent white water that floats just above the muddy surface, visible only where their alpha channel is light.

The third row corresponds to water in snowy weather. I drew the texture for waterbots.ace and watermids.ace to represent dark water under snow-covered ice; watermids.ace is made partly transparent. Both of these are visible through cracks in the top layer, which is white due to snow; the transparent cracks are drawn as black lines on the alpha channel of watertops.ace. The texture of watertops.ace is the same as the terrain texture that I use to produce deep snow in the route, so the snow-covered rivers look the same as the snow-covered terrain around it (this deep-snow texture was made by Ruben Geerling, and used with permission).

[NEW SINCE V1.106]

### 3.15.4 PRECIPITATION

Precipitation includes both rain and snow. The main aspects that you can control are the type of precipitation (rain vs. snow), and the density of rain or snow. You can also change the texture for either rain or snow. To some extent, you can also control the vertical speed of rain or snowflakes.

These aspects are controlled through a section in the *.env files that starts like this (we again use a default rain.env as sample):

```
world_precipitation
(
  world_precipitation_type ( world_precipitation_type_rain )
  world_precipitation_density ( 2048 )
  world_precipitation_speed ( 40 )
```

In the line with `world_precipitation_type`, the `rain` is replaced by `snow` to get snow instead of rain.

The `world_precipitation_density` sets the number of raindrops (or snowflakes) that you see. Perhaps a power of 2 (like 2048, 1024 or 4096) is more efficient than a random number.

The `world_precipitation_speed` controls how fast the raindrops (or snowflakes) drop vertically. However, for rain this is coordinated closely with other numbers given below, so changing this speed may give strange results. Snow seems to be more tolerant of changes in this speed.

Next comes a section that controls the motion of the raindrops and snowflakes. Although this looks very similar to a section in the water definition, it is not clear now how this section affects the rain or snow.

```
    world_anim_shader
    (
      world_anim_shader_framelen ( 0.5 )
      world_anim_shader_frames
      (
        4
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( 0 0.5 0.5 0 )
        )
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( 0.5 0 1 0.5 )
        )
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( 0 0.5 0.5 1 )
        )
        world_anim_shader_frame
        (
          world_anim_shader_frame_uvstamp ( 0.5 0.5 1 1 )
        )
      )
      world_shader ( BlendATexDiff
        terrain_texslots ( 1
        terrain_texslot ( rain.ace 1 0 )
        )
        terrain_uvcalcs ( 1
          terrain_uvcalc ( 1 0 0 0 )
        )
      )
    )
  )
```

At the end of the above section you see the specification of the texture file rain.ace (it would be snow.ace for snow). This file should be partly transparent. It is used for the raindrops or snowflakes that you see falling from the sky, and also for the drops that form on the train's windows or the camera's lens.

[NEW SINCE V1.106]

### 3.15.5 WIND

The main effect of the wind in MSTS is to change the angle at which rain and snow fall. Rain and snow will blow more sideways with a stronger wind; and more turbulence will cause more swirling of the snow flakes and perhaps also of rain drops.

NOTE: I find the effect of wind on rain and snow to be unreliable: how the wind functions is not entirely clear!

The cloud and wave motions are set independently of the winds. Apparently, wind has no effect on the smoke of steam locomotives.

These wind properties are controlled through the last section in the *.env files (we again use a default rain.env as sample). It starts like this:

```
world_wind
(
  _world_wind_layers
  (
    2
    world_wind_layer
    (
      world_wind_layer_maxheight ( 200 )
      world_wind_layer_direction ( 1 0 0 )
      world_wind_layer_speed ( 3 )
      world_wind_layer_turbulencep ( 0 )
    )
```

The quantities I understand here are the (horizontal) speed and turbulence.

I am not sure of the effect of `maxheight`, but it may relate to the altitude above sea level, while changing the `direction` had no effect in my tests.

Another wind layer follows, which may affect only the higher altitudes:

```
    world_wind_layer
    (
      world_wind_layer_maxheight ( 100000 )
      world_wind_layer_direction ( 0 0 0 )
      world_wind_layer_speed ( 0 )
      world_wind_layer_turbulencep ( 0 )
    )
  )
```

Next comes another pair of wind layers (which are absent from default snow.env and sun.env files), and whose function is not clear:

```
    world_wind_layers
    (
                    1
      world_wind_layer
      (
        world_wind_layer_maxheight ( 100000 )
        world_wind_layer_direction ( 1 0 0 )
        world_wind_layer_speed ( 3 )
        world_wind_layer_turbulencep ( 0.2 )
      )
      world_wind_layer
      (
        world_wind_layer_maxheight ( 100000 )
        world_wind_layer_direction ( 0 -1 0 )
        world_wind_layer_speed ( 0 )
        world_wind_layer_turbulencep ( 0.2 )
      )
    )
  )
```

[NEW SINCE V1.106]

### 3.15.6 ENVIRONMENTAL EXAMPLES IN First Route

Included in my latest version of First Route (**`First Route MV3`**), packaged with this Guide, are 12 variations of the environment, selectable in MSTS by any combination of season and weather. (But, if you load **`First Route MV3`** into the Route Editor, you will only see the default environment for a clear summer day.)

NOTE: These environments were designed for a mountain route, so the water is muddy with the spring runoff, and frozen in winter, as illustrated in section 3.15.3b. They are not ideal for London! This illustrates that the environment (especially the water) should be tailored to your route.

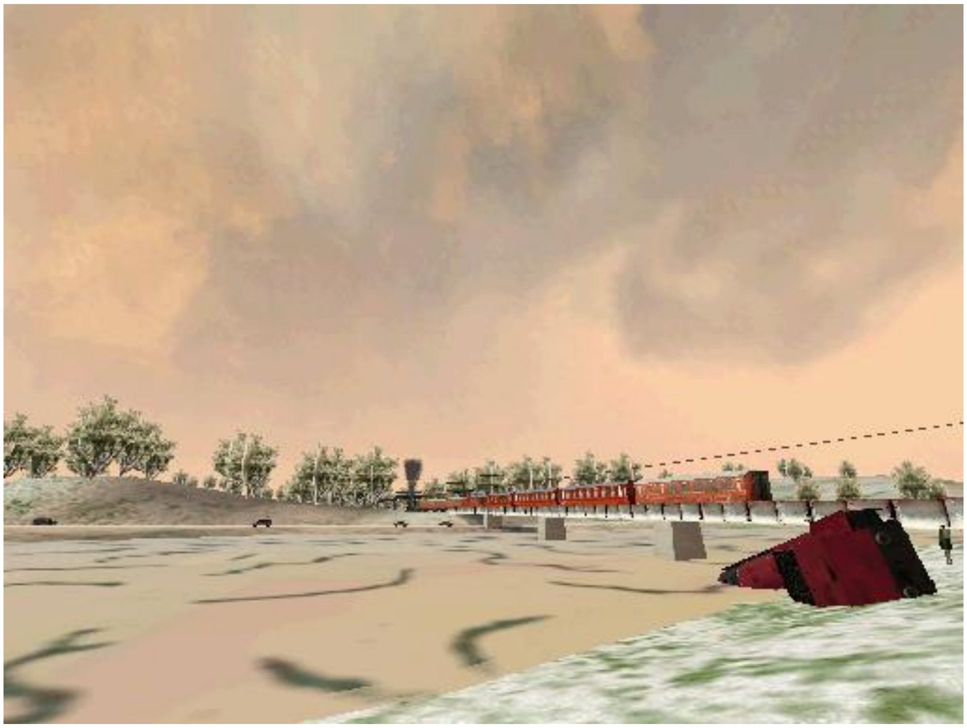
Here are a few explanations of what environment you get in **`First Route MV3`** depending on season and weather:

**Spring / Clear**: Dense "ground fog" is matched by low white clouds as "sky fog", against a high "sheep-like" sky. Here as in many other cases, planes fly against that same high cloud layer: the planes come from a simple two-dimensional drawing on an otherwise transparent intermediate sky layer, moving relatively fast. The water is muddy, with wiggling "white water" on top to mimic motion; its level its relatively high.

**Summer / Clear**: Planes fly against a background of some white clouds, without ground fog. The water is clear, showing a rocky bottom, and reflecting the cloudy sky.

**Autumn / Clear**: Light "ground fog" does not affect the dark clouds against a blue sky. The clear water is relatively low, and reflecting.

**Winter / Clear**: As seen in the next picture from First Route, a semi-transparent version of the earlier few white clouds blocks part of a mixed higher cloud layer. The water is low and frozen, with sky reflections and transparent cracks showing water motion below.



**Spring / Rain**: Heavy rain causes flooding around the main river. Lightning is visible both as lightning streaks and as big diffuse flashes: both are drawn as partly transparent sky layers that move very fast compared to the frame rate (so they jump more than the sky's width from one frame to the next). There is light "ground fog" and a dark higher sky.

**Summer / Rain**: As shown in the next picture (with an airplane passing high overhead), a rainbow is part of an opaque sky layer that obscures sun and moon, in light rain.

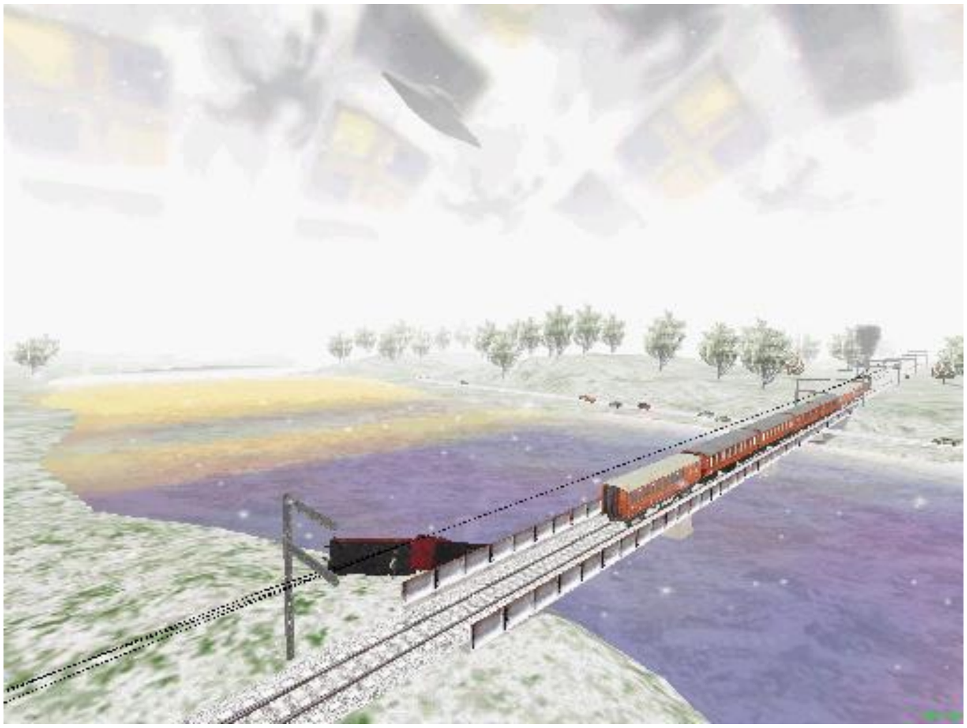
**Autumn / Rain**: Lightning is now drawn as two different lightning streaks on the same texture, and as a wider flash on a separate sky layer. It shows against an opaque dark sky that blocks sun and moon, under medium-dense rain, as illustrated next.

**Winter / Rain**: Dense "ground fog" and matching "sky fog" block much of the sky, in light rain. The water is frozen.

**Spring / Snow**: A dense cover of dark clouds accompanies medium-dense "ground fog", with medium snowfall. The water is muddy and high.

**Summer / Snow**: Summer snow in London? Anyway, a more-transparent version of the default snowskylo.ace covers about half the very dark high sky, under light snow.

**Autumn / Snow**: The high sky shown in the next picture illustrates the freedom to draw anything you like in the sky: this texture is a "cloudy" rendition of the flag of the Swiss canton of the Grisons (Graubuenden in German), for which these environments were designed. A low sky layer also displays a slow-moving flying saucer above the "ground fog". The water is made to reflect the special upper cloud layer.



**Winter / Snow**: Dense "ground fog" does not affect the gray and white sky, which does block the sun and moon. The snowfall is dense.

[NEW SINCE V1.106]

### 3.15.7 HANDLING THE CONFLICT WITH Sky! Conductor

Sky! Conductor (a free utility made by "how in the world", http://www.howintheworld.com/downloads.shtml) changes the sky and weather in MSTS (it does not affect the RE) for routes that you select. But in so doing it may remove your own environmental settings for those routes.

As a result, in MSTS **you can use EITHER your own environmental settings, OR Sky! Conductor's weather, but NOT BOTH at the same time**.

The problem is that Sky! Conductor overwrites the *.env file, in which you have defined your environmental properties. This overwriting restores MSTS default settings for the environment (but still uses those of your textures that have default names, such as waterbot.ace, watermid.ace and watertop.ace). This is in particular true of Sky! Conductor's option to select the "TS Default" sky choice: you lose your environmental settings, and get the default MSTS environment.

The way Sky! Conductor operates is this: You must first declare your route to be changeable by Sky! Conductor. In this process, its Route Adder will place copies of its sky textures (named hitw-*.ace) into your route's ENVFILES\TEXTURES folder (this should not overwrite any of your own textures, except in the unlikely case that you have named them like hitw-*.ace). If you then run Sky! Conductor to select a specific weather type, it will copy its own sun.env, rain.env and snow.env files over yours: these *.env files call the special Sky! Conductor textures (and make some other changes, such as rain, snow and fog density).

NOTE: If you have changed the names of the *.env files or added such names (including in your route's *.trk file), Sky! Conductor may not work in the first place, or may get confused. In that case, you will need to look carefully at the situation that you have.

Therefore, in the normal case, **if you have applied Sky! Conductor to your route, and if you want to recover your environmental properties,** you have to restore to their pre-Sky! Conductor version the *.env files that Sky! Conductor changed. That you can **do as follows**, since Sky! Conductor made a backup copy of your *.env files:

- find the folder Sky!_Files\myroute\EnvFiles inside the Train Simulator folder (where "myroute" is the name of your route).

- copy all the *.env files that you find there (normally sun.env, rain.env and snow.env) to your route's ENVFILES folder;

- reload your route into MSTS: you should see your own environment again, and no Sky! Conductor skies.

## 4. RUNNING TESTS IN MSTS

### 4.1 Need to use Activities

You should **run tests after each significant track addition or change**, especially after adding or changing loops and/or wyes. That means **running a train** on your route in MSTS to see that it works as planned. And that in turn means that **you have to create at least one activity**.

If you don't run such tests, you will probably end up with a route that does not work, and, worse, you will have no idea which part (or parts) of the route causes the problem! You would then have to hunt down the bad parts of the route by blind trial and error, a very time-consuming and frustrating exercise.

Another purpose of testing is to inspect the terrain and scenery that you create: the view from the train is what counts for the end-user, rather than the view from the RE camera. See section 4.7 for more details.

To run a test in MSTS, you must make at least one activity (without activities, MSTS will not start your route, even if you only want to use the Explore Route mode). But since activities depend on the route structure, they become invalid after each significant route change (like adding or deleting switches): therefore, **you must delete old activities and make new activities for new tests**. So it is best to make the simplest possible activity for each test.

### 4.2 Deleting existing Activities

**Before making new activities for new tests, delete all the old ones** (if switches have been added or deleted, or if track connections have been made or broken):

- delete all *.act files in your route's ACTIVITIES subfolder;

- delete all *.pat files in your route's PATHS Subfolder;

- delete all *.srv files in your route's SERVICES subfolder.

### 4.3 Creating new Activities

See section 5 for instructions on how to create new activities.

For testing purposes, the simplest activities are fine. Their main purpose is to define starting points (and directions) for your train(s), so you can use the Explore Route mode to drive around your route, especially toward and around the changed parts.

## 4.4 Tests during running

**The main things to test are:**

- try starting MSTS;

- find and select your route;

- use mainly the Explore Route mode (an actual activity freezes some switches, reducing your ability to test): select a train (not just a lonely locomotive - bad switches can break up a train, which is one sign of trouble in your route); and select start and end positions;

- drive towards and around the changed parts of your route;

- look out for bad joints: they give the train a jolt when you pass over them;

- try switching all switches by pressing G and Shift-G (this will not work at entrances to loops: those switches will not switch; this is normal in MSTS);

- run over as many switches as possible (note that a problem in one area can be due to a track error far away!);

- try winter and night conditions, if you have imported objects and/or textures.

## 4.5 Main causes of problems

**The main trouble spots are usually loops, wyes, and bad track joints. Also, switches with unconnected exits make MSTS fail (you must attach tracks to all three exits of a switch).**

However, your test ride may fail elsewhere on your route. If changing a particular switch freezes up MSTS, that switch is not necessarily the cause of your problem! The error probably lies in the way you set up the last-changed part of the route. Every joint between tracks may be fine, but the presence or location of a loop or wye can itself be the problem.

One useful way to identify which loop or wye is responsible for your problems is to break open a loop or wye by removing one of its track sections; then run a new test (with a new activity) at the same location where the last one failed. If the test is now successful, change that loop or wye!

In a wye, it seems that the problematic part is the Y-switch itself, not the other parts of the wye. Therefore, it may help to replace the Y-switch by a normal (asymmetrical) switch.

Concerning bad joints, see section 3.3.12.

It should be obvious when you have failed to import the proper files for objects and/or textures and/or sounds:  MSTS will crash.

NOTE:  A free track end is not a problem:  there is no need to finish off a line with a buffer (bumper), or anything else.

## 4.6 Main solutions

Some solutions to problems you have found include:

- **put problematic loops and wyes on secondary branch lines** (the "farther" they are from the mainline, the better, where "farther" means that the route passes more switches to get there);  if that does not work, **replace a bad loop by a wye;  or eliminate the bad loop or wye**;

- **fix bad joints**, for example:  reinstall tracks;  choose shorter track sections in slopes;  fine-tune dynamic tracks;

- **import missing files**.

## 4.7 Visual scenery inspection

Running a train and closely inspecting what the driver and passengers will see is very revealing and useful.  Many details that you may have overlooked while shaping the terrain and placing object in RE can show up when you watch that scenery go by from the train.

A good inspection view is given by MSTS option Shift-1:  that shows the tracks and all close objects in much detail.

A very useful tool for this visual inspection is the ability to manipulate the external views (options 2, 3 and 4 in MSTS).  You get much greater freedom of viewing by pressing Ctrl-Shift-9 in those two modes:  you can then swing the view around by dragging the mouse while holding its right button depressed.  You can still raise or lower the viewpoint, and swing it around the train while in motion.  (You may cancel that extra freedom by pressing Ctrl-Shift-9 again.)

## 5. THE ACTIVITY EDITOR (AE)

This discussion of the Activity Editor explains **how to make the simplest possible activities, for the purpose of testing your route** (see section 4). It also discusses the relationship between route building and activity design in the area of switches and signals: to make interesting activities possible, it is important to carefully select manual vs. automatic switches, and to properly place signals.

For a much more complete explanation of activities and for more information about other options, see Richard Garber's **Step-by-Step Guide to Train Sim Activities**, the MSTS Help and the Train-Sim.com Activity Forum.

### 5.1 Uses of Activities

If you distribute your route to other users, remember that they can only drive on it if they have activities designed for that route: so you should also supply at least one activity for them to use. Such activities can be as complex as you like. An important use of activities is to show the users interesting features of your route: an activity can start up the user at a good location with a suitable train, in interesting weather at a time of day that gives nice lighting. But **make final activities for distribution only after you have tested your route!**

**One very useful function of activities is to create starting locations for the Explore Route option of MSTS.** By supplying ending locations, the start direction of a train is also specified. And that is the function that we use for testing routes. So all you strictly need to specify is a starting location and an ending location; however, an activity requires other information to run, so you will supply that as well.

[NEW SINCE V2] TIP: To describe a start direction, it may be more useful to specify a compass direction (like "west") or other clear designation (such as "downslope"), rather than a specific location (like "Coaltown"). If you specify an ending location like "Coaltown", the user needs to know the local geography well enough to understand which direction is implied. The names you use for starting and ending locations need not be related with platform names or siding names: so you may choose whatever is clearest to the user, such as from "Oakdale yard" toward "east".

[NEW SINCE V2] NOTE: The order in which activities, locomotives, consists, starting locations and ending locations appear in the MSTS selection lists is often confusing: it is not only based on the alphabetical order, but also depends on such things as the use of upper- vs. lower-case letters. It is not known how to change that ordering logic.

WARNING: If you design an activity that uses non-default locomotives or cars, you must tell the end user which those add-ons are and where to find them.

## 5.2 Planning Activities

IMPORTANT: **activities should not pass through a loop or wye** - otherwise MSTS will almost certainly fail when starting that activity. This severely restricts the possibilities for activities.

[NEW SINCE V2] It is possible **to turn a train around during an activity**, by using a wye that does not use a wye-switch: the activity should have a reverse point before the wye, and the player should proceed beyond the reverse point into the wye, throwing the switches as needed.

My suggestion for a **really simple activity** for testing purposes is this: **place a starting location anywhere you like**, but away from any new tracks you want to test (so your train will not start on problematic tracks); **then place an ending location nearby**, such as a few hundred meters away from the starting location in the direction in which you plan to start driving.

For this purpose, you don't need to have defined names in RE, such as platform, siding or other names. But it helps to have a map (such as a hand-drawn map) with "place names": these are just names that you will recognize when you see them in the Explore Route mode of MSTS.

## 5.3 Making a simple Activity

**The needed steps to create such a simple activity are the following (you must fill in all the requested information: otherwise your activity will not work!):**

- start Editors and Tools;

- start the Activity Editor;

- select **File**, then **New…**;

- type in an Activity display name: for instance, "A to B" (without the quotation "" marks), where A is a starting location (such as Central London) and B an ending location (such as North London); I recommend that you copy that name "A to B" to the clipboard (press Ctrl-C), so that you can insert it later in the many boxes that require information from you;

- select any non-zero duration (1:00 is fine);

- click on **Edit Activity description** and paste "A to B" in its box (without the quotes "", of course);

- do the same with **Edit Activity brief**;

- click on **New** under the blank Player service box: this opens the Service editor;

- paste "A to B" again in the top two boxes;

- choose a consist (train), by opening the drop-down box and making your choice (since you will be using the Explore Route mode of MSTS, you will be able to make any other choice of consist/train while in MSTS, so this choice does not matter for testing purposes);

- click on **New** under the blank Path box;

- paste "A to B" for the Path name, and click OK;

- paste "A to B" for the Path display name, and click OK;

- the Path editor opens: enter "A" for the Starting location, and "B" for the Ending location (still without the quotes "");

- shift the Path editor window away from the map, if it lies over the map;

- on the map, define the start point of your path (see section I for an illustration of how to do that): if necessary, you can drag the map around with the mouse, and magnify it by dragging the mouse up/down while pressing the right mouse button; move the circle with the mouse to your desired start point (which need not be a platform or siding or whatever); right-click on that circle to open a menu, and click on **Place start point** (note that this point is where the tail of your train will be located);

- a green line should now light up to define a path that goes as far as the editor can predict it (I think by going straight through switches; it may back up on itself if it hits a loop, confusing things!): if the path starts in the wrong direction, right-click again on the start position and then click on **Toggle start direction**;

- now place your end point in the same manner as the start point, on the track (again, there is no need to choose a platform or siding or whatever); for testing purposes, I recommend putting it a few train lengths away from the start point in the traveling direction toward the destination B;

- click on **Leave path editor**;

- answer **Yes** to "Do you want to save path changes?";

- press **OK** on the Service editor;

- if you want (omit this for testing purposes), you can enter start time, season and weather;

- exit the Activity Editor: click on **File - Save As…**;

- paste "A to B" in the box for the File name;

- answer **Yes** to "Do you want to exit Train Simulator?";

- IMPORTANT: **before starting to make another activity, exit from and restart the Editors and Tools** (choose Yes if offered to Play).

You can also **edit an existing activity**:

- open AE;

- click **File** - **Open…**;

- select your route's folder;

- select its Activities folder;

- select the activity to be edited;

- make any changes you wish (I recommend not changing the major names, to avoid creating new files and confusion);

- save the result.

[NEW SINCE V1.106]

## 5.4 Manual vs. automatic switches

MSTS offers **two kinds of switches**:

- **manual switches**, which the train driver (the game player) can throw during any activity or in Explore Route mode;

- **computer-controlled switches**, which the player can only throw in Explore Route mode.

**Computer-controlled switches are also called automatic switches**. Manual switches contain the letters "Mnl" in their names, such as A1tPnt10dLftMnl.s. The automatic switches are not labeled as such: A1tPnt10dLft.s is the automatic version of A1tPnt10dLftMnl.s. (Note that manual and automatic switches cannot be distinguished visually in the Activity Editor; in both RE and MSTS a manual switch has a visible handle that moves when the switch is thrown, while an automatic switch does not have such a handle.)

The main reason for the presence of **automatic switches** is to **protect computer-controlled trains** from adverse actions by the player or other trains. **Computer-controlled trains are also called traffic trains or AI trains** (AI stands for Artificial Intelligence); they only exist in activities; they do not run in Explore Route mode.

Automatic switches are set by the computer to allow a traffic train to follow its planned path. Without this protection, a player or other train could throw a switch against a traffic train, disrupting its run:  after that the activity may fail, since the traffic train may not be where expected.

The main disadvantage of automatic switches is obvious:  since a player can't throw them during an activity, the player's switching options are restricted.  If all switches were of automatic type, the player could not make any switching maneuvers for picking up or dropping off wagons on a spur or siding or in a yard, for example.

Therefore, **a compromise must be found between safety (many automatic switches, few manual switches) and flexibility (few automatic switches, many manual switches)**.  The compromise clearly depends on the activities that are to be run on the route.  Therefore, in principle, the route builder should know beforehand exactly where to place manual switches and automatic switches.  This will be illustrated in section 5.7.

In other words, ideally the activities should be planned in detail before the switch type can be selected and the switches laid!  That is certainly not practical, because most route designers first build the route and only then start to think about activities;  this is especially true with fictional routes, which may grow in an unplanned fashion.  Also, even for well-planned routes, it is desirable to be able to create new activities later on.

Fortunately, it is possible to decide on the type of switch after the route has been completed:  for example, the route builder may initially place only manual switches (because they often will be more numerous than automatic switches).  Later, after the activities have been planned, the route builder can come back and change some switches to the other type (automatic).  This is not without risks, though, so it is important to back up the route before doing this.

(CAUTION:  When you replace switches, or any other tracks, you must first remove all track-related objects connected to those switches or tracks, like signals, mileposts, sounds, etc.  Some of these can be moved temporarily, while others must be deleted.  After placing the new switches or tracks, you can restore the track-related objects, moving them back or placing them again.  However, it is generally a bad idea to attach any track-related objects to switches!)

[NEW SINCE V1.106]
## 5.5 Signal location

Signals are very important in activities that include traffic trains: they prevent collisions between trains. **Signals must be properly placed to provide the necessary protection within the planned activities.**

(Signals are not important for safety in the Explore Route mode, since there are no traffic trains to worry about: then the signals automatically turn green for the player's train.)

It must be stated that we don't know very much about how the signals in MSTS work; we do know that the signals in different default routes work differently. So this discussion will have to remain rather general and vague.

In our discussion, **the stretch of track which lies between two successive signals will be called a block**, no matter how short or long that stretch may be. A signal thus protects one block, which runs until the next signal.

Each signal has a logic that determines how it controls trains: the logic depends on the presence and direction of travel of other traffic, and may depend on other signals to which it is coupled. There are many different signals in MSTS, with different logic. We can here only discuss a few aspects of this logic.

**The first general principle of the signal logic in MSTS is: "first come, first serve".** The first train to approach a block and claim access to that block will get permission to enter that block. No trains are given a higher priority over others, as for express trains over local trains, or passenger trains over freight trains. In this connection, it is useful to know that **the player train is always the first to start in an activity**.

When placing signals for an activity, it is important to understand **another general rule of MSTS signals**: **their logic looks ahead several blocks (not just one block) and tries to reserve those blocks for the approaching train**. The number of blocks varies from two to four, depending on the signals used in the different routes.

**Let's illustrate** what this means for **the 4-block case** (this case applies to the UK route, the JAPAN1 route, the USA1 route, as well as the First Route of this Guide, as discussed further in section 5.7). Consider a train that moves along tracks with simple signals; it may be a player train or a traffic train. Then the first four blocks ahead of this train will be protected for this train, if there are no other trains there: no other trains will be able to enter those blocks. A block will be opened up to another train only when the first train has left that block. If you display the signals in the Activity Editor, you can see those four reserved blocks marked by four green signals. (Actually, in MSTS, the signals may not all be green: with simple signals, the four next signals may show green-yellow-red-red. After the train has passed the first of these signals and moved into the next block, they will show red-green-yellow-red.)

This 4-block logic implies that when an activity is launched, the first train to start will reserve four blocks ahead for itself (that can be a very long stretch, if the signals are widely spaced!). The second train to start will attempt to do the same: it will try to reserve four blocks ahead, only limited by what the first train has already reserved. The same happens with any trains that start in the activity later on: they reserve the next free blocks, up to four.

This multi-block logic easily leads to stalemates: trains can block each other's progress, if the placement of signals is not planned carefully.

Also, despite the protection given by the signals, it remains possible to cause accidents. For example, if your activity starts two trains within the same block, they are not protected from each other and may collide.

Once the trains have started in an activity, they can get as close together as in adjacent blocks. Exactly how close together they may come depends on the logic of the individual signals: each different signal in MSTS has a different logic, and therefore behaves in its own way towards passing trains.

**There are many kinds of signals in MSTS** (as there are in reality): **each default route has its own set of specific signals that perform a variety of functions**. **Their logic has, for the most part, not yet been published or deciphered.** Another complication is the possibility to interconnect signals, so that the behavior of one signal depends closely on the behavior of others. Until more is known about them, it will be difficult to plan the use of the MSTS signals exactly.

It is therefore impossible to give general guidelines about where to place signals: this will depend very much on the route, the types of signals used and the activities being planned. This will be further discussed in section 5.7 for a specific route, First Route.

**A trial-and-error approach is the best suggestion** I can make at this point. Start by designing activities with simple assumptions about how signals control access to tracks, then try out them out and fine-tune as needed.

[NEW SINCE V1.106]

## 5.6 The structure of activities

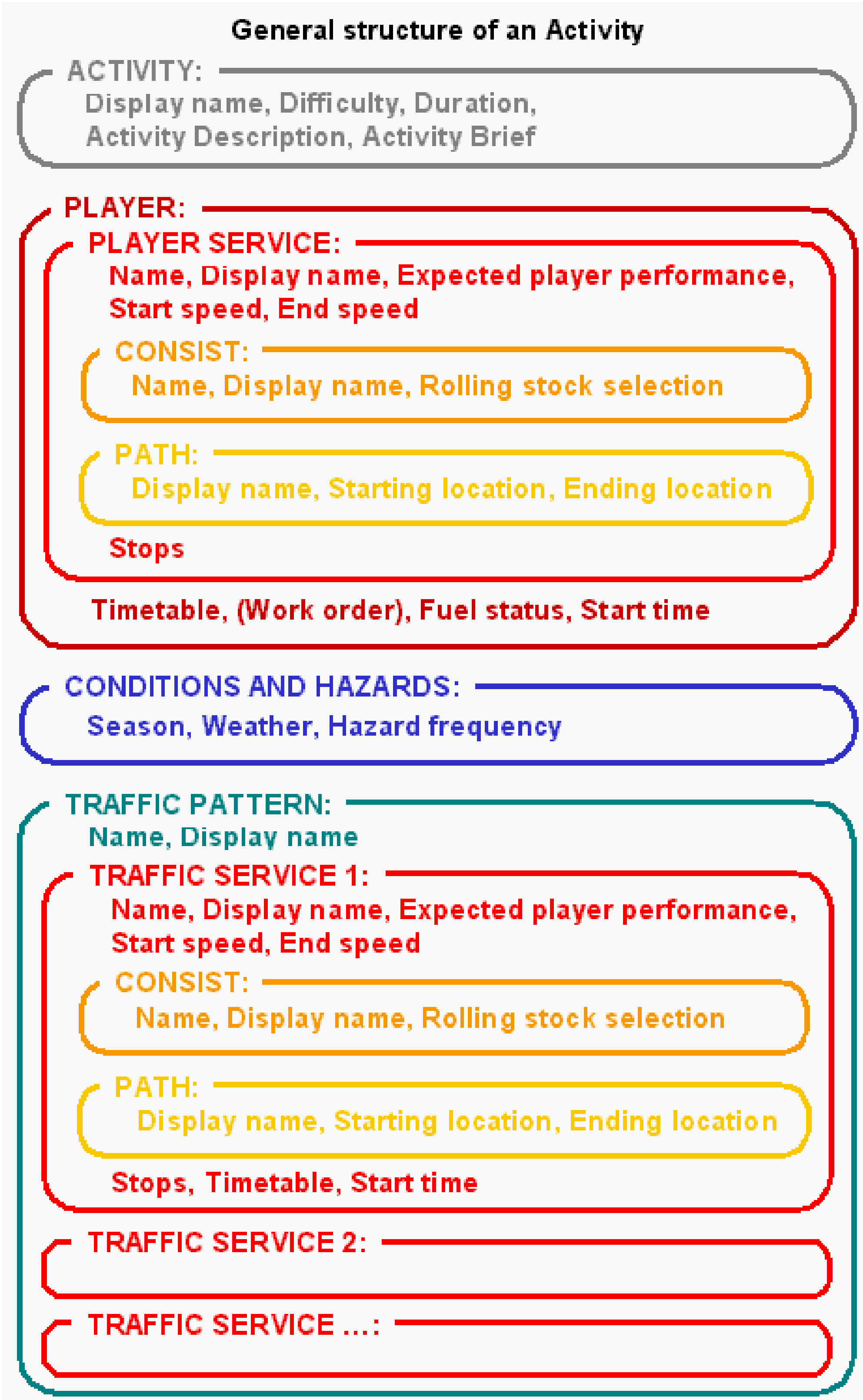
This section is intended to give a brief overview of the way activities are structured in MSTS and the Activity Editor. That may help you to understand how interesting activities can be designed, and therefore how routes should be planned to enable such activities. For more details consult Richard Garber's Step by Step Guide to Train Sim Activities.

**In general terms, an activity allows the player to drive one predefined train (the player train) along predefined tracks, starting at a predefined time of day in a predefined season, with predefined weather.** (Predefined means that these aspects are set by the activity designer, and cannot be changed by the player.) A timetable may be included, as well as a work order: MSTS will compare the player's performance against the timetable and the work order (however, the Activity Editor does not allow creating a work order). The activity may include "traffic": one or more other trains (we call them traffic trains), which are computer-controlled. Each traffic train also follows predefined tracks, starting at a predefined time.

To use the Activity Editor of MSTS, it is useful to know the **terminology used by MSTS** to describe activities and their parts (see also the graphic below):

- an **activity** contains a **player service** and an optional **traffic pattern**, as well as a **timetable**, a **work order**, a **fuel status**, a **start time**, **conditions and hazards**, a **difficulty** and a **duration**;
- the **player** is the user of MSTS: the player drives one train (which we call the player train);
- a **service** is the combination of a **consist** and a **path**, together with any **stops** at stations;
- a **traffic pattern** is a combination of computer-controlled services, each of which has a start time: it thus defines all traffic trains, other than the player train;
- a **consist** is any combination of coupled locomotives and/or wagons;
- a **path** is the unique track sequence that a train is supposed to follow during the activity, from a starting location to an ending location; it can also include **location events**, **action events**, and **time events**, as well as immobile consists placed at specific locations on the route;
- **events** are situations which trigger an action or command, depending on location, action or time (such as the order to pick up wagons from a particular siding, or the display of an on-screen message);
- a **timetable** lists desired departure and arrival times at station stops;
- a **work order** summarizes the actions to be taken by the player;
- a **fuel status** specifies the amount of fuel with which the player train starts;
- a **start time** indicates when the player train starts;
- **conditions and hazards** specify season, weather, and the frequency of animals and people;
- a **difficulty** informs the player of the degree of difficulty of the activity;
- a **duration** informs the player of the expected time needed to complete the activity.

[NEW SINCE V2]



General structure of an Activity
ACTIVITY:
Display name, Difficulty, Duration,
Activity Description, Activity Brief
PLAYER:
PLAYER SERVICE:
Name, Display name, Expected player performance,
Start speed, End speed
CONSIST:
Name, Display name, Rolling stock selection
PATH:
Display name, Starting location, Ending location
Stops
Timetable, (Work order), Fuel status, Start time
CONDITIONS AND HAZARDS:
Season, Weather, Hazard frequency
TRAFFIC PATTERN:
Name, Display name
TRAFFIC SERVICE 1:
Name, Display name, Expected player performance,
Start speed, End speed
CONSIST:
Name, Display name, Rolling stock selection
PATH:
Display name, Starting location, Ending location
Stops, Timetable, Start time
TRAFFIC SERVICE 2:
TRAFFIC SERVICE ...:

**The simplest activities must have at least the following elements:**

- a consist;
- a path;
- a start time;
- a duration (may not be zero);
- a season and weather.

Such a simple activity can be used to allow the user to explore a route, either by running the activity itself, or in the Explore Route mode of MSTS: that is what we have discussed in previous sections.

**In the more general case, an activity may have the following elements:**

- a player-drivable consist: the player train (with any number of locomotives and wagons);
- the initial fuel status of the player train;
- a start time for the player train;
- a path for the player train (this path may contain a variety of "events", such as picking up or setting out wagons);
- starting and ending speeds for the player train;
- a timetable for the player train, with prescribed stops at preselected stations;
- an activity brief and a work order for the player train, explaining the player tasks;
- immobile consists of wagons and/or locomotives standing at predefined locations (without path), which the player train can pick up;
- a difficulty level (to warn the player);
- a duration and an expected player performance (these may be used in the evaluation of the player's execution of the activity, but play no role otherwise);
- a season and weather;
- hazards (people or deer appearing on or near the tracks);
- a traffic pattern of other trains (traffic trains): this combines any number of services with given start times (each service specifies one consist running along a particular path; the locomotives used in traffic consists may be player-drivable or only computer-controlled).

Since it can be delicate to coordinate the running of several trains, **the Activity Editor includes a very convenient activity simulator** (called VCR Tool in the Sybex book and in Richard Garber's Guide): it allows you to see the entire route in map form, and to simulate on that map the motion of all the trains of an activity, including the effects of manual vs. automatic switches, and of signals.

The activity simulator makes it possible to check out the overall traffic pattern of an activity, as it evolves over time. This is very helpful to avoid train conflicts that otherwise would be difficult to see from the player train in MSTS: the simulator actually shows all trains moving along their individual paths, running at speeds determined by the speed limits, stopping at red signals and stations. This simulator is an essential tool for establishing a reasonable timetable for several trains running over the same tracks.

Note, however, that this activity simulator does not accurately reproduce the actual train speeds, as they will be when running the activity in MSTS (for example, trains may not be able to reach the speed limit on grades, and will therefore be slowed down in MSTS relative to the activity simulator;  also, station stops and wagon pickup maneuvers may be mistimed):  therefore, it is still necessary to run the activity yourself in MSTS, to check and fine-tune the scheduling of all trains.

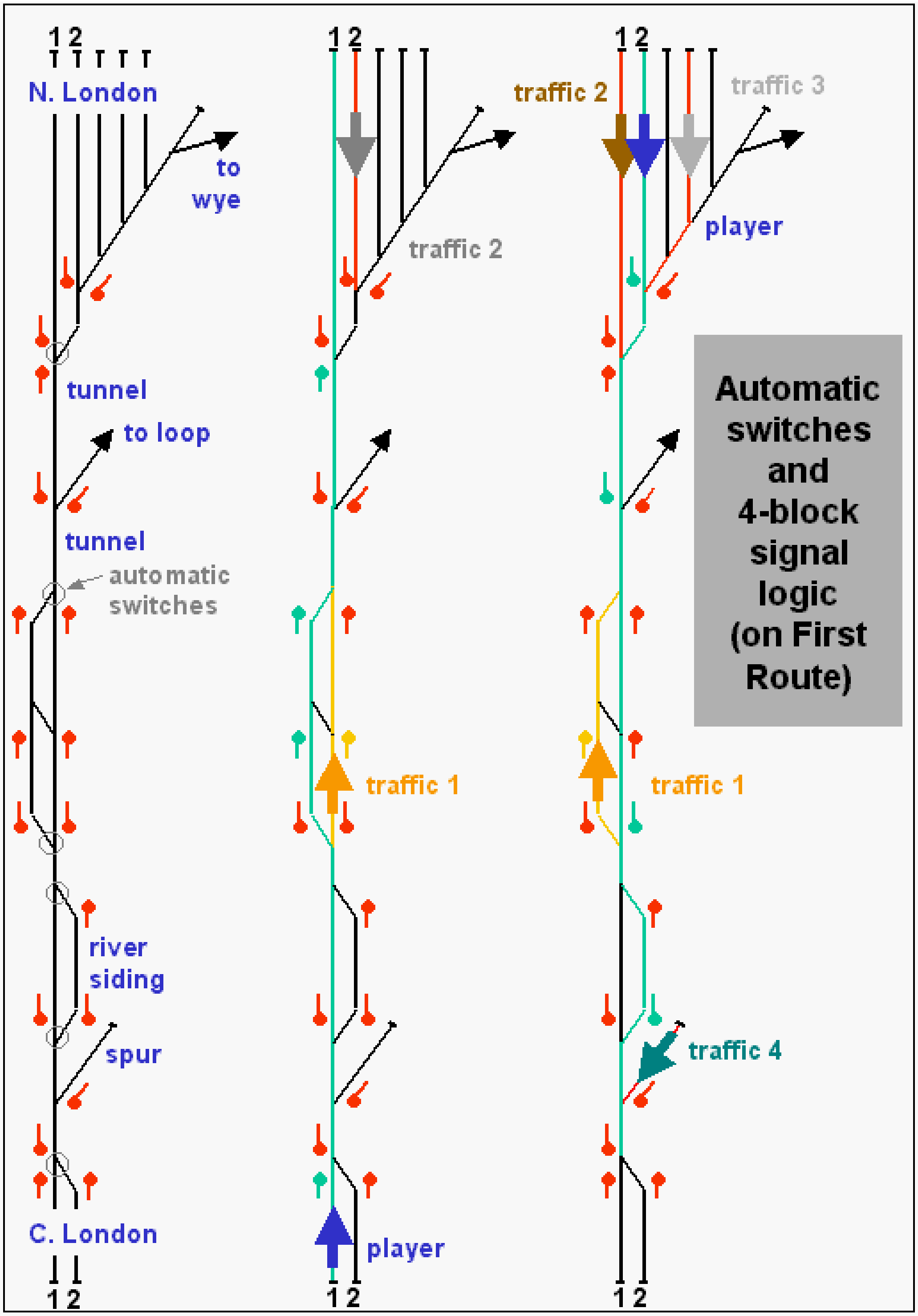
[NEW SINCE V1.106]

## 5.7 Discussion on First Route

We here use our First Route to illustrate some of the issues presented above, especially switches and signals.  First Route is really too short for interesting activities, but is long enough to discuss some general points valid for most routes.

First, **loops and wyes are off-bounds for activities**:  any activity that tries to use a loop or a wye will most likely fail.  So we can forget about the loop and wye in First Route.  That severely restricts the possibilities for activities.  It means in particular that we cannot turn around a train or locomotive during an activity (but backing up is possible).  It also means that activities will necessarily be point-to-point operations.  And since closed circuits do not work in MSTS, no activity can include repeated circuits either.

**The next figure shows First Route in schematic form**, excluding the loop and the wye.




The gray circles around some switches in the left part of the figure indicate where I would put **automatic switches** (I did not do that in First Route, since I produced no activities for it with traffic trains). As you can see, these automatic switches are on the main line, along which I would run traffic trains that need protection. But this choice depends a lot on what activities are planned, as we discuss next.

Suppose that I plan to design activities that have no operations to the spur or across the high-speed crossover. In this case, I can leave the spur and crossover switches as manual (as illustrated in the figure), since no train will throw them against traffic trains.

I have "frozen" the river siding with automatic switches, to allow an activity to safely send trains either through the siding or along the mainline. But this implies that the player will not be able to throw the switch and choose between the siding or the adjacent main line (on the other hand, the activity may make the player's train back up using a reversing point, and then force him/her to branch the other way, but the player still has no freedom of choice).

Suppose that I wanted to make the player use the spur freely, for example to pick up or set out wagons. Then that spur switch should still remain manual, although it would endanger the traffic trains: if the spur switch were made automatic, the player could not use it to drive onto the spur. The same holds true for the North London switch that leads to the yard (it is the switch on platform track 2): if I want the player to be able to use both track 2 and the yard, that switch must remain manual.

These situations show that there is **no complete safety with complete flexibility**: a compromise is needed whenever traffic trains share the same tracks as the player train.

Now consider **signals**. The red symbols in the figure above represent the signals that I placed in my version of First Route: their heads point in the direction of travel that they control.

The middle and right diagrams in the above figure show what happens when we start an activity with signals that obey the **4-block logic**. **The first train to start in an activity is always the player train** (traffic trains must start at a later time, even if only a few seconds later).

Imagine an activity in which the player train starts on platform 1 at Central London, and travels to North London, following a path that stays on the leftmost tracks all the way. In the middle diagram, the thick blue arrow shows the player train's starting location and direction. Since there can be no other train on the route at this stage, the player train reserves the next four blocks for itself: the green line shows the part of the player train's path that is protected this way, marked by the green signals (as you would see them on the Activity Editor map). So **the player train is always safe to depart**.

Note how the green line extends all the way to the destination! That is because the four reserved blocks add up to the complete length of this short route. If the signals used 2-block logic, the green line would only extend to the signal just before the first tunnel: that signal and the next one would be red.

Using the 4-block logic again, let's start up a second train ("traffic 1") soon after the player train has started. If we start "traffic 1" on the high-speed dual track headed north (see its yellow arrow in the middle diagram), then it can move forward through one signal to just before the first tunnel (the yellow signal tells it to expect a red signal next): there it is stopped by the player train, even though the player train may still be far behind and not a real danger! If **2-block logic** applied, then "traffic 1" could move into the tunnel and beyond, as long as the player train had not left the Central London station yet (once the player train leaves the station, it would try to turn the leftmost tunnel-entrance signal green).

Again with 4-block logic, let's start a traffic train ("traffic 2") southbound on platform track 2 at North London: see the gray arrow in the middle diagram. It is immediately blocked by the player train, even if the player train has not started moving yet! It will not be able to leave the station until the player train has moved onto platform track 1 at North London! With 2-block logic, it would still be blocked by "traffic 1". If "traffic 1" did not exist, "traffic 2" could start, as long as the player train had not left the Central London station yet; then "traffic 2" and the player train would meet somewhere midway (of course, their paths must be so chosen that they do not meet head-on in a stalemate, but on parallel tracks).

The right-hand diagram in the above figure shows similar situations for a southbound player path: the player train now starts on platform track 2 at North London, and stays on the leftmost tracks. The 4-block logic immediately reserves the track for the player train up to the entrance to the Central London station (the signal into platform track 2 at Central London will turn green as soon as the player train passes its first signal).

If we start "traffic 1" soon after the player train, it will be blocked on the yellow tracks before the tunnel until the player train has passed. The trains "traffic 2" and "traffic 3" are of course also blocked. But they could start moving as soon as the player train passes its second signal (between the two tunnels); I am not sure which of "traffic 2" and "traffic 3" will go first. Even "traffic 4" is blocked until the player train has passed! With 2-block logic, only "traffic 4" could start moving right away (as long as the player train has not passed its second signal).

We have so far mentioned only trains that start near the beginning of the activity. A traffic train can start at any later time as well. For example, a traffic train could mimic the player train at a later time and duplicate its run along the same path. This kind of scenario includes a danger: if this traffic train starts too soon, so that the player train is still in the station, it will bump into the back of the player train and halt the activity due to a collision. If this traffic train follows the player train with a safe delay, it will be able to stay one block away from the player train all the way to the destination.

Another aspect is the risk of **stalemates**, where two trains block each other. A simple example is the entrance to a dead-end station platform track (as in First Route): if a platform track is already occupied, a second train will not be able to enter it, and the arriving train may not be able to back up to clear the way for the other (in MSTS activities a train that backs up usually finds red signals).

Stalemates are serious risks on single-track lines: two trains coming face to face will block each other, and thus may block the whole activity. In such cases, it is important to provide dual-track passing tracks, suitably protected with automatic switches and well-placed signals.

**The examples that we have discussed here show how restrictive a route can be in terms of what is possible within activities.** It is clear that you would **gain by having more closely-**

**spaced blocks (using more signals)**, so that no train can monopolize large parts of the route for itself, freezing out interesting movements by other trains. Also, **more tracks (including parallel tracks) and switches often help**. After all, the purpose of having other trains on the route is to see them in action rather than waiting at distant signals!

As you can tell, planning a "traffic pattern" with several trains starting at different places and at different times can get quite complicated. But my main point here is that **your activity plans will depend on having made the right choices for automatic switches and signals**.

[NEW SINCE V1.106]

## 5.8 Introductory Train Rides (ITRs)

**MSTS allows the user to run an automated Introductory Train Ride (ITR) along a route.** This permits the user to "ride along" and enjoy the view: the MSTS Artificial Intelligence engineer does the driving. An ITR gives the new user a good opportunity to become familiar with the route.

An ITR is provided for the First Route included with this Guide.

The user selects the route's ITR as the top option on the initial startup screen of MSTS. Selecting "Introductory Train Ride" opens a list of routes that have an ITR. After selecting the desired route, the user may deselect the "Change Views" option: this prevents MSTS from automatically switching the viewing cameras during the ride, so the user can make the camera choices that he/she desires. Pressing Start then loads the route and starts the automated ride.

You make an ITR for your route just like a normal activity, but then you change its "Mode" so it is interpreted by MSTS as an ITR rather than an activity.

However, it is good to keep a number of issues in mind when creating an ITR: not every activity works flawlessly when converted to an ITR! In the following we will therefore describe not only how to create an ITR, but also what aspects will need special attention.

We will call the viewer's train the ITR train in the following: it is an Artificial Intelligence (AI) train, operated by the MSTS software rather than the viewer.

CREDIT: I am grateful to Jeff Bush for bringing the topic of ITRs to my attention, and for explaining how to create and manage them.

### CAN I CONVERT ANY ACTIVITY TO AN INTRODUCTORY TRAIN RIDE (ITR)?

In general, yes. However, you will find that some problems can arise, because the ITR train is totally out of the viewer's control. For example, the ITR train may not be able to climb steep slopes, so the ride gets stuck. Or another traffic train may cause a stalemate, blocking the ITR train forever.

You may include in an ITR many features of normal activities: station stops, reverse points, timetables, other AI traffic, etc.

However, some activity features appear <u>not</u> to operate in MSTS, including restricted speeds, wait points, and all events.

Also, the AI Engineer will probably drive differently from you, so you need to test the ITR, and perhaps even change your route in some ways to accommodate this Engineer's driving habits, as discussed below.

## HOW DO I CREATE AN ITR?

To create a new Introductory Train Ride, do the following:

- create and save a normal activity (you could call it "Introductory Train Ride", or "ITR", or anything else, as long as its file name is different from that of the other activities for this route);

- exit from the Activity Editor;

- open the activity's *.act file (with a Unicode-capable editor like WordPad; the file should be in the route's ACTIVITIES folder);

- find the line

```
Mode ( 2 )
```

- in that line, change the 2 to 0;

- save the file.

After this mode change, this "activity" will no longer appear in MSTS in the list of activities.

NOTE: Modifying the ITR later again with the Activity Editor keeps the Mode set to 0, so it stays an ITR.

## CAN I CREATE MORE THAN ONE ITR?

Yes, but MSTS only uses the first ITR that it finds in your route's ACTIVITIES folder.

If you want to use various ITRs, you can place all your ITRs in a subfolder of ACTIVITIES, such as ACTIVITIES\ITRS, and copy the one you want to ride to the ACTIVITIES folder, removing any ITRs that are there: then MSTS will pick and run that one ITR.

## HOW DO I AVOID THE MSTS NARRATIVE?

By default, an ITR is accompanied by a spoken introduction to MSTS. To eliminate this narrative, do the following (instead, you could of course include your own spoken narrative, but it will play in every route!):

- find the ITR.TUT file in the GLOBAL\TUTOR folder;

- open it and delete ALL its lines;

- save the (now empty) file.

This also makes the HUDs work (see next).

## HOW DO I MAKE THE HUDs WORK?

By default, most HUDs ("head-up displays" opened with the F keys in MSTS) are disabled in ITRs. To enable them, make the same change given above to avoid the spoken narrative. It is not known how to switch off the green ITR HUDs.

## HOW DO I IMPROVE LOCOMOTIVE PERFORMANCE?

You will find that ITR trains tend to alternatively speed up and slow down on straightaways, and to slow down in turns and on grades, even going down: no simple solution has been found to avoid this. A complication is that the behavior of ITR trains seems to depend on how long they have traveled in MSTS.

If a locomotive can't climb a grade in an ITR, you can of course change to another locomotive, or reduce the number or weight of cars it pulls. You may also change the locomotive's performance, realizing that this will affect not only the ITR, but all driving. For example, you could prevent its wheels from slipping by inserting in the "wagon" section of its *.eng file the line

```
AntiSlip ( )
```

(see the MSTS Help Doc for details).

## HOW DO I SET THE START POINT AT A PLATFORM?

You may want to start an ITR at an interactive platform (which is the green interactive track object placed down the middle of a track, separate from the visible platform structure), in the hope that it will wait a bit before moving and leaving the station.

If the ITR train is placed too much forward, it may start immediately without waiting. If you place it too far back, it will assume that it still has to move forward to reach the platform (although it is already there). This needs fine-tuning by trial and error of the start point of the path.

## HOW DO I MAKE AN ITR TRAIN STOP PROPERLY AT A PLATFORM?

This is a delicate matter, which may require fine-tuning by trial and error both in the Activity Editor and in the Route Editor. Here are some options to try.

To make a given ITR train stop at a desired point along the platform, you may need to move the two green end points of the interactive platform in the Route Editor.

An ITR train seems to notice a platform only when it reaches its near end, so it may easily overshoot the far end of the platform.

If your station has two tracks with a platform on each, like most of the EUROPE1 stations, you may, in the Route Editor, slide the far end of the platform in toward the center of the visible station, so that the consist, when it inevitably overruns the near end of the platform, ends up still nicely stopped at the visible station. On a single-track line with a single platform, the best solution is to slide both ends of the platform to the center of the station, making for a very short Interactive Object. That way, ITR trains from either direction can overrun but still stop nicely at the visible object. A drawback of this approach is that it makes it harder for the driver of normal activities to stop at the right spot along the platform.

Probably the best option is to help an ITR train stop properly by controlling its speed with speed limits in the Route Editor: slowing down the train with speed limits will make it easier for it to stop where you want. (But remember that restricted speeds and wait points set in the Activity Editor have no effect on ITR trains.)

You can also affect the stopping time (as in a normal activity) by adjusting the number of passengers at that platform. This can be done either in the Activity Editor or in the Route Editor (by changing the platform's properties). Note that the resulting stopping time depends on the number of passenger cars in the train!

## HOW DO I MAKE AN ITR TERMINATE WITH A LONG STOP RATHER THAN ABRUPTLY?

If the last stop is at a platform, you can cause a (very) long stop by increasing the number of passengers to 9999: then the user can terminate the ITR whenever desired, by pressing Esc.

## HOW DO I HANDLE REVERSE POINTS IN PATHS?

It appears that the engine <u>starts</u> reversing when it hits the reverse point that you place in the Activity Editor: that means that the ITR train will start slowing down at the reverse point, come to a stop and only then back up. So you must fine-tune this, and perhaps add track length (in the Route Editor) to allow the ITR train to stop before the end of the track (however, be careful not to create broken paths as you move the reverse points around in the Activity Editor!).

If necessary, you can place extra tracks underground (in the Route Editor), so they are invisible.

A reverse point placed too close to a turnout may terminate the ITR, as if the ITR had ended normally.

## ARE WYES AND REVERSAL LOOPS ALLOWED IN ITRs?

As in normal activities, wyes are allowed, but reversal loops are not.

## HOW DO I HANDLE SIGNALS?

Normally, signals work as you would expect. However, when combined with a reverse point, a signal may block the way for the ITR train, thinking that the same train is also still approaching (as if it saw two trains instead of one). You may need to delete a signal or rethink your strategy in such cases. These issues are not fully understood, so you may need to do some experimenting here.

[NEW SINCE V1.106]

# 6. DOCUMENTATION AND ILLUSTRATION

If you distribute your route, you will need to provide at least some documentation, especially to instruct the user on how to install the route on his or her computer, and to describe your route for download from a web site, with a small "advert".

Furthermore, you may want to provide some illustrations, for example a thumbnail for posting the route at a website; you could also produce a logo, a map screen and a load screen visible when the route is selected in MSTS to illustrate its character and provide additional information.

## 6.1 Documentation

### 6.1.1 ADVERT AND THUMBNAIL

To post your route for download from web sites like Train-Sim.com and Avsim.com, you need to provide a short "advert" and optionally a thumbnail image.

The advert informs the potential downloader what to expect in your route. It should be as clear and specific as possible to maximize the number of satisfied users and minimize the number of users who download something they don't want.

**The advert should contain simple text with a short title.** It should be limited to 450 characters, according to the rules at Train-Sim.com (although we see many exceptions being accepted): that is very short and will require very compact statements!

**The advert should ideally do the following:**
- give a route title that is unique and characteristic for your route (such as "MSTS Tokyo-Osaka Shinkansen Route", but not "MSTS Northwest Line": every place on Earth is northwest of some other place, except the South Pole!);
- specify whether the package offers an update (self-contained), or a fix (requiring an earlier version);
- make clear that the contents of the package is a route (not rolling stock, or scenery objects, or only activities for a route);
- state which route it is, if the title is not sufficiently clear (if the title is "Shinkansen Route", but the route is limited to Tokyo-Osaka, make that clear);
- describe the route character (say whether it involves mainly switching, high-speed passenger runs, main line or branch line freight runs, mountainous stretches, etc.);
- specify the approximate length of the route (a 10 mile route may appeal to different people than a 1000 mile route);
- specify whether activities are included (other than the simplest ones that are needed to run on the route);
- make clear whether the route is freeware, shareware, donationware, a demo, or something else (the Train-Sim.com library has useful information on all this);

- specify whether add-ons are needed that are not included in the package and that a user needs to download separately (such as non-default trains used in your consists, or add-on scenery objects, etc.);
- list the author(s) (such as "By Roland Schmid.").

The advert file can be named something like newroute.txt (it is limited to 8 characters before the point); it should use the same name as your package file: newroute.zip.

Some web sites (like Avsim.com) also require that you provide a copy of this advert file newroute.txt called file_id.diz (it will contain exactly the same text, but is used by automatic software at some web sites). To make this file, copy newroute.txt, then rename the copy to file_id.diz.

The web sites will post your **thumbnail picture** (not required, but highly recommended) next to your advert text. This thumbnail picture is very useful to give a graphical impression of the character of your route. A screenshot from MSTS is convenient for this. Note that a thumbnail is quite small (150 pixels wide and at most about 150 pixels high), so don't expect to be able to show much ("a picture is worth a thousand words" may not be accurate here!). The thumbnail should be in GIF format, and should use the same name as the advert and package, such as newroute.gif. Be careful to make the image bright: many published thumbnails are too dark to be useful!

NOTE: Appendix J tells you how to get **screenshots** of your route, and methods to increase the amount of control you have over the cameras that you use for that purpose.

### 6.1.2 ROUTE INSTALLATION AND INFORMATION

**Installation instructions are the most important information that you should provide the user.** Remember that many users have little or no experience with manipulating files, so it is essential to make the installation as simple and foolproof as possible. See chapter 7 for more details on preparing a route for distribution and for information on the installation process.

It is important to offer a special installation file, which the user can easily find after opening the package: this is best provided as a *.txt file, called something like readme1st.txt or install.txt. You should make a special effort to attract the user to the installation instructions. Otherwise, he or she may try some incorrect installation method and run into severe trouble; this gives the author a lot of extra work, as he or she will be bombarded with requests for help (and worse)!

To make the route more interesting, you could supply **background information** about it. This is especially true of routes that model real-life tracks. That way you can add a lot of atmosphere to your route.

You should explain in the documentation what **activities** you have prepared, and how to use them. This is especially useful for more complex activities.

Often, it may help the users to have **maps of your route**: these could be included within the documentation file or as separate images (see section 6.3).

Include in the documentation any **credits and acknowledgments** to people who have provided add-on objects or advice.

You should provide your name and contact information (usually an e-mail address), so users can contact you.

Documentation can be provided in various file formats. For simple documentation, a simple .txt file is sufficient. For more complex documentation, a Word document is generally fine; however, there may be a few users who don't have Word, so it is a good idea to include a *.txt version of your Word document. Another popular format is html, especially when images are included.

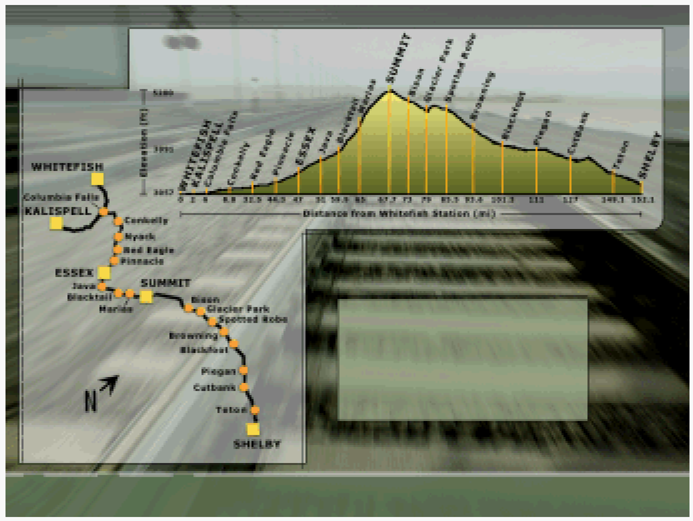
## 6.2 Front images

When a user selects a route in MSTS, a route logo is displayed, if available. Also, a "Map" screen is accessible that can offer the user graphical or other information about the route, such as a route map or a route profile. After selection of the route, a load screen is displayed while the loading takes place: for the MSTS default routes this load screen is a slightly modified copy of the map screen, or could be any other graphic.

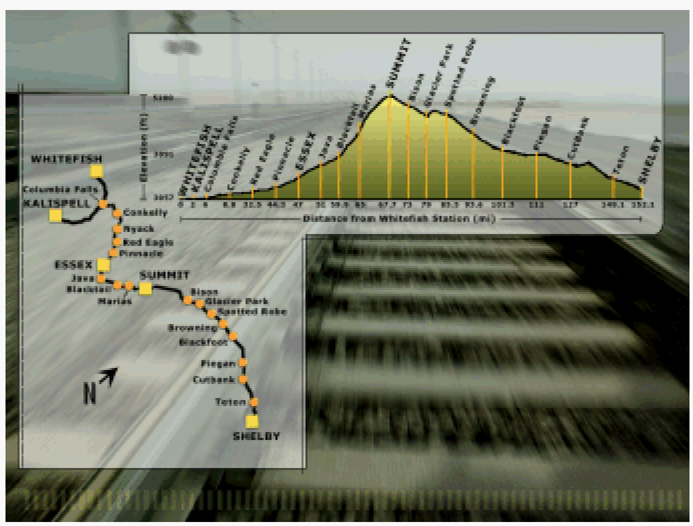
These images are shown in the next figure for the Marias Pass route (folder USA2), with their file names. Note that the logo (called bnsf.ace in that route) contains a graphic and an "alpha channel": this is a transparency image that softens its edges (it makes the edges transparent, so that the background behind the image shows through).

USA2: bnsf.ace


USA2: details.ace




USA2: load.ace




Note that complex map and load screens require large files.

None of these images is needed for a route to work properly. But they add character, information and a professional touch to your route.

### 6.2.1 IMAGE CONVERSION

To make these front images, you will have to start with BMP images and convert them to ACE format using the MakeACE utility that comes with MSTS. (The MakeACE utility was updated with the first patch of MSTS, so it is recommended that you install this patch, available at http://www.microsoft.com/games/trainsim/.) To make BMP images, you can use software like Microsoft Photo Editor, Paint, etc.

**The process is basically as follows** for each of the three pictures (I thank Pere Casulleras for teaching me this): you will first prepare a BMP image of proper size; then you will paste it into a bigger square blank BMP image (leaving blank space); this square BMP image can be converted and compressed by MakeACE to an ACE file (which will cut out the blank space again).

The reason for this circuitous approach is that MakeACE accepts as input only square bitmaps with certain sizes. (The Windows version MakeACEWin is not usable, because it does not accept 1024 x 1024 pixel BMP input.)

I have prepared **three batch files that will run MakeACE for the 3 files of interest here**. These batch files are included in this package: if you installed my First Route, you will find the following batch files in the Docs folder of **`First Route MV3`**:

MakeAce-LOGO-Bmp(256x256)ToAce(177x142).bat
MakeAce-DETAILS-Bmp(1024x1024)ToAce(800x600).bat
MakeAce-LOAD-Bmp(1024x1024)ToAce(800x600).bat

Copy them to the UTILS folder of MSTS; you may rename them to logo.bat, details.bat and load.bat, respectively, to match the file names used below.

If you wish, or if you can't find the three files given above, you can also prepare these batch files yourself as follows.

In any word processor, first type in the single line

```
MakeACE logo.bmp logo.ace -zlib -bmp -width:177 -height:142
```

Then save this to a text file named logo.txt. Next rename this file to logo.bat (which makes it a batch file), and place this file in the UTILS folder of MSTS.

Do the same for the other two batch files. The second should contain the single line:

```
MakeACE details.bmp details.ace -zlib -bmp -width:800 -height:600
```

Save this as file details.txt and rename it to details.bat, placing it again in the UTILS folder of MSTS.

The third file should contain the line:

```
MakeACE load.bmp load.ace -zlib -bmp -width:800 -height:600
```

Save it to load.txt and rename it to load.bat, placing it in UTILS.

### 6.2.2 LOGO (routename.ace)

The logo image will be contained in a file whose name can be defined in the route's *.trk file through the line `Graphic ( *.ace )`. The *.ace file can have any name you wish. By default, in a new route, this file is given the name of the route, such as NewRoute.ace, but there is no image in it.

We will use the name "logo.ace" for the process given below, because this particular name is required by the batch file logo.bat: otherwise it will not work. You may change "logo" to another name afterward.

**The logo image must have a size of 177 pixels in width by 142 pixels in height.** So you could first prepare a file logosmall.bmp with those dimensions. Alternatively and more directly, you could immediately define a 256 x 256 BMP image and restrict your drawing to the 177 x 142 pixel area at top left: save this bigger image directly as logo.bmp.

You can insert graphics from a screenshot taken in MSTS, for example; then, with suitable graphical software, you can extract a part of the image and resample it to give you these specific dimensions 177 x 142 pixels (at this stage, the image does not need to have exactly these dimensions: excess will be cut off later). Be careful to keep the same aspect ratio so the image is not distorted.

If you have created the 177 x 142 file logosmall.bmp, prepare a blank BMP image of size 256 x 256 pixels, and paste the image of logosmall.bmp into its top left corner (leaving blank space to the right and bottom): save this larger image as logo.bmp.

Then place logo.bmp in the UTILS folder of MSTS, and double-click on **logo.bat** (prepared as discussed in section 6.2.1). Close the batch window when it says Finished at top left (click on the X at top right to close the window).

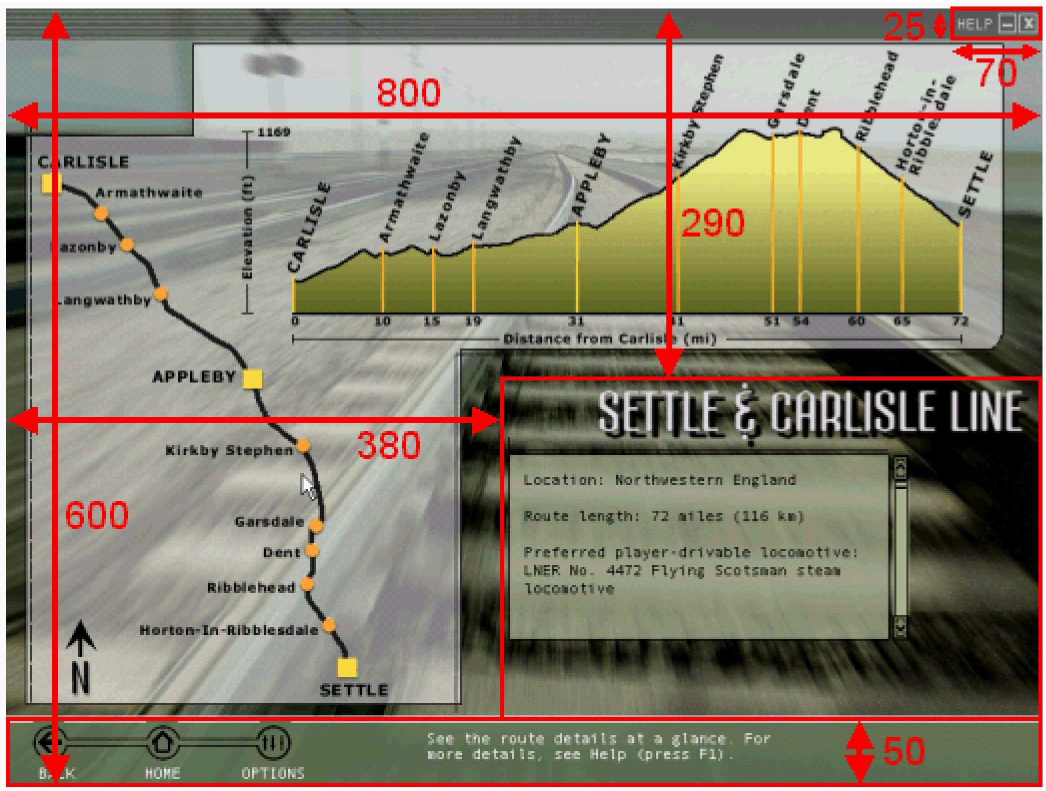
A file logo.ace has now been created. You may now rename it to something like NewRoute.ace, if desired, as mentioned above. Place this file in the root folder of the route, and make sure it has the same name as is declared in the route's *.trk file through `Graphic ( *.ace )`.

NOTE: MakeACE does not allow you to easily include a transparency of the type shown in the figure above. See Appendix H about transparency.

## 6.2.3 MAP SCREEN (details.ace)

The process for making a map screen is very similar to that given for the logo image (and no alpha channel is used). The main difference is that it will have **final dimensions 800 x 600 pixels** and that it should be added at the top left of a larger blank BMP file of 1024 x 1024 pixels. To convert to ACE format, you should prepare a file details.bmp (with 1024 x 1024 pixels), with your graphics limited to the top left 800 x 600 pixels. Then use **details.bat** to convert it to file details.ace. Place this file details.ace in the root folder of the route (you cannot change the file name for this map screen, as it does not appear in the route's *.trk file).

Before doing this, however, you should keep in mind that MSTS will write text and command buttons over your map screen, using your graphics as background. This is illustrated in the following screenshot of the map screen for the default Settle & Carlisle Route:




The buttons and text at top right and in the bottom margin are written by MSTS over your map screen (I have enclosed them in red boxes). So are the route name ("Settle & Carlisle Line" in the screenshot) and the box text, also enclosed in a red box.

Note that the route name and the box text are generated automatically by MSTS from information that you supplied earlier: you should not include them on your map screen. However, the bare box itself is part of the map screen: you may draw that box as well, but it is not necessary.

To allow the MSTS additions in the red boxes to be easily viewed against your map screen, you should provide an image that is neither white nor black in those areas that I have outlined in the screenshot. A medium-intensity background is best there, with gentle colors and reduced contrast preferable. I have shown pixel numbers on the screenshot to guide you in your own graphical design: these pixel numbers indicate the available space.

The main area available to show graphical information is the L-shaped space at top and left. This has a somewhat inconvenient shape, best used to display two or three separate graphs (or an L-shaped route). See section 6.3 for generating route maps.

### 6.2.4 LOAD SCREEN (load.ace)

The screen shown during loading of the route also has a size of **800 x 600 pixels**. It is created just like the map screen (no alpha channel is used). You should place it at top left in a 1024 x 1024 BMP file named load.bmp and convert it with **load.bat** to load.ace.

You may subsequently change its name to anything else (but different from other *.ace files in the same folder), since its name is adjustable in the route's *.trk file through `LoadingScreen ( *.ace )`.

One difference compared to the map screen is that you need not worry much about overwriting by MSTS onto the load screen: the whole 800 x 600 pixel screen is available for display. However, it is good to avoid yellow and white in the bottom margin, because that is where the yellow load progress bar is drawn and the white progress text is written.

On this load screen, you may put another copy of the map screen (as is done in the MSTS default routes), or anything else you wish, such as one or more screenshots from your route.

## 6.3 Route Maps

One weakness of MSTS is the lack of detailed maps for its default routes. Especially useful would have been maps showing the tracks and switches in stations and yards. You can fairly easily generate such maps for your own new route, after it is completed. If you distribute your route to other users, they will be very happy to have a route map to guide them, especially if your route is complicated.

Note however that a route map that shows sufficient detail in stations and yards will normally be too small to show the overall route. It may require many maps at different scales to clearly show everything.

You have seen a route map in the Activity Editor: you can use that (after choosing which features are drawn on it).

One way to generate a route map is to **make a screenshot of the map from the Activity Editor** (press Alt-PrintScreen), and then to paste it into a graphics program. Using that graphics program, you may write further explanations on it. If the route is long and stretched out, you may need to break up the route map into parts.

Store and distribute the resulting map in compressed format: gif is best; jpg may compress more, but reduces the quality of line figures. (Never distribute figures in bmp format, since they are far too large.)

## 7. DISTRIBUTION OF YOUR ROUTE

Distributing a route to users means the following:

- you must package those files which the user needs from you;

- you should avoid including those files which the user already has within MSTS, to minimize the size of the package that will be distributed;

- you must include instructions (documentation) on how to install the route on the user's computer;

- you can add any information you like, such as a description of the route and its activities, as well as a route map;

- a thumbnail picture and a short "advertisement" text are usually needed if you upload the package to one of the major train simulation sites (such as Train-Sim.com or Avsim.com - see their instructions on their sites);

- include your name and contact information;

- specify whether this package is freeware, or something else.

See chapter 6 for more discussion of the information that you should or could provide users.

If you are planning to make and distribute a route for MSTS commercially or as payware, you should be aware of the limitations imposed by the Train Simulator EULA (End-User License Agreement). Basically, if you distribute a route for money, that route may not use any of the default objects, textures, transfers, etc.; the only exceptions are the track and road pieces (you must supply your own textures for the tracks and roads). This means that you may not ask the user of your route to import files from the MSTS default routes into your route. Instead you must supply a completely new set of such files with your new route: so you must create and supply your own set of objects, textures, transfers, etc.

[NEW SINCE V1.106]

### 7.1 Route and file size

**Minimizing the size of a route saves hard-disk space and transfer time over the web.** Since routes tend to require much disk space and transfer time, it is useful to pay attention to their size. So it is useful to understand what affects the size of a route, such as number of tiles, number of objects placed, number of objects used, etc.

The more critical issue is transfer time over the web, for routes that are distributed to other users. This aspect is discussed in detail in section 7.2, dealing with the packaging and distribution of routes. The basic objective there is to avoid sending those files that the end user already has on his/her hard disk as part of default MSTS routes; this makes a very large difference in transfer time, because typically one half of the size of a route is composed of textures that the end user already has (in fact, after compression into a zip file, almost the entire resulting file size may be due to textures!).

**In the following, we focus on the disk space used by a route that is ready to run in MSTS.**

To better understand the space requirement of a route, let's look first at a typical default route, EUROPE1 (the Carlisle-Settle route). The next table lists the sizes of the subfolders in that route. (Note that I quote folder size as opposed to disk space used: Windows needs something like 15-30% more disk space than the file size would indicate.)

| EUROPE1 total folder size | 150 Mb |
|---|---|
| Activities/Paths/Services/Traffic | 1 Mb |
| Environment | 1.5 Mb |
| Shapes | 1.7 Mb |
| Sound | 8 Mb |
| Td | 0.4 Mb |
| TerrTex | 30 Mb |
| Textures | 45 Mb |
| Tiles | 57 Mb |
| World | 1.1 Mb |

(See Appendix F for a description of what each folder contains.)

**Textures** (that is most of the graphics used in a route) **take about** 30 + 45 = 75 Mb, or **half of the size of this route**. This size does not directly depend on the length of the route, and is therefore **essentially independent of the number of tiles** that the route occupies; it is also **independent of the number of copies of each object** that appear in the route.

**The next biggest part is the Tiles folder, which typically amounts to 1/5 to 1/2 of the size of a route**: this basically defines the terrain shape, water, etc. for the route, tile by tile, and thus increases directly with the number of tiles.

**The World folder, although usually small compared to the Tiles folder, also increases with the number of tiles**: it lists all objects in each tile (including tracks, roads, buildings, trees, sounds, etc.). More importantly, the World folder **increases directly with the number of objects** that are placed in the route; this includes each copy of each object, so that dense scenery will directly increase the size of the World folder.

The other folders are usually of lesser importance in terms of size: they either don't depend much on the nature of the route, or are normally small.

Let's now **compare the critical subfolder sizes for several default and add-on routes**, and let's consider the number of tiles in each, as shown in the next table.

| Route\Folder size (Mb) | Total | TerrTex | Textures | Tiles | World | No. of tiles | Total size per tile |
|---|---|---|---|---|---|---|---|
| EUROPE1 | 150 | 30 | 45 | 57 | 1.1 | ~100 | ~1.5 |
| EUROPE2 | 200 | 39 | 74 | 49 | 0.7 | ~80 | ~2.5 |
| JAPAN1 | 140 | 7 | 74 | 34 | 1.6 | ~60 | ~2.3 |
| JAPAN2 | 169 | 19 | 78 | 50 | 1.1 | ~100 | ~1.7 |
| USA1 | 246 | 12 | 77 | 128 | 2.1 | ~230 | ~1.1 |
| USA2 | 269 | 22 | 76 | 133 | 3 | ~230 | ~1.2 |
| Albula Line 2 | 205 | 40 | 99 | 28 | 5.5 | 58 | ~3.5 |
| Ohio RR | 510 | 26 | 263 | 139 | 1.3 | ~250 | ~2.0 |
| TimeSaver | 22 | 2.6 | 17 | 0.5 | 0.01 | 1 | ~22 |
| Cliff Hanger | 16 | 0.4 | 11 | 0.5 | 0.5 | 1 | ~16 |
| First Route MV3 | 39 | 5 | 16 | 4 | 0.5 | 4 | ~10 |

The default MSTS routes differ in a few ways. The USA1 and USA2 routes are about twice as long as the others (they use more than double the number of tiles, and their Tiles folders are proportionally large); their density of scenery (size of the World folder compared to the number of tiles) is average; but, because they use roughly the same amount of textures as the other default routes, the total size per tile ends up being smaller than average. JAPAN1 is the shortest of the default routes: although it uses fewer than average textures, its few tiles give it a relatively large total size per tile.

The largest of the tabulated routes is Ohio RR (by Richard Garber): its large size is mainly due to a large number of textures (both from the default MSTS routes and for add-on objects); not separately tabulated are another 80 Mb for its many Shapes, its on-line music, various sounds, a large manual and other extras; but its density of objects is below average (its World folder is small compared to the number of tiles); so the total size per tile remains average. Albula Line 2 (by me), a route still under construction, has many textures and a very high density of objects (very large World folder relative to the number of tiles); this gives it an above-average total size per tile; however, many of the 10 Mb of shapes and 140 Mb of textures actually will not be used, so that this route could be compacted much better: this will be done before distribution.

The smaller routes (TimeSaver by Ron Paludan, and Cliff Hanger and First Route MV3 by me) are extreme but informative examples. Both occupy just one tile, so the textures dominate their size, giving a large size per tile. Cliff Hanger in fact uses almost none of its textures (because it uses no objects besides tracks), wasting some 7Mb of space, but it has a very high density of tracks in that one tile, hence the respectable World folder size. By contrast, TimeSaver has few tracks, but a good amount of objects.

I have also listed my First Route for comparison. This is version 3, which adds about 1.5 Mb of graphics for the start-up screens and about 3 Mb of additional environmental textures in its ENVFILES\TEXTURES subfolder, compared to its version 2: these graphics make the zip file for distribution grow from 0.24 to 4.5 Mb between versions 2 and 3!

We can draw **some lessons** from these observations:

- **textures are normally the largest component of a route**:  this is where the most attention to size should be directed;  this can mean reducing the variety of objects that the route uses (for example by using more copies of fewer objects), and eliminating all those objects that are not actually used;  also make sure the textures are compressed as much as possible;  note that textures are normally already compressed, so that they rarely compress further when being placed in a zip file, unlike most other files;

- **the second largest component of a route is usually the Tiles folder, which is proportional to the number of tiles in the route**;  this can be reduced by avoiding unused tiles (see the discussion of fringe and unused tiles in section 2.2);  You could also use a higher Terrain Detail Scaling Factor (before starting building the route);  and, if you have the freedom to do so, you can reduce the number of tiles (without reducing the length of your route) by bending your route around so it uses the same tiles more than once (perhaps behind a hill);

- **typical routes require 1-4 Mb per tile** (less for routes with few objects, more for routes with few tiles);  this number will tend to decrease if you increase the number of tiles, because the largest size component comes from the textures, which normally remain about constant as a route is lengthened.

**One effective way to minimize the size of a route is to delete unused objects and especially textures.**  There is no simple way to do that within RE, but **some add-on utilities can help you:**

With **Route Control** (available as a free demo or complete payware at http://www.howintheworld.com/routecontrol.shtml), each add-on route that you install takes an average of only 10MB of space;  this is achieved by a form of archiving that eliminates all the files that can be recovered from the default routes.  You must "unarchive" the route before using it again.  As its author states, "note that you might occasionally come across a route that simply does not work properly with Route Control.  Due to the many different ways routes are put together there are just some times that a route ends up not being compatible.  Over 98% of routes released so far work fine in Route Control".

Another, more convenient option is the freeware called **Route-Riter** by Mike Simpson (available from Train-Sim.com).  Route-Riter can inspect a route to eliminate all objects and textures that are not used by a route, leaving a fully working version of the route.  This often saves large amounts of space.

## 7.2 Preparing the files to be distributed

Make sure you have minimized the number of tiles in your route:  see section 2.4 about deleting tiles.

Next you should identify those files that you must assemble for distribution to other users of MSTS.  Also minimize the number of these files, by avoiding files that all users already have.

[NEW SINCE V1.106] This can be done in two ways:

- an "automated" method for routes that don't use unusual "tricks" (such as replacing some terrain textures with others to obtain a heavy-snow ground cover in winter instead of the standard light-snow ground cover); this automated method uses the freeware add-on Route-Riter, described in section 7.2.1;

- or a much more laborious "manual" method for more "tricky" routes; this manual method is described in sections 7.2.2 and 7.2.3.

[NEW SINCE V1.106]
### 7.2.1 USING THE ADD-ON FREEWARE ROUTE-RITER

A very convenient and effective tool is available to "compress" a route and to produce a batch file to help other users install the route: it is the software called Route-Riter, written by Mike Simpson and available for free download from Train-Sim.com.

Route-Riter has two separate but connected functions (the second one requires the first one being done first):

- Make a working minimized route: Route-Riter can be used to delete all unused files, and compress the world tile files: the result is a route that occupies minimum space on your hard drive, and works properly in MSTS (and RE); this version, however, is still not suitable for distribution, because it normally contains many files that are common with default routes.

- Make a reduced (non-working) route folder, with a batch file for distribution: Starting from the working minimized route, Route-Riter will delete all files that are available from MSTS default routes, and replace each with a copy command in a batch file (called Installme.bat); you should package the remaining files and include the batch file, so the end user can assemble all the needed files and assemble a working minimized route.

Before you use Route-Riter, it is important that you back up a copy of your route, because Route-Riter will change your route folder contents. This is not only a safety measure, but also enables you to later continue building your route with objects that you already imported but did not use yet: you may have imported objects that you intend to use later and don't want to import a second time, so you may prefer to continue building your route from its original non-minimized version. (To run both the minimized and the non-minimized versions in MSTS, you could change the name and folder of one of them, as explained in section 8.1: for instance, if you have a minimized First Route, and want to further develop a non-minimized version, you could call this version First Route 2.)

When using Route-Riter, you should be aware that it may not be able to handle special situations, in particular "tricks" that you play with files to generate unusual effects. An example is the copying of texture files that have the same name but different contents (such as to produce heavy-snow textures instead of light-snow textures). Usually the textures are the same from route to route, but there are many exceptions, such as acleantrack1.ace, which is used to give tracks a different appearance on different continents. For instance, take the file tree4.ace: it is present in 8 seasonal and other versions in each of the 6 default routes, so there are altogether 48 files named tree4.ace, each of which could in principle contain different textures. If you copy such textures onto each other, Route-Riter will not be able to detect it, so that you may end up with different textures in the reassembled route.

So it is important to test-drive your route in different seasons (especially summer and winter, and also night) after creating the batch file for distribution: this allows you to check that no textures are missing or have been changed by Route-Riter. If you do notice a change, you will have to manually edit the batch file (with a normal text editor) to prevent that change.

As mentioned, Route-Riter will produce a reduced (non-working) route folder, containing the only files that you need to send to other users. It includes the Installme.bat batch file that the user will need to reassemble the route. You may also add files with installation instructions (see section 7.3) and other documentation.

NOTE: Don't forget to somehow include any non-default consists (trains) that your activities require: you could place them in a new subfolder of your route called CONSISTS, and arrange fro them to be copied during installation from there to the MSTS TRAINS\CONSISTS folder.

You should then compress the resulting reduced (non-working) route into a zipped file, including path names (depending on the zipping utility that you use, you might right-click on the route folder name, and then select Send To… Winzip or similar, etc.).

Finally, and most importantly, you need to test your own package and installation instructions: see section 7.4.

### 7.2.2 HOW MSTS ORGANIZES A NEW ROUTE

If you need to "manually" prepare a route for distribution, it is important for you to understand how MSTS organizes a new route when you create it.

When creating a new route, the Route Geometry Extractor places a copy of the MSTS TEMPLATE folder into the ROUTES folder, and gives it the name of the new route, for example "NewRoute": this copy of the TEMPLATE folder, renamed to NewRoute, will thus be at the same level as the default MSTS route folders (EUROPE1, etc.).

At this stage, your folder structure should look something like this (if you look at it with Windows Explorer or My Computer):

```
Train Simulator
... 1033
... FONTS
... ...
... ROUTES
...... EUROPE1
...... EUROPE2
```

```
...... ...
...... NewRoute <--- copy of TEMPLATE folder renamed to NewRoute
...... ...
...... USA2
... SAMPLES
... ...
... TEMPLATE <--- source TEMPLATE folder
... TRAINS
... UTILS
```

After you have defined the geographical location of the new route, the Route Geometry Extractor places some new files in subfolders of this folder NewRoute:  for example, the new tiles on which the route will reside are placed within its TILES subfolder.  When you lay tracks, add and place objects, etc., you change some of those files and add others.  Those new files inserted by the Route Geometry Extractor, and any files modified by you, are the ones that you must send to other users;  there is no need to send the files that are left unchanged in the NewRoute folder.

[NEW SINCE V1.106] Appendix F gives a more complete listing of what kinds of files exist in the different subfolders of a route.

### 7.2.3 SELECTING THE FILES TO DISTRIBUTE

One convenient way to select the files for distribution is the following (using Windows Explorer or My Computer):

- copy the entire NewRoute folder to a location outside the MSTS folder (called Train Simulator), such as C:\temp\NewRoute (but don't overwrite the backup copy that you made earlier, if you still want it!);

- from here on, work only on this copy located outside the MSTS folder, leaving the original untouched;

- **with two exceptions**, **delete all <u>old</u> files from the copied NewRoute folder and all its subfolders** (these are the original Microsoft files with dates prior to your receiving MSTS):  **the two exceptions are NewRoute.ref and NewRoute.ace**, which you will find in the top level of the NewRoute folder (again NewRoute is only an example name:  use the name of your route here);  these two files may have old dates (if you did not change them by adding objects or changing textures), but they are needed by the end user, so don't delete them;

- **in the SHAPES subfolder, delete all *.thm files** (they are only thumbnail pictures used for display in the Object selector, created each time the user asks for them);

- **in the TILES subfolder, delete all *_e.raw and *_n.raw files** (they will be regenerated the first time the end user loads your route into MSTS or RE);

- **delete all backup files (*.bk)**;

- **delete any empty subfolders** within the NewRoute folder; normally you will only be left with the subfolders ACTIVITIES, PATHS, SERVICES, TD, TILES, WORLD, and a few files directly in the NewRoute folder; if you added objects from non-default routes (downloaded from the web, for example), you would still have their files in the subfolders SHAPES and TEXTURES; if you changed the environment, you would have files left in the ENVFILES subfolder and perhaps the ENVFILES\TEXTURES subfolder; if you created low-resolution distant mountains (from DEM data), they would be left in the LO_TILES subfolder;

- **you could add a subfolder named CONSISTS** within the NewRoute folder, where you can place any non-default consists (trains) that the user will need with your activities;

- **you could add a subfolder named Docs** within the NewRoute folder to include any documents you wish the end user to have in a convenient location, such as installation instructions, a route and activity description, and a route map;

- **compress the resulting NewRoute folder into a zipped file**, including path names within the NewRoute folder (depending on the zipping utility that you use, you might right-click on the NewRoute folder name, and then select Send To… Winzip or similar, etc.).

## 7.3 Installation instructions for the end user

The users who install your route on their computers will have to do a few simple steps: you should tell them these steps, after tailoring them to your route's requirements.

The steps will depend on whether you packaged the route with the automated method of Route-Riter (section 7.2.1), or with the manual method (section 7.2.2 and 7.2.3).

[NEW SINCE V1.106]

### 7.3.1 AUTOMATED METHOD USING ROUTE-RITER

If you prepared your route with Route-Riter, the user should:

- unzip the reduced (non-working) route from the zip file to a new folder in the MSTS ROUTES folder: the new route folder should have the same name that you also used for that folder;

- run the batch file Installme.bat (in order to copy all needed files from other default routes); make sure that any non-default consists are copied to the MSTS CONSIST folder.

### 7.3.2 MANUAL METHOD

If you used the manual method of preparing your route for distribution, the users who install your route on their computers will have to do the following steps:

- copy the TEMPLATE folder of MSTS into the ROUTES folder of MSTS (using Windows Explorer or My Computer): this copy of the TEMPLATE folder should be at the same level as the default MSTS route folders (EUROPE1, etc.);

- rename that copy of the TEMPLATE folder to NewRoute (or whatever folder name your route has): so the folder NewRoute will be at the same level as the default MSTS route folders (EUROPE1, etc.); the resulting folder structure should look something like this:

```
Train Simulator
... 1033
... FONTS
... ...
... ROUTES
...... EUROPE1
...... EUROPE2
...... ...
...... NewRoute <--- copy of TEMPLATE folder renamed to NewRoute
...... ...
...... USA2
... SAMPLES
... ...
... TEMPLATE <--- source TEMPLATE folder
... TRAINS
... UTILS
```

- unzip the files from the zipped package into the new NewRoute folder: make sure to unzip the packaged files into this new NewRoute folder using path names (accept any overwriting that may occur);

- copy any needed files from MSTS default routes to the corresponding subfolders of NewRoute;

- copy any non-default objects to the route's SHAPES and TEXTURES subfolders;

- copy any non-default consists to the MSTS TRAINS\CONSISTS folder.

These steps may be automated for the user by supplying a batch file (*.bat) that the user can execute under Operating Systems that understand DOS commands (such as Windows 2000, Me and XP). This batch file should contain DOS-type copy commands to place the needed files in the correct folders. As an example, such a .bat file for installing the "First Route" project of this guide would look in part like this (omitting here a number of terrain textures, which are mentioned above in the First Route building project):

```
copy JAPAN1\Sound\crossing.sms FirstRoute\Sound\crossing.sms

copy EUROPE1\ssource.dat FirstRoute\ssource.dat
copy EUROPE1\ttype.dat FirstRoute\ttype.dat
copy EUROPE1\Sound\*.* FirstRoute\Sound\*.*
copy EUROPE1\Sound\gen_uk_nat1.wav FirstRoute\Sound\gen_uk_nat1.wav

copy USA2\Shapes\US2Crossx1Nbar.s FirstRoute\Shapes\US2Crossx1Nbar.s
copy USA2\Shapes\US2Crossx1Nbar.sd FirstRoute\Shapes\US2Crossx1Nbar.sd

copy USA2\Textures\crossing.ace FirstRoute\Textures\crossing.ace
```

```
copy USA2\Textures\Snow\crossing.ace FirstRoute\Textures\Snow\crossing.ace

REM note change of file name (UKHedge1.ace to terrain.ace) in next two lines!

copy EUROPE1\Terrtex\UKHedge1.ace FirstRoute\Terrtex\terrain.ace
copy EUROPE1\Terrtex\Snow\UKHedge1.ace FirstRoute\Terrtex\Snow\terrain.ace

copy EUROPE1\Terrtex\UKHedge1.ace FirstRoute\Terrtex\UKHedge1.ace
copy EUROPE1\Terrtex\Snow\UKHedge1.ace FirstRoute\Terrtex\Snow\UKHedge1.ace
copy EUROPE1\Terrtex\OEDirt.ace FirstRoute\Terrtex\OEDirt.ace
copy EUROPE1\Terrtex\Snow\OEDirt.ace FirstRoute\Terrtex\Snow\OEDirt.ace
...
... CAUTION: lines omitted here!
...
copy EUROPE1\Terrtex\OEDirt.ace FirstRoute\Textures\OEDirt.ace
copy EUROPE1\Terrtex\Snow\OEDirt.ace FirstRoute\Textures\Snow\OEDirt.ace
```

(You would replace each FirstRoute in this list by your route's folder name, such as NewRoute.)

This batch file could also include commands to copy the contents of the TEMPLATE folder to the new route's folder.

You can prepare such a .bat file with any word or text processor that allows exporting in simple text (ASCII) format.  Export to the file NewRoute.bat (or something similar;  if necessary, first export to NewRoute.txt and then rename the file to NewRoute.bat).  The user should install this NewRoute.bat file in the ROUTES folder of Train Simulator (not elsewhere), then double-click on this file in Windows Explorer or My Computer.  The batch file will then make all the needed copies.

NOTES:

- a user may **not** have installed all default routes, in which case the last step may require copying files from the MSTS CD's;

- in the batch file, you can easily cause all files of a subfolder to be copied with a single command line, using wildcards;  for example:

```
copy EUROPE1\Terrtex\*.* NewRoute\Textures\*.*
```

- a freeware batch file is available to copy all files from all default routes to your new route's folder:  it is called ezstuff2.bat, by Jim "Sniper" Ward, and is contained in gitstufd.zip, which is downloadable from the Train-Sim.com library;

- it is not necessary to supply the appropriate *.ref file, which declares all the objects and transfers that your route will require;  the *.ref file is only used by the Route Editor for placing new objects (thus allowing the user to modify the route);  it is not used by MSTS to run the route;  however many users (and especially route builders) like to be able to explore your route within the Route Editor or to modify your route for their own purposes:  they need the *.ref file to do this.

After this installation, the new route should show up as a route option (such as NewRoute) when the user starts MSTS and selects Drive a Train, and its Activities should show up as well.

To completely **uninstall your route**, the user can simply delete your route's entire folder (such as NewRoute) from the MSTS routes folder (using Windows Explorer or My Computer);  in addition, the user would have to delete any new consists that your route needed.

## 7.4 Testing your installation instructions

VERY IMPORTANT:  **Test your own installation instructions!  If possible, have your route beta-tested by others.**

As you may have noticed, many users complain about problems they encounter when installing new routes in MSTS.  The main causes are incorrect or unclear installation instructions (of course some people don't even read instructions, even after the first failure, but they are the minority!).  Given that hundreds and maybe thousands of people may try to install your route, you can avoid a huge amount of lost time, frustration and criticism by making sure that your instructions are correct.  Answering unclear instructions will most likely take much more of your time than creating clear instructions.

How do you test your own installation instructions?  That is quite easy, thanks to the simple organization of MSTS:

- if you have not done so yet, back up your route (copy the entire folder NewRoute, if that is your route's name, to some other safe place);

- delete the entire NewRoute folder containing your new route in MSTS (now MSTS has no trace of your route:  it is as if your route never existed, just as it will be for a new user preparing to install your route);

- now follow your own installation instructions **to the letter (not from memory!)**;

- load your route into RE, and move along your route (to check that no files are missing);

- start MSTS and see whether your route works correctly (by driving around, including in winter and at night).

If your route does not work, revise your installation instructions and repeat this installation test until it does work.

If your route does work, I suggest that you delete the installed copy and bring back your back-up copy into the MSTS ROUTES folder (just in case it included some files not needed for distribution, but needed in your future development of this route).

You are now ready to upload the compressed route file for others to use.

I strongly encourage you to **have your route beta-tested**, before releasing the route for general distribution.  Many things can go wrong despite your own testing, especially on different computers with different operating systems (different versions of Windows) and different installations of MSTS.  Among the most difficult aspects to test yourself is your use of consists (trains):  since consists are separate from your route, but used by your route's activities, it is easy to overlook that you may be using consists that others do not have on their computers;  beta-testers will discover such a problem very quickly.  If you don't know who to ask for beta-testing, place an invitation in a MSTS-related Forum for volunteers to beta-test your route, or ask recommendations from the authors of routes that you have downloaded.  Surprisingly many people are happy to beta-test!

## 8. RENAMING, COPYING AND DELETING ROUTES

Once you have created a new route, you may want to change the name(s) and description you chose for it: there are several names to consider for a given route, as explained in section 8.1.

You may also wish to copy a route, either to create another route derived from it, or for safekeeping as a backup. This is discussed in section 8.2.

And you may want to delete a route, either because it has problems and you want to replace it with a backup copy, or because you don't want to use it any more and you need to reclaim the disk space that it occupies. This is addressed in section 8.3.

### 8.1 Renaming a route

**A route is identified by several names and descriptions**, illustrated here with two cases, the First Route and the Settle & Carlisle Line (Europe1):

- a folder name, called RouteID (this name is also used in Activity files of type *.act):
    FirstRoute
    EUROPE1

- a Name that will be displayed in the route lists of both RE and MSTS (the quotation marks are needed because of the spaces in these names):
    "First Route"
    "Settle & Carlisle Line"

- a Description that will be displayed in MSTS on the Route and Activity Selection screen after you highlight a route:
    "" (no description was entered for First Route)
    "Location: Northwestern England…" (a long description of the route, broken into new lines by \n\n)

- a Graphic file name with a picture displayed in MSTS (this type of file cannot be made with normal graphics software):
    FirstRoute.ace (this file is empty for new routes, unless you supply it)
    scotsman.ace (showing a Flying Scotsman logo)

- a FileName that is used for all the route-specific files contained in the main folder of the route (such as the files *.ace, *.ref, *.tdb, etc.):
    FirstRoute
    SettleCa

These names are all defined in the route's *.trk file, as shown next, with the relevant lines highlighted in red.

The file First Route MV3.trk looks like this:

```
SIMISA@@@@@@@@@@JINX0r1t______

Tr_RouteFile (
	RouteID ("First Route MV3" )
	Name ("First Route MV3" )
	Description ( " This is version 3 of \"First Route\", a route building
project that is part of the \"Step-by-Step Guide to Building Routes for
Microsoft Train Simulator\" by Michael Vone, published by Abacus.\n " )
	Graphic ( "First Route.ace" )
	LoadingScreen ( Load.ace )
	FileName ( "First Route MV3" )
	Electrified ( 00000001 )
	Mountains ( 00000000 )
	OverheadWireHeight ( 5.4 )
	PassengerRuleSet ( 0 )
	FreightRuleSet ( 0 )
	SignalSet ( 0 )
	GantrySet ( 0 )
	TrackGauge ( 0 )
	Era ( 0 )
	SpeedLimit ( 35.7632 )
	Environment (
		SpringClear ( springclear.env )
		SpringRain ( springrain.env )
		SpringSnow ( springsnow.env )
		SummerClear ( summerclear.env )
		SummerRain ( summerrain.env )
		SummerSnow ( summersnow.env )
		AutumnClear ( autumnclear.env )
		AutumnRain ( autumnrain.env )
		AutumnSnow ( autumnsnow.env )
		WinterClear ( winterclear.env )
		WinterRain ( winterrain.env )
		WinterSnow ( wintersnow.env )
	)
	TerrainErrorScale ( 1 )
	RouteStart ( -6079 14928 -64 64 )
	DefaultCrossingSMS ( crossing.sms )
	DefaultSignalSMS ( signal.sms )
	DefaultWaterTowerSMS ( wtower.sms )
	DefaultCoalTowerSMS ( ctower.sms )
	DefaultDieselTowerSMS ( dtower.sms )
	TempRestrictedSpeed ( 6.7056 )
)
```

The corresponding file for the Settle & Carlisle Line is europe1.trk:

```
SIMISA@@@@@@@@@@JINX0r1t______

Tr_RouteFile (
	RouteID ( EUROPE1 )
	Name ( "Settle & Carlisle Line" )
	Description ( "Location: Northwestern England\n\n"+
		"Route length: 72 miles (116 km)\n\n"+
```

```
            "Preferred player-drivable locomotive: LNER No. 4472 Flying
Scotsman steam locomotive\n\n"+
            "Locomotive owner, circa 1930: London and North Eastern Railway
(LNER)\n\n"+
            "Locomotive owner, today: Flying Scotsman Railway\n\n"+
            "Computer-controlled trains on route: The Royal Scot (Royal Scot
Class), Pendennis Castle(Castle Class)\n\n"+
            "Built in the 1870s to provide a faster route for the growing
Midland Railway Company's traffic between England and Scotland, the Settle &
Carlisle Railway ('the S&C') is considered the most dramatic train line in
England. The line travels through Yorkshire Dales National Park and into the
Pennine Chain, skirting Lake District National Park to the west.\n\n"+
            "Train Simulator includes the entire line from Settle to
Carlisle as it was in the late 1920s. You'll pass through beautiful
countryside, where stone walls separate fields of barley, cow pastures, and
country lanes. You must skillfully control your steam as you climb up to the
wild Blea Moor, and then carefully make your way down the grade while keeping
your passengers safe and comfortable. Can you keep up with Flying Scotsman's
illustrious history of on-time arrival?" )
      Graphic ( scotsman.ace )
      LoadingScreen ( E1Load.ace )
      FileName ( SettleCa )
      Electrified ( 00000000 )
      Mountains ( 00000000 )
      OverheadWireHeight ( 0 )
      PassengerRuleSet ( 0 )
      FreightRuleSet ( 0 )
      SignalSet ( 0 )
      GantrySet ( 0 )
      TrackGauge ( 0 )
      Era ( 0 )
      SpeedLimit ( 46.9392 )
      Environment (
            SpringClear ( UKsun.env )
            SpringRain ( UKrain.env )
            SpringSnow ( UKsnow.env )
            SummerClear ( UKsun.env )
            SummerRain ( UKrain.env )
            SummerSnow ( UKsnow.env )
            AutumnClear ( UKsun.env )
            AutumnRain ( UKrain.env )
            AutumnSnow ( UKsnow.env )
            WinterClear ( UKsun.env )
            WinterRain ( UKrain.env )
            WinterSnow ( UKsnow.env )
      )
      TerrainErrorScale ( 1 )
      RouteStart ( -6113 15054 -629.371 -962.07 )
      MilepostUnitsKilometers ( )
      DefaultCrossingSMS ( crossing.sms )
      DefaultSignalSMS ( signal.sms )
      DefaultWaterTowerSMS ( wtower.sms )
      DefaultCoalTowerSMS ( ctower.sms )
      DefaultDieselTowerSMS ( "" )
      TempRestrictedSpeed ( 6.7056 )
      DerailScale ( 2 )
      TimetableTollerance ( 60 )
```

)

[NEW SINCE V1.106] (The LoadingScreen file contains the screen that you see while a route loads:  see section 6.3.  The file E1Load.ace is not present by default in the routes folders, and even cannot be found on the hard drive:  it is a mystery!)

[NEW SINCE V1.106] **If you only want to change the Route name and/or the Route description, you can do this simply within RE after loading the route:**

- select Route | Properties;
- change the Route name, if desired;
- change the Route description, if desired;
- click OK, and save the route.

**You can also make these and other changes directly in the *.trk file with a Unicode-capable word processor** such as WordPad.  This allows you to change the various names as you wish.  However, you must make sure that you change any related folder and file names at the same time.  Here are a few examples.

WARNING:  Make the following changes when NOT running RE or MSTS (or at least when not running the relevant route in RE or MSTS).

WARNING:  It is important to not change the parentheses within the *.trk files.

**To change only the displayed name of a route** shown in the RE and MSTS route lists, for example from First Route to North London, do the following:

- within *.trk, change the Name from First Route to North London.

NOTE:  This change of the displayed name can also be done with the RGE.  To do that, open RGE, select your route, select Edit, and then Route values:  change the "Route name seen by user" and exit from RGE (no saving is needed).

**To change only the folder name of a route**, for example from FirstRoute to NorthLondon, do the following:

- change the name of the *.trk file, from FirstRoute.trk to NorthLondon.trk;

- within the renamed *.trk file, change the RouteID from FirstRoute to NorthLondon;

- change the RouteID in all Activity files *.act (in the ACTIVITIES subfolder) from FirstRoute to NorthLondon (otherwise, starting an unchanged Activity can crash MSTS);

- change the name of the folder itself from FirstRoute to NorthLondon.

NOTE:  If you have copied a route (such as First Route) to make a second version (such as First Route 2), you will also have to change the displayed route Name.  This way, they will not have the same displayed name, and you can distinguish the two routes to load the one you wish. For example:

- within the renamed *.trk file, change the Name from "First Route" to "First Route 2".

If you wish, you may also make other changes, described in the next section on Copying.

**To change only the description of a route**, do the following:

- within the *.trk file, change the Description from an "Old description" to a "New description".

Of course, **you may combine the changes described above**, for example changing both the folder name and the description of a route.

## 8.2 Copying a route

You may **copy a route to make a different version** of this route. You do this as follows, for example to make First Route 2 from First Route:

- copy the route's complete folder (FirstRoute) with all its subfolders to a new folder;

- give the new route folder its new name, such as FirstRoute2;

- change the name of the *.trk file, from FirstRoute.trk to FirstRoute2.trk;

- within the renamed *.trk file, change the RouteID from FirstRoute to FirstRoute2;

- within the renamed *.trk file, change the Name from "First Route" to "First Route 2";

- you may change the Description as desired;

- you may change the FileName as desired: do that within the *.trk file and also change the names of all the files that have that route's name in the route's main folder (*.ace, *.ref, *.tdb, etc.);

- change the RouteID in each Activity file *.act (in the ACTIVITIES folder of the route) to equal the new folder name.

NOTE: If you copy a route and then change the tracks in the new version, the old activities will no longer be valid, so you will have to delete the old ones and create new ones. In that case, rather than changing names in the Activity files, delete these files (and any corresponding files in the PATHS and SERVICES subfolders).

**You may copy a route for backup**: for that you place a copy of the route's complete folder outside the MSTS ROUTES folder. The backup route cannot be accessed by MSTS. It may be helpful to change the backup's folder name so you can later identify which backup version it is.

**To retrieve a backup copy of a route**: delete the current version's complete folder from within the MSTS ROUTES folder (after making a backup copy of that, if you wish); copy the backed-up version into the MSTS ROUTES folder; if you changed its folder name for identification purposes, change that folder name back to what it was originally.

## 8.3 Deleting a route

**You can delete a route** very easily: delete the route's complete folder from the MSTS ROUTES folder. MSTS will see no trace of your deleted route. It is advisable to keep a backup if you wish to use the route again later.

(Note that Windows will temporarily keep a copy of the complete route folder in the Recycle Bin, occupying the same hard disk space until Windows or you clean out the Recycle Bin.)

## APPENDIX A. ROUTE EDITOR COMMANDS

The following key and mouse commands can be used in the MSTS Route Editor. The Mode column indicates in which mode RE should be to act as listed (a blank means any mode is OK).

### GENERAL COMMANDS

| Command | Pad | Mode | Action |
|---|---|---|---|
| F1 | | | opens Editors and Tools Help |
| Ctrl-S | | | save to file, after deselecting any selected track or objects |
| Shift-Z | | | show frame rate (Frames Per Second) |
| Page Up | | | increase track visibility range |
| Page Down | | | decrease track visibility range |

### CAMERA COMMANDS

| Command | Pad | Mode | Action |
|---|---|---|---|
| Up-arrow | ArrowPad | | move camera forward |
| Down-arrow | ArrowPad | | move camera backward |
| Left-arrow | ArrowPad | | move camera left |
| Right-arrow | ArrowPad | | move camera right |
| Ctrl+up-arrow | ArrowPad | | move camera up |
| Ctrl+down-arrow | ArrowPad | | move camera down |
| Shift | | | speed up arrow-key motions |
| End | | | slow down arrow-key motions |
| Home | | | move camera to new coordinates |
| / | | | toggle camera/terrain collision (allow camera to go underground) |

## CONTROL OF ROUTE EDITOR'S MODE OF OPERATION

| Command | Pad | Mode | Action |
|---|---|---|---|
| F2 | | | select "Select Object" mode |
| F3 | | | select "Move Object" mode |
| F4 | | | select "Rotate Object" mode |
| F5 | | | select "Place Object" mode |
| F6 | | | select "Edit Object Properties" mode |
| F7 | | | select "Terrain Texturing" mode |
| F8 | | | select "Draw on Textures" mode |
| F9 | | | select "Alter Terrain" mode |

## TRACK HANDLING

| Command | Pad | Mode | Action |
|---|---|---|---|
| F2 | | | select "Select Object" mode |
| F5 | | | select "Place Object" mode |
| F6 | | | select "Edit Object Properties" mode |
| | | | |
| G | | F2 | toggle placement restriction to 100m x 100m grid |
| O | | | reset original orientation of selected track |
| T | | | toggle selected track orientation |
| Y | | | fit terrain height to selected track section |
| | | | |
| Left-click | | F2 | select track section |
| Right-click | | | open pop-up menu for selected track section |
| Delete | | | delete selected track section |
| | | | |
| RightButton+MouseDrag | | | F4 slope selected track section up/down |
| Shift | | | speed up sloping |
| End | | | slow down sloping |

*For first or unconnected track sections only:*

| Command | Pad | Mode | Action |
|---|---|---|---|
| F3 | | | select "Move Object" mode |
| F4 | | | select "Rotate Object" mode |
| H | | | fit selected object height to terrain height |
| N | | | orient selected object perpendicular (normal) to terrain slope |
| Ctrl+MouseDrag | | F3 | move selected track section horizontally |
| Up-arrow | NumPad | F3 | raise selected track section |
| Down-arrow | NumPad | F3 | lower selected track section |
| Left-arrow | NumPad | F3 | move selected track section to left |
| Right-arrow | NumPad | F3 | move selected track section to right |

## OBJECT HANDLING

| **Command** | **Pad** | **Mode** | **Action** |
|---|---|---|---|
| F2 | | | select "Select Object" mode |
| F3 | | | select "Move Object" mode |
| F4 | | | select "Rotate Object" mode |
| F5 | | | select "Place Object" mode |
| F6 | | | select "Edit Object Properties" mode |
| G | | F2 | toggle placement restriction to 100m x 100m grid |
| H | | | fit selected object height to terrain height |
| N | | | orient selected object perpendicular (normal) to terrain slope |
| O | | | reset original orientation of selected object |
| X | | | toggle snapping-on of some objects |
| Y | | | fit terrain height to selected object |
| Left-click | | F2 | select object |
| Ctrl+Left-click | | F2 | select multiple objects |
| Right-click | | | open pop-up menu for selected object |
| Delete | | | delete selected object |
| Ctrl+C | | | copy selected object(s) to clipboard |
| Ctrl+V | | F5 | paste copied object(s) |
| MouseDrag | | F3 | move selected object left/right or up/down parallel to screen |
| Ctrl+MouseDrag | | F3 | move selected object horizontally |
| Up-arrow | NumPad | F3 | raise selected object |
| Down-arrow | NumPad | F3 | lower selected object |
| Left-arrow | NumPad | F3 | move selected object to left without tilting |
| Right-arrow | NumPad | F3 | move selected object to right without tilting |
| MouseDrag | | F4 | turn selected object left/right and/or tilt it northward/southward |
| Left-arrow | NumPad | F4 | turn selected object to left without tilting |
| Right-arrow | NumPad | F4 | turn selected object to right without tilting |
| Ctrl+Left-arrow | NumPad | F4 | turn selected object to left while keeping its altitude constant |
| Ctrl+Right-arrow | NumPad | F4 | turn selected object to right while keeping its altitude constant |

| | | | |
|---|---|---|---|
| Up-arrow | NumPad | F4 | tilt selected object northward |
| Down-arrow | NumPad | F4 | tilt selected object southward |
| Ctrl+Up-arrow | NumPad | F4 | tilt selected object westward |
| Ctrl+Down-arrow | NumPad | F4 | tilt selected object eastward |
| | | | |
| Shift | | | speed up motions |
| End | | | slow down motions |

## TERRAIN HANDLING

| **Command** | **Pad** | **Mode** | **Action** |
|---|---|---|---|
| F7 | | | select "Terrain Texturing" mode |
| F8 | | | select "Draw on Textures" mode |
| F9 | | | select "Alter Terrain" mode |
| | | | |
| Left-click | | F7 | select patch |
| Right-click | | F7 | open pop-up menu for selected patch/tile |
| | | | |
| D | | F9 | toggle mouse dragging of Alter Terrain tool |
| F | | F9 | smoothen ("flatten") selected terrain rectangle |
| J | | | refresh terrain and shadows |
| V | | F9 | toggle grid point (with lines and terrain) visible/invisible |
| W | | | toggle terrain wire frame |
| | | | |
| Up-arrow | NumPad | F9 | raise selected grid point |
| Down-arrow | NumPad | F9 | lower selected grid point |
| Shift | | | speed up arrow-key motions |
| End | | | slow down arrow-key motions |
| | | | |
| 1 through 9 | KeyPad | F7 | assign texture to selected patch |
| Space | | F7 | assign current texture to selected patch |
| Q | | F7 | cycle through textures |
| | | | |
| + (=) | KeyPad | | move Sun westward |
| - | KeyPad | | move Sun eastward |
| | | | |
| ` (back-quote) | | F9 | undo last terrain changes |

# APPENDIX B. TABLE OF AVAILABLE TRACK SECTIONS

The following table describes all the track sections that are available in the MSTS Route Editor (excepting the dynamical track section), grouped in categories. The lengths of curved sections are measured along the curved arcs and are approximate. At right are the track section names found in the Object selector.

| TRACK DESCRIPTION | LENGTH | NAME |
|---|---|---|
| | | |
| **Single Track - Straight** | | |
| | | |
| 7m long | 7m | A1t7mStrt.s |
| | | |
| 10m long | 10m | A1t10mStrt.s |
| 10m long + round tunnel | 10m | A1t10mStrtRndTun.s |
| 10m long + high tunnel | 10m | A1t10mStrtTun.s |
| 10m long + water trough | 10m | A1t10mStrtWtr.s |
| 10m long skew left | 10m | A1t10mSkew.s |
| | | |
| 50m long | 50m | A1t50mStrt.s |
| 50m long + round tunnel | 50m | A1t50mStrtRndTun.s |
| 50m long + high tunnel | 50m | A1t50mStrtTun.s |
| 50m long + water trough | 50m | A1t50mStrtWtr.s |
| | | |
| 100m long | 100m | A1t100mStrt.s |
| 100m long + round tunnel | 100m | A1t100mStrtRndTun.s |
| 100m long + high tunnel | 100m | A1t100mStrtTun.s |
| 100m long + water trough | 100m | A1t100mStrtWtr.s |
| | | |
| 250m long | 250m | A1t250mStrt.s |
| | | |
| 500m long | 500m | A1t500mStrt.s |
| | | |
| | | |
| **Single Track - Buffers** | | |
| | | |
| Single Buffer | 7m | A1tBuffer.s |
| Single US Buffer | 7m | A1tUSBuffer.s |
| | | |
| | | |
| **Single Track - Curves** | | |
| | | |
| ~67m radius 45° turn left yard | ~53m | A1t45dYardCrvLft.s |
| ~67m radius 45° turn right yard | ~53m | A1t45dYardCrvRgt.s |
| | | |
| 500m radius 20° turn | ~175m | A1t500r20d.s |
| 500m radius 10° turn | ~88m | A1t500r10d.s |
| 500m radius 10° turn + round tunnel | ~88m | A1t500r10dRndTun.s |
| 500m radius 10° turn + high tunnel | ~88m | A1t500r10dTun.s |
| 500m radius 5° turn | ~44m | A1t500r5d.s |
| 500m radius 5° turn + round tunnel | ~44m | A1t500r5dRndTun.s |
| 500m radius 5° turn + high tunnel | ~44m | A1t500r5dTun.s |
| | | |

| | | |
|---|---|---|
| 870m radius 4° turn left | ~61m | A1t870r4dLft.s |
| 870m radius 4° turn left | ~61m | A1t870r4dRgt.s |
| | | |
| 1000m radius 20° turn | ~349m | A1t1000r20d.s |
| 1000m radius 10° turn | ~175m | A1t1000r10d.s |
| 1000m radius 10° turn + round tunnel | ~175m | A1t1000r10dRndTun.s |
| 1000m radius 10° turn + high tunnel | ~175m | A1t1000r10dTun.s |
| 1000m radius 5° turn | ~88m | A1t1000r5d.s |
| 1000m radius 5° turn + round tunnel | ~88m | A1t1000r5dRndTun.s |
| 1000m radius 5° turn + high tunnel | ~88m | A1t1000r5dTun.s |
| | | |
| 1500m radius 20° turn | ~524m | A1t1500r20d.s |
| 1500m radius 10° turn | ~262m | A1t1500r10d.s |
| 1500m radius 10° turn + round tunnel | ~262m | A1t1500r10dRndTun.s |
| 1500m radius 10° turn + high tunnel | ~262m | A1t1500r10dTun.s |
| 1500m radius 5° turn | ~131m | A1t1500r5d.s |
| 1500m radius 5° turn + round tunnel | ~131m | A1t1500r5dRndTun.s |
| 1500m radius 5° turn + high tunnel | ~131m | A1t1500r5dTun.s |
| | | |
| 2000m radius 20° turn | ~698m | A1t2000r20d.s |
| 2000m radius 10° turn | ~349m | A1t2000r10d.s |
| 2000m radius 5° turn | ~175m | A1t2000r5d.s |
| 2000m radius 5° turn + round tunnel | ~175m | A1t2000r5dRndTun.s |
| 2000m radius 5° turn + high tunnel | ~175m | A1t2000r5dTun.s |
| | | |
| | | |
| **Single Track - End of Switch Curves** | | |
| | | |
| ~172.7m radius 10° turn left | ~30m | A1tEndPnt10dLft.s |
| ~172.7m radius 10° turn left + round tunnel | ~30m | A1tEndPnt10dLftRndTun.s |
| ~172.7m radius 10° turn left + high tunnel | ~30m | A1tEndPnt10dLftTun.s |
| ~172.7m radius 10° turn right | ~30m | A1tEndPnt10dRgt.s |
| ~172.7m radius 10° turn right + round tunnel | ~30m | A1tEndPnt10dRgtRndTun.s |
| ~172.7m radius 10° turn right + high tunnel | ~30m | A1tEndPnt10dRgtTun.s |
| | | |
| ~869m radius 5° turn left | ~76m | A1tEndPnt5dLft.s |
| ~869m radius 5° turn left + round tunnel | ~76m | A1tEndPnt5dLftRndTun.s |
| ~869m radius 5° turn left + high tunnel | ~76m | A1tEndPnt5dLftTun.s |
| ~869m radius 5° turn right | ~76m | A1tEndPnt5dRgt.s |
| ~869m radius 5° turn right + round tunnel | ~76m | A1tEndPnt5dRgtRndTun.s |
| ~869m radius 5° turn right + high tunnel | ~76m | A1tEndPnt5dRgtTun.s |
| | | |
| **Single Track - Switches (Points)** | | |
| | | |
| ~67m radius 45° turn left yard automatic | ~53m curve 20m straight | A1t45dYardLft.s |
| ~67m radius 45° turn left yard manual | ~53m curve 20m straight | A1t45dYardLftMnl.s |

| | | |
|---|---|---|
| ~67m radius 45° turn right yard automatic | ~53m curve<br>20m straight | A1t45dYardRgt.s |
| | | |
| ~172.7m radius 10° turn left automatic | ~30m curve<br>40m straight | A1tPnt10dLft.s |
| ~172.7m radius 10° turn left manual | ~30m curve<br>40m straight | A1tPnt10dLftMnl.s |
| ~172.7m radius 10° turn right automatic | ~30m curve<br>40m straight | A1tPnt10dRgt.s |
| ~172.7m radius 10° turn right manual | ~30m curve<br>40m straight | A1tPnt10dRgtMnl.s |
| ~172.7m radius 10° turn wye automatic | ~30m | A1tYPnt10d.s |
| ~172.7m radius 10° turn wye manual | ~30m | A1tYPnt10dMnl.s |
| | | |
| ~870m radius 5° turn left automatic | ~75m curve<br>80m straight | A1tPnt5dLft.s |
| ~870m radius 5° turn left manual | ~75m curve<br>80m straight | A1tPnt5dLftMnl.s |
| ~870m radius 5° turn right automatic | ~75m curve<br>80m straight | A1tPnt5dRgt.s |
| | | |
| ~2870m radius 2.5° turn left automatic | ~125m curve<br>130m<br>straight | A1tPnt2_5dLft.s |
| ~2870m radius 2.5° turn left manual | ~125m curve<br>130m<br>straight | A1tPnt2_5dLftMnl.s |
| ~2870m radius 2.5° turn right automatic | ~125m curve<br>130m<br>straight | A1tPnt2_5dRgt.s |
| | | |
| | | |
| **Single Track - Crossovers** | | |
| | | |
| 5° crossover | 50m | A1tXOver5d.s |
| | | |
| 2.5° crossover | 100m | A1tXOver2_5d.s |
| | | |
| | | |
| **Double Track - Straight** | | |
| | | |
| 2 x 6m long | 6m | A2t6mStrt.s |
| | | |
| 2 x 10m long | 10m | A2t10mStrt.s |
| 2 x 10m long + high tunnel | 10m | A2t10mStrtTun.s |
| | | |
| 2 x ~43m long skew right | ~43m | A2tSkewRgt.s |
| | | |
| 2 x 50m long | 50m | A2t50mStrt.s |
| | | |
| 2 x 100m long | 100m | A2t100mStrt.s |
| 2 x 100m long + high tunnel | 100m | A2t100mStrtTun.s |
| | | |
| 2 x 250m long | 250m | A2t250mStrt.s |
| 2 x 250m long + high tunnel | 250m | A2t250mStrtTun.s |

| | | |
|---|---|---|
| 2 x 500m long | 500m | A2t500mStrt.s |
| 2 x 500m long + high tunnel | 500m | A2t500mStrtTun.s |
| | | |
| | | |
| **Double Track - Curves** | | |
| | | |
| 2 x 250m radius 20° turn | ~87m | A2t250r20d.s |
| 2 x 250m radius 20° turn + high tunnel | ~87m | A2t250r20dTun.s |
| 2 x 250m radius 10° turn | ~44m | A2t250r10d.s |
| 2 x 250m radius 10° turn + high tunnel | ~44m | A2t250r10dTun.s |
| 2 x 250m radius 5° turn | ~22m | A2t250r5d.s |
| | | |
| 2 x 500m radius 20° turn | ~175m | A2t500r20d.s |
| 2 x 500m radius 20° turn + high tunnel | ~175m | A2t500r20dTun.s |
| 2 x 500m radius 10° turn | ~88m | A2t500r10d.s |
| 2 x 500m radius 10° turn + high tunnel | ~88m | A2t500r10dTun.s |
| 2 x 500m radius 5° turn | ~44m | A2t500r5d.s |
| 2 x 500m radius 5° turn + high tunnel | ~44m | A2t500r5dTun.s |
| | | |
| 2 x 1000m radius 20° turn | ~349m | A2t1000r20d.s |
| 2 x 1000m radius 20° turn + high tunnel | ~349m | A2t1000r20dTun.s |
| 2 x 1000m radius 10° turn | ~175m | A2t1000r10d.s |
| 2 x 1000m radius 10° turn + high tunnel | ~175m | A2t1000r10dTun.s |
| 2 x 1000m radius 5° turn | ~88m | A2t1000r5d.s |
| 2 x 1000m radius 5° turn + high tunnel | ~88m | A2t1000r5dTun.s |
| | | |
| 2 x 1500m radius 20° turn | ~524m | A2t1500r20d.s |
| 2 x 1500m radius 20° turn + high tunnel | ~524m | A2t1500r20dTun.s |
| 2 x 1500m radius 10° turn | ~262m | A2t1500r10d.s |
| 2 x 1500m radius 10° turn + high tunnel | ~262m | A2t1500r10dTun.s |
| 2 x 1500m radius 5° turn | ~131m | A2t1500r5d.s |
| 2 x 1500m radius 5° turn + high tunnel | ~131m | A2t1500r5dTun.s |
| | | |
| 2 x 2000m radius 20° turn | ~698m | A2t2000r20d.s |
| 2 x 2000m radius 20° turn + high tunnel | ~698m | A2t2000r20dTun.s |
| 2 x 2000m radius 10° turn | ~349m | A2t2000r10d.s |
| 2 x 2000m radius 10° turn + high tunnel | ~349m | A2t2000r10dTun.s |
| 2 x 2000m radius 5° turn | ~175m | A2t2000r5d.s |
| 2 x 2000m radius 5° turn + high tunnel | ~175m | A2t2000r5dTun .s |
| | | |
| | | |
| **Triple Track - Straight** | | |
| | | |
| 3 x 10m long | 10m | A3t10mStrt.s |
| | | |
| 3 x 50m long | 50m | A3t50mStrt.s |
| | | |
| 3 x 100m long | 100m | A3t100mStrt.s |
| | | |
| 3 x 250m long | 250m | A3t250mStrt.s |
| | | |
| 3 x 500m long | 500m | A3t500mStrt.s |
| | | |
| | | |

| | | |
|---|---|---|
| **Triple Track - Curves** | | |
| | | |
| 3 x 500m radius 20° turn | ~175m | A3t500r20d.s |
| 3 x 500m radius 10° turn | ~88m | A3t500r10d.s |
| 3 x 500m radius 5° turn | ~44m | A3t500r5d.s |
| | | |
| 3 x 1000m radius 20° turn | ~349m | A3t1000r20d.s |
| 3 x 1000m radius 10° turn | ~175m | A3t1000r10d.s |
| 3 x 1000m radius 5° turn | ~88m | A3t1000r5d.s |
| | | |
| 3 x 1500m radius 20° turn | ~524m | A3t1500r20d.s |
| 3 x 1500m radius 10° turn | ~262m | A3t1500r10d.s |
| 3 x 1500m radius 5° turn | ~131m | A3t1500r5d.s |
| | | |
| 3 x 2000m radius 20° turn | ~698m | A3t2000r20d.s |
| 3 x 2000m radius 10° turn | ~349m | A3t2000r10d.s |
| 3 x 2000m radius 5° turn | ~175m | A3t2000r5d.s |
| | | |
| | | |
| **Quadruple Track - Straight** | | |
| | | |
| 4 x 10m long | 10m | A4t10mStrt.s |
| | | |
| 4 x 50m long | 50m | A4t50mStrt.s |
| | | |
| 4 x 100m long | 100m | A4t100mStrt.s |
| | | |
| 4 x 250m long | 250m | A4t250mStrt.s |
| | | |
| 4 x 500m long | 500m | A4t500mStrt.s |
| | | |
| | | |
| **Quadruple Track - Curves** | | |
| | | |
| 4 x 500m radius 20° turn | ~175m | A4t500r20d.s |
| 4 x 500m radius 10° turn | ~88m | A4t500r10d.s |
| 4 x 500m radius 5° turn | ~44m | A4t500r5d.s |
| | | |
| 4 x 1000m radius 20° turn | ~349m | A4t1000r20d.s |
| 4 x 1000m radius 10° turn | ~175m | A4t1000r10d.s |
| 4 x 1000m radius 5° turn | ~88m | A4t1000r5d.s |
| | | |
| 4 x 1500m radius 20° turn | ~524m | A4t1500r20d.s |
| 4 x 1500m radius 10° turn | ~262m | A4t1500r10d.s |
| 4 x 1500m radius 5° turn | ~131m | A4t1500r5d.s |
| | | |
| 4 x 2000m radius 20° turn | ~698m | A4t2000r20d.s |
| 4 x 2000m radius 10° turn | ~349m | A4t2000r10d.s |
| 4 x 2000m radius 5° turn | ~175m | A4t2000r5d.s |

## APPENDIX C. TABLE OF AVAILABLE ROAD SECTIONS

The following table describes all the road sections that are available in the MSTS Route Editor, grouped in categories. The lengths of curved sections are measured along the curved arcs and are approximate. At right are the road section names found in the Object selector.

(Note: A few more road sections, not tabulated below, are available in some default MSTS routes, namely CurveRoad in EUROPE1, and JP1CurveRoad2 and JP1StraightRoad in JAPAN1, and ten more in EUROPE2. In the Object selector, they show up without the ".s". These are non-functional road sections that do not automatically connect to other road sections, and to which the terrain can't be adjusted by pressing Y. They should instead be viewed and used as standard objects.)

| ROAD DESCRIPTION | LENGTH | NAME |
|---|---|---|
| | | |
| **Dirt Road - 2 lanes** | | |
| | | |
| 10m straight | 10m | DirtRoad2LStrt10m.s |
| 50m straight | 50m | DirtRoad2LStrt50m.s |
| 100m straight | 100m | DirtRoad2LStrt100m.s |
| 250m straight | 250m | DirtRoad2LStrt250m.s |
| | | |
| ~40m radius 45° turn | ~31m | DirtRoad2L45deg.s |
| ~40m radius 22.5° turn | ~16m | DirtRoad2L22deg.s |
| | | |
| T junction | 20m x 10m | DirtRoad2LaneT.s |
| + crossing | 20m x 20m | DirtRoad2LCross.s |
| | | |
| 2-to-4 lane T junction | 20m x 10m | DirtRoad4to2LnT.s |
| | | |
| | | |
| **Dirt Road - 4 lanes** | | |
| | | |
| 10m straight | 10m | DirtRoad4LStrt10m.s |
| 50m straight | 50m | DirtRoad4LStrt50m.s |
| 100m straight | 100m | DirtRoad4LStrt100m.s |
| 250m straight | 250m | DirtRoad4LStrt250m.s |
| | | |
| ~40m radius 45° turn | ~31m | DirtRoad4L45deg.s |
| ~40m radius 22.5° turn | ~16m | DirtRoad4L22deg.s |
| | | |
| T junction | 20m x 10m | DirtRoad4LaneT.s |
| + crossing | 20m x 20m | DirtRoad4LCross.s |
| | | |
| 4-to-2 lane T junction | 20m x 10m | DirtRoad4to2LnT.s |
| | | |
| **Dark asphalt road (no stripes) - 2 lanes** | | |
| | | |
| 10m straight | 10m | JP2Rd10mStrt.s |
| 50m straight | 50m | JP2Rd50mStrt.s |
| 100m straight | 100m | JP2Rd100mStrt.s |
| | | |

| | | |
|---|---|---|
| ~40m radius 45° turn | ~31m | JP2Rd2L45deg.s |
| ~40m radius 22.5° turn | ~16m | JP2Rd2L22deg.s |
| 250m radius 20° turn | ~87m | JP2Rd250r20d.s |
| 250m radius 10° turn | ~44m | JP2Rd250r10d.s |
| 250m radius 5° turn | ~22m | JP2Rd250r5d.s |
| 500m radius 20° turn | ~175m | JP2Rd500r20d.sd |
| 500m radius 10° turn | ~88m | JP2Rd500r10d.s |
| 500m radius 5° turn | ~44m | JP2Rd500r5d.s |
| 1000m radius 20° turn | ~349m | JP2Rd1000r20d.s |
| 1000m radius 10° turn | ~175m | JP2Rd1000r10d.s |
| 1000m radius 5° turn | ~88m | JP2Rd1000r5d.s |
| 1500m radius 20° turn | ~524m | JP2Rd1500r20d.s |
| 1500m radius 10° turn | ~262m | JP2Rd1500r10d.s |
| 1500m radius 5° turn | ~131m | JP2Rd1500r5d.s |
| | | |
| T junction | 20m x 10m | JP2Rd2LaneT.s |
| + crossing | 20m x 20m | JP2Rd2LCrs.s |
| | | |
| | | |
| **Light asphalt road (with stripes) - 2 lanes** | | |
| | | |
| 10m straight | 10m | Road2LStrt10m.s |
| 50m straight | 50m | Road2LStrt50m.s |
| 100m straight | 100m | Road2LStrt100m.s |
| 250m straight | 250m | Road2LStrt250m.s |
| | | |
| ~40m radius 45° turn | ~31m | Road2L45deg.s |
| ~40m radius 22.5° turn | ~16m | Road2L22deg.s |
| 250m radius 20° turn | ~87m | US1Rd2L250r20d.s |
| 250m radius 10° turn | ~44m | US1Rd2L250r10d.s |
| 250m radius 5° turn | ~22m | US1Rd2L250r5d.s |
| 500m radius 20° turn | ~175m | US1Rd2L500r20d.s |
| 500m radius 10° turn | ~88m | US1Rd2L500r10d.s |
| 500m radius 5° turn | ~44m | US1Rd2L500r5d.s |
| 1000m radius 20° turn | ~349m | US1Rd2L1000r20d.s |
| 1000m radius 10° turn | ~175m | US1Rd2L1000r10d.s |
| 1000m radius 5° turn | ~88m | US1Rd2L1000r5d.s |
| 1500m radius 20° turn | ~524m | US1Rd2L1500r20d.s |
| 1500m radius 10° turn | ~262m | US1Rd2L1500r10d.s |
| 1500m radius 5° turn | ~131m | US1Rd2L1500r5d.s |
| | | |
| T junction | 20m x 10m | Road2LaneT.s |
| + crossing | 20m x 20m | Road2LCross.s |
| | | |
| 2-to-4 lane T junction | 20m x 10m | Road4to2LnT.s |
| | | |
| | | |
| **Light asphalt road (with stripes) - 4 lanes** | | |
| | | |
| 10m straight | 10m | Road4LStrt10m.s |
| 50m straight | 50m | Road4LStrt50m.s |
| 100m straight | 100m | Road4LStrt100m.s |
| 250m straight | 250m | Road4LStrt250m.s |
| | | |

| | | |
|---|---|---|
| ~40m radius 45° turn | ~31m | Road4L45deg.s |
| ~40m radius 22.5° turn | ~16m | Road4L22deg.s |
| | | |
| 250m radius 20° turn | ~87m | US1Rd4L250r20d.s |
| 250m radius 10° turn | ~44m | US1Rd4L250r10d.s |
| 250m radius 5° turn | ~22m | US1Rd4L250r5d.s |
| 500m radius 20° turn | ~175m | US1Rd4L500r20d.s |
| 500m radius 10° turn | ~88m | US1Rd4L500r10d.s |
| 500m radius 5° turn | ~44m | US1Rd4L500r5d.s |
| 1000m radius 20° turn | ~349m | US1Rd4L1000r20d.s |
| 1000m radius 10° turn | ~175m | US1Rd4L1000r10d.s |
| 1000m radius 5° turn | ~88m | US1Rd4L1000r5d.s |
| 1500m radius 20° turn | ~524m | US1Rd4L1500r20d.s |
| 1500m radius 10° turn | ~262m | US1Rd4L1500r10d.s |
| 1500m radius 5° turn | ~131m | US1Rd4L1500r5d.s |
| | | |
| T junction | 20m x 10m | Road4LaneT.s |
| + crossing | 20m x 20m | Road4LCross.s |
| | | |
| 2-to-4 lane T junction | 20m x 10m | Road4to2LnT.s |

## APPENDIX D.  TABLE OF SLOPES AND GRADIENTS

The following table gives, for selected track slopes, the correspondence between slope angles and gradients, expressed in different forms.

The gradient (x : 1) is obtained from the angle (a in °) as x = tan (a * 3.141592654 / 180).  The gradient and rise values in the third and fourth columns are obtained by multiplying x in the 2nd column by 100 and 1000, respectively.

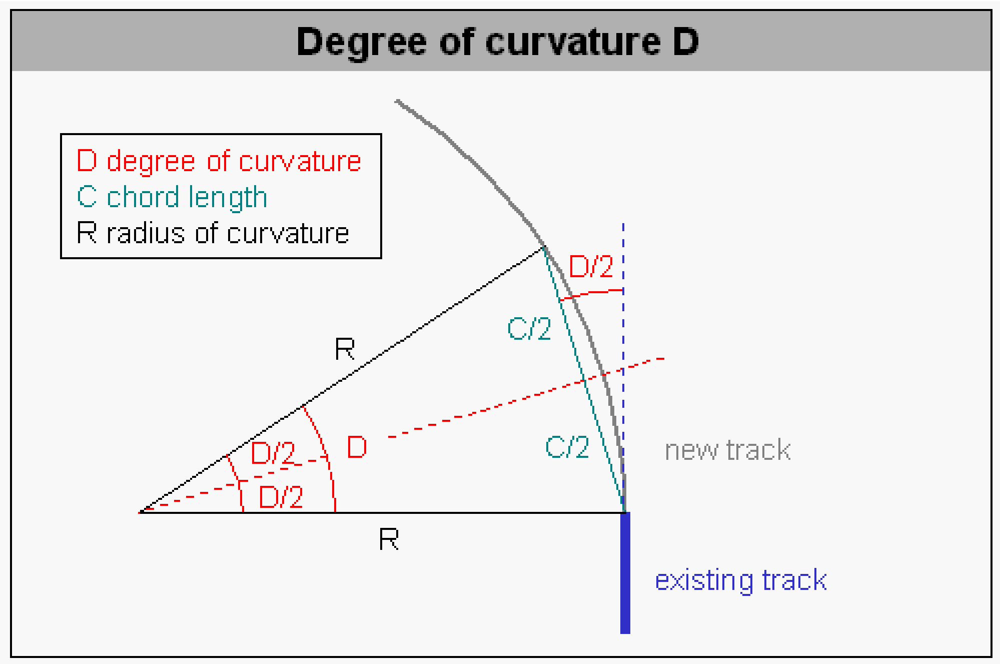
The slope angle (a in °) is obtained from the gradient (x : 1) as a = arctan (x) * 180 / 3.141592654.  To convert from the values in the third column to the slope angle, use a = arctan (x / 100) * 180 / 3.141592654.  To convert from the values in the fourth column to the slope angle, use a = arctan (x / 1000) * 180 / 3.141592654.

| slope angle (°) | gradient (x : 1) | grade (%) = rise (in m or ft) per 100 (m or ft) | rise (in m or ft) per 1000 (m or ft) |
|---|---|---|---|
| 0 | 0.0000 : 1 | 0.00 | 0.00 |
| 0.3 | 0.0052 : 1 | 0.52 | 5.24 |
| 0.6 | 0.0105 : 1 | 1.05 | 10.47 |
| 0.9 | 0.0157 : 1 | 1.57 | 15.71 |
| 1.2 | 0.0209 : 1 | 2.09 | 20.95 |
| 1.5 | 0.0262 : 1 | 2.62 | 26.19 |
| 1.8 | 0.0314 : 1 | 3.14 | 31.43 |
| 2.1 | 0.0367 : 1 | 3.67 | 36.67 |
| 2.4 | 0.0419 : 1 | 4.19 | 41.91 |
| 2.7 | 0.0472 : 1 | 4.72 | 47.16 |
| 3 | 0.0524 : 1 | 5.24 | 52.41 |
| 4 | 0.0699 : 1 | 6.99 | 69.93 |
| 5 | 0.0875 : 1 | 8.75 | 87.49 |
| 6 | 0.1051 : 1 | 10.51 | 105.10 |
| 7 | 0.1228 : 1 | 12.28 | 122.78 |
| 8 | 0.1405 : 1 | 14.05 | 140.54 |
| 9 | 0.1584 : 1 | 15.84 | 158.38 |
| 10 | 0.1763 : 1 | 17.63 | 176.33 |
| 12 | 0.2126 : 1 | 21.26 | 212.56 |
| 15 | 0.2679 : 1 | 26.79 | 267.95 |
| 18 | 0.3249 : 1 | 32.49 | 324.92 |
| 21 | 0.3839 : 1 | 38.39 | 383.86 |
| 24 | 0.4452 : 1 | 44.52 | 445.23 |
| 27 | 0.5095 : 1 | 50.95 | 509.53 |
| 30 | 0.5774 : 1 | 57.74 | 577.35 |
| 33 | 0.6494 : 1 | 64.94 | 649.41 |
| 36 | 0.7265 : 1 | 72.65 | 726.54 |
| 39 | 0.8098 : 1 | 80.98 | 809.78 |
| 42 | 0.9004 : 1 | 90.04 | 900.40 |
| 45 | 1.0000 : 1 | 100.00 | 1000.00 |

[NEW SINCE V1.106]

## APPENDIX E.  TABLE OF DEGREES OF CURVATURE

In MSTS, curves are defined by their radius of curvature (R in the next figure) and turn angle (such as D in the next figure).  In real-life practice, this is not convenient for laying track: especially with large radii, it is usually impractical to go to the center of curvature (a distance equal to the radius away from the actual track) and then sweep a rope of length equal to the radius to draw out the circle that the track should follow;  even with more modern laser equipment, it may be impractical to physically place it at the center of curvature.  Real-life engineering (for example for American railroads) in practice uses the degree of curvature instead of the radius of curvature.




**The degree of curvature D is the angle swept by a straight chord of predefined length C**, as shown in the figure above.  A given degree of curvature D and a given chord length C imply a certain radius of curvature R, through the relation

$$R = 0.5\ C / \sin(0.5\ D).$$

Conversely, a given C and a given R imply a degree of curvature D (which is the turn angle) given by

$$D = 2 \arcsin(0.5\ C / R)$$

(arcsin is often also written as $\sin^{-1}$, the inverse of the sine function).

Typically, in America a chord length of 100 ft is used, while in the metric system a value of 20 m is common.

So the two relations become, for a chord of 100 ft, and expressing R in feet:

$R = 50 / \sin(0.5\ D)$ and $D = 2 \arcsin(50 / R)$.

Likewise, using a 20 m chord, and expressing R in meters, the relations are

$R = 10 / \sin(0.5\ D)$ and $D = 2 \arcsin(10 / R)$.

We now relate this to the track sections available in RE.

If you use curving **dynamic track**, you need a **radius in meters** and a turn angle. If you have degrees of curvature given in English units (based on a **chord of 100 ft**), this radius in meters is given by the relation

**$R = 15.24 / \sin(0.5\ D)$ meter**.

If you have data given in the metric system (with a **chord of 20 m**), this radius in meters is given by

**$R = 10 / \sin(0.5\ D)$ meter**.

The turn angle to be used in the dynamic track is any angle you wish (it is not necessarily the angle D defined by the chord).

**The table below lists radii (in meters) corresponding to various degrees of curvature, both for a 100 ft chord and a 20 m chord.**

If you do not want to use dynamic track sections, but **regular track sections** with preset radii and turn angles, you can use the table below to find the radius (in meters) corresponding to the given degree of curvature. Then you can find (in Appendix B) track sections that have a radius (in meters) close to this value. Again, the turn angle is an independent issue: you can use as many regular track sections as needed to add up to the desired turn angle; however, you will still need a dynamic track section to obtain a turn angle that is not a multiple of 5°.

| degree of curvature (°) | radius (m) for 100 ft chord | radius (m) for 20 m chord |
|---|---|---|
| 0.2 | 8731.88 | 5729.58 |
| 0.4 | 4365.95 | 2864.79 |
| 0.6 | 2910.64 | 1909.87 |
| 0.8 | 2182.99 | 1432.41 |
| 1.0 | 1746.40 | 1145.93 |
| 1.5 | 1164.28 | 763.97 |
| 2.0 | 873.23 | 572.99 |
| 2.5 | 698.61 | 458.40 |
| 3.0 | 582.19 | 382.02 |

| | | |
|---|---|---|
| 3.5 | 499.04 | 327.46 |
| 4.0 | 436.68 | 286.54 |
| 4.5 | 388.18 | 254.71 |
| 5.0 | 349.39 | 229.26 |
| 5.5 | 317.64 | 208.43 |
| 6.0 | 291.20 | 191.07 |
| 6.5 | 268.82 | 176.39 |
| 7.0 | 249.64 | 163.80 |
| 7.5 | 233.02 | 152.90 |
| 8.0 | 218.47 | 143.36 |
| 8.5 | 205.64 | 134.94 |
| 9.0 | 194.24 | 127.45 |
| 9.5 | 184.04 | 120.76 |
| 10.0 | 174.86 | 114.74 |
| 11.0 | 159.01 | 104.33 |
| 12.0 | 145.80 | 95.67 |
| 13.0 | 134.63 | 88.34 |
| 14.0 | 125.05 | 82.06 |
| 15.0 | 116.76 | 76.61 |
| 16.0 | 109.50 | 71.85 |
| 17.0 | 103.11 | 67.65 |
| 18.0 | 97.42 | 63.92 |
| 19.0 | 92.34 | 60.59 |
| 20.0 | 87.76 | 57.59 |
| 25.0 | 70.41 | 46.20 |
| 30.0 | 58.88 | 38.64 |
| 35.0 | 50.68 | 33.26 |
| 40.0 | 44.56 | 29.24 |
| 45.0 | 39.82 | 26.13 |
| 50.0 | 36.06 | 23.66 |
| 55.0 | 33.00 | 21.66 |
| 60.0 | 30.48 | 20.00 |
| 65.0 | 28.36 | 18.61 |
| 70.0 | 26.57 | 17.43 |
| 75.0 | 25.03 | 16.43 |
| 80.0 | 23.71 | 15.56 |
| 85.0 | 22.56 | 14.80 |
| 90.0 | 21.55 | 14.14 |

[NEW SINCE V1.106]
## Appendix F. STRUCTURE OF A ROUTE FOLDER

This Appendix explains how the folder of a route is organized.

### The subfolders of a route have the following purpose:

**ACTIVITIES**: This subfolder contains *.act files that define all Activities associated with the route. Each *.act file refers to paths, services and traffic in other subfolders.

**ENVFILES**: The *.env files define the weather and water appearance. It uses textures in subsubfolder TEXTURES.

**LO_TILES**: This subfolder is similar to the TILES subfolder, but for low-resolution distant scenery (this subfolder exists in only a few routes).

**PATHS**: The *.pat files define the specific track routings (paths) used in the various activities that exist in subfolder ACTIVITIES.

**SERVICES**: The *.srv files define the specific combinations of consists (trains) and paths that are used in the activities of this route.

**SHAPES**: The *.s and *.sd files define the geometry of the different visual objects that can be placed in this route. The shape files *.s also refer to textures that are stored in subfolder TEXTURES. The extended shape files *.sd in addition specify which seasonal or other textures (night, snow) exist for each object. Not included in this subfolder are certain global objects (such as tracks and roads).

**SOUND**: The *.sms and *.wav files define the various environmental sounds that can be placed in this route.

**TD**: The *.td and *.dat files define the geography of the tiles of this route.

**TERRTEX**: These *.ace files are terrain textures that can color the ground in each patch of a tile. The SNOW subsubfolder includes snow-covered versions.

**TEXTURES**: These *.ace files contain the coloring of objects defined in subfolder SHAPES. Several subsubfolders (AUTUMN, AUTUMNSNOW, NIGHT, SNOW, SPRING, SPRINGSNOW, WINTER, WINTERSNOW) contain variants for different seasons and lighting or weather conditions.

**TILES**: The *.t and *y.raw define the terrain shapes (altitudes) and the corresponding terrain textures. The *f.raw files contain information about platform and siding names. The *e.raw and *n.raw files, if not already present, are automatically generated from the other files in this subfolder, to accelerate the rendering of terrain shapes and lighting (so they need not be provided).

**TRAFFIC**: The *.trf files define the services and timing of artificial intelligence trains.

**TUTOR**: This subfolder may contain a tutorial for teaching how to use this route or drive a locomotive. By default, only the Tutorial Route includes such a tutorial, but each add-on route is provided with an empty folder for a tutorial (no instructions are available to generate such a tutorial).

**WORLD**: The *.w files specify the positions and orientations of all track, road and other objects on each tile. The *.ws files likewise specify the positions of all sounds on each tile.

**The files in the root folder of a route have the following purpose** (note that the <routename> need not be the same for each of the files listed):

**<routename>.ace**: A logo image that is displayed before loading this route in MSTS (empty by default in a new route).

**<routename>.mkr**: This route's marker file (not required).

**<routename>.rdb**: This route's road database. It specifies how road sections are connected (absent if there are no roads).

**<routename>.ref**: A file that lists and organizes objects for the Placement tool of RE.

**<routename>.rit**: A file that defines the positions of road objects: car spawners, level crossings.

**<routename>.tdb**: This route's track database. It specifies how track sections are connected.

**<routename>.tit**: A file that defines the positions of track objects: platforms, sidings, signals, speedposts (speed limit signs and mileposts), level crossings, hazards, track sound regions.

**<routename>.trk**: A file that specifies the basic parameters of this route, including: route name, route description, associated graphic files, electrification status, speed limits, environmental files used, units displayed (miles vs. kilometers), default sounds.

**carspawn.dat**: Specifies the vehicles that drive on the roads equipped with a car spawner.

**deer.haz**: Controls the animated deer (Deer_on_track).

**details.ace**: A "Map" screen that can be displayed before loading a route in MSTS (not required).

**forests.dat**: Defines and controls different types of forests that can be placed on this route.

**gantry.dat**:  Controls the automatic placement of gantries, if this route is electrified (only required if this route is electrified).

**load.ace** (may have a different name):  A load screen that is displayed while this route is loaded in MSTS (not required).

**sigcfg.dat**:  Defines the signal shapes and functions for this route.

**sigscr.dat**:  Defines the logic of the signals for this route.

**speedpost.dat**:  Defines the speedpost shapes (for speed limit signs and mileposts).

**spotter.haz**:  Controls the people that only show in activities (Person).

**ssource.dat**:  Defines the various environmental sounds available in this route.

**telepole.dat**:  Controls the placement of telephone poles.

**tsection.dat**:  Lists dynamic tracks in this route.

**ttype.dat**:  Lists the different track sounds available in this route.

[NEW SINCE V1.106]

## Appendix G. SUGGESTED CHECKLIST FOR ROUTE BUILDING

This checklist shows a suggested order of operations: it minimizes risks by taking into account that some operations are best done when others have been completed.

The best order can vary with the type of the route and with what is included or what is omitted. It is possible to complete the building of one part of a route before starting another part of the same route.

| action | comments |
|---|---|
| plan route | |
| generate tiles (in RGE) | if changing an existing route, add tiles as needed |
| generate DEM terrain | if desired |
| redefine tile altitude limits | if needed and if no DEM data used |
| if changing existing route: delete all track objects attached to tracks that will be changed or removed | |
| place guides for laying tracks | use markers, trees or water |
| lay all tracks | adjust terrain roughly along tracks |
| place railway bridges | do this before making valleys below tracks; set error bias to 0 in each patch with a bridge |
| place underground structures | do this before raising terrain above track level |
| shape terrain everywhere | if no DEM data used |
| add water | |
| make tunnel entrances | |
| fine-tune terrain along tracks | set error bias to 0 in each patch |
| add mileposts | use a few signals or platform names to measure distances |
| add siding and platform names | also place siding names at any points of interest |
| add yard definitions | |
| add all signals | |
| add speed limits | |
| add roads, level crossings and road bridges | |
| add road traffic | |
| change terrain textures | |
| add station platforms | |
| add people, animals (interactive) | place static people and animals later |
| add pickup objects | |
| add sounds (regions, environmental) | |
| electrify route | |
| add gantries | use autoplacement tool with manual corrections and additions (especially in yards) |
| add forests | |

| | |
|---|---|
| add all other objects, default and add-on:<br>buildings<br>fences/walls<br>rocks/piles/junk<br>power boxes<br>platform objects: benches, etc.<br>terrain patches (including whitewater)<br>static vehicles and trains<br>static people and animals<br>lamp posts<br>telephone poles<br>etc. | |
| make activities and consists (in AE) | if needed, first change switches between manual and automatic and/or adjust signals (in RE) |
| make documentation | optionally make logo, loading and detail screens |
| package route for distribution | test installation instructions |

[NEW SINCE V1.106]

## Appendix H. CHANGING AND MAKING TEXTURE GRAPHICS

This Appendix gives a brief introduction into changing and making the graphics that define textures in MSTS. The purpose is to enable you to adjust existing textures to your liking, or to create new ones for inclusion in your route.

Such textures can be used in various ways in MSTS: as terrain textures, as transfers, as object textures, as water textures, as sky textures, as opening screens for your route, etc.

All textures used as graphics in MSTS exist in the ACE format. This Appendix focuses on how textures are organized within ACE files, and how you can produce such ACE files. It does not deal with the purely graphical part of drawing, scanning, or modifying the graphics themselves. For that, you need to know or learn how to use any of the many graphics programs available. In many cases, Microsoft Paint and Photo Editor are sufficient to handle the purely graphical tasks.

### WHAT ARE ACE FILES?

The ACE format is not directly readable by most graphics software, including Microsoft Paint and Photo Editor. One reason is that ACE files in fact can contain two images: the graphic itself, and an "alpha channel" that defines the degree of transparency of each pixel in the graphic. You can find examples of such graphic/transparency pairs in section 3.15.3d.

### WHAT ARE ALPHA CHANNELS?

An alpha channel is additional information that defines the level of transparency to be applied to a graphic. Transparency is given on a gray scale: transparency is greatest when the alpha channel is darkest - black gives full transparency, and white is opaque.

You can think of the alpha channel image as being superimposed on top of the graphic that it belongs to: where the transparency is white (opaque), the viewer will see only this graphic, but where the transparency is black (fully transparent), the viewer will see right through this graphic to whatever is in the background. For gray tones in between white and black, the viewer will see a mixture of the graphic and the background that lies behind it: so, if the transparency is 80% black (dark gray), what the viewer sees is 80% background and 20% graphic.

### WHAT SOFTWARE CAN HANDLE ACE FILES?

MSTS comes with two utilities called MakeACE and MakeACEWin (which should be in its UTILS folder; if not, get it from the MSTS CDs): they allow producing ACE files from existing files in Bitmap or Targa formats.

The MakeACEWin utility (which runs under Windows) has certain limitations that reduce its usefulness: it cannot handle files of 1024x1024 pixels, and it cannot crop (cut off edges from) an image.

**The DOS-based MakeACE is particularly convenient for non-square images: see section 6.2 for details.**

However, in my experience, neither MakeACE nor MakeACEWin is convenient for handling alpha channels (transparency).

**You get the most flexibility with the free Windows-based software TGATool2** (available at http://fly.to/mwgfx), which I highly recommend. It actually uses MakeACE.

You need to pay special attention to a few aspects of the installation of TGATool2. First, TGATool2 relies on external graphics software to make the graphical modification or creations. You can set up TGATool2 to work together with your choice of graphics software; Microsoft Paint is the default.

NOTE: I find it more convenient to use Photo Editor instead of Paint as the graphics program. Even though it has limited graphical capabilities, Photo Editor is very convenient for viewing and handling several graphics and alpha channel images at the same time; for instance, Photo Editor allows you to copy a new BMP image over the old one and thereby replace the old one quickly (the new and old image sizes must be equal); Photo Editor also offers some graphics options that Paint does not have, such as color balancing and smoothing (softening). So I let TGATool2 send its images to Photo Editor, and I transfer those to Paint for further treatment. Of course, more sophisticated software can do all this within the same program.

Also, TGATool2 can compress files that it writes in ACE format: this is strongly recommended, especially for routes that will be distributed to other users. When you run TGATool2, you will see to the right a few options.

- I strongly recommend that you use the "Compress Ace Files" option;

- you should also use the "Use DXT Compression" option for textures WITHOUT an alpha channel (it prevents efficient compression for textures with an alpha channel); note that DXT Compression only works if you have Patch #1 - see next;

- in the drop-down list, look for "Patch #1 MakeAce" and select it so it becomes visible in that box; the "Patch #1" refers to the first patch of MSTS published by Microsoft on its Train-Sim web site; if you have not installed that patch, now is the time to do it.

**HOW DOES ONE MODIFY AN ACE FILE WITH TGATool2?**

When you open an ACE file in TGATool2, you will see its graphic at top left in the main window. It will often be square. If you see an image at top right under the words "Alpha Channel", the file that you opened contains an alpha channel that defines, pixel by pixel, the transparency of the graphic shown at top left.

**If you want to modify the graphic, follow these steps:**

- back up the original ACE file before you modify it;

- load the ACE file (select file type "ACE Image");

- open menu item Image and click on Send Main Image to Editor:  that will open your preselected graphics software, which will display that image in standard bitmap format;

- you can now edit that image in your graphics software;

- save it, without changing its file name (tgatool2.bmp);

- return to TGATool2;

- open menu item Image again, and click on the upper Reload After Edit:  the modified graphic should appear at top left, overwriting the old one;
- save the result as described below.

**You may also modify the alpha channel image**.  Do this **as follows**, similarly to the previous case:

- in TGATool2, open menu item Image and click on Send Alpha Channel to Editor:  that will open your preselected graphics software, which will display that alpha channel in standard bitmap format;

- you can now edit that alpha channel in your graphics software;

- save it, without changing its file name (tgatrns2.bmp);

- return to TGATool2;

- open menu item Image again, and click on the lower Reload After Edit:  the modified alpha channel should appear at top left, overwriting the old one;
- save the result as described below.

**If your file did not include an alpha channel, you can create one:**

- in TGATool2, click on Create Alpha Channel at right:  this will open a white alpha channel, which you can then modify as discussed above.

**If you want to create a new alpha channel, based on your image, you can create a template for one:**

- in TGATool2, click on Create Alpha Template at right:  this will create an alpha channel that is a gray-scale version of your image;  it can be used as a guide to produce a transparency that closely matches the details of your image:  it can be modified as discussed above.

**Now you can compress and save the resulting image and alpha channel as an ACE file** (the format required by MSTS), **as follows**:

- in TGATool2, open menu item Export and click on "Ace (no Alpha)" or "Ace (Alpha)", depending on whether you have an alpha channel or not (if the alpha channel has a gray level that makes it indistinguishable from the TGATool2's gray background, you may not be sure whether there is an alpha channel;  in that case, look at the title bar: if it says "+Trans", you have an alpha channel).

## HOW DOES ONE CREATE A NEW ACE FILE?

Before you create a new ACE file, you need to consider its size:  compression with TGATool2 only works properly for so-called "power of 2" sizes (32x32, 64x64, 128x128, 256x256, 512x512 or 1024x1024 pixels).  Exactly which sizes can be handled depends on a number of factors:  for details check in TGATool2's Help and in the file "MakeACE Release Notes.txt" that comes with the MSTS utilities (in folder UTILS).  For other than "power of 2" sizes, you should use MakeACE rather than TGATool2, as described in section 6.2.

**To create a new ACE file with TGATool2, do the following:**

- make its graphic with your graphics software, and save it in bitmap format with proper "power of 2" square size;

- read that graphic into TGATool2 as a BMP Image;

- if desired, create and modify an alpha channel, as described earlier;

- save to an ACE file, as described above.

CAUTION:  Occasionally, you will find strange effects, such as unexpected edges around the resulting texture.  In my experience, this seems to be due to the use of non-black or non-white pixels along the edges:  making the edge pixels strictly black or white tends to solve those problems.  The newer versions of TGATool and MakeACE are less likely to cause such problems.

[NEW SINCE V1.106]

## Appendix I. COLOR DEFINITION: HEXADECIMAL CODE

**In a few places, MSTS allows you to define colors using a "hexadecimal code" of the type "ff808080".** This happens, for example, in the *.env files found in the ENVFILES folder of a route.

**The hexadecimal color code defines a color by specifying its degree of transparency, and the amount of red, green and blue components. The code takes the form**

```
colour ( ttrrggbb )
```

**where tt = transparency, rr = red intensity, gg = green intensity, and bb = blue intensity.**

As far as I know, MSTS and RE do not use the transparency part "tt" of the color code, and it should always be kept as "ff".

**Each character t, r, g, or b can take any of the 16 values**

> **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f**

where 0 is lowest and f is highest.

> [Let me explain here how the hexadecimal code works. Hexadecimal means 16, just like decimal means 10: the hexadecimal system extends the 10-digit system that we normally use to a 16-digit system that is better adapted to computers because 16 is a power of 2: 16 is 2 to the power 4, since 2 x 2 x 2 x 2 = 16.
>
> While the decimal system uses 10 digits (0, 1, 2, 3, 4, 5, 6, 7, 8, and 9), the hexadecimal system uses 16 digits. It thus needs 6 more digits than the 10 that we use in the decimal system: those 6 additional digits are simply called a, b, c, d, e, and f. So counting from 0 upwards in the hexadecimal system looks like this:
>
> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f
>
> What happens when we have reached "f"?
>
> In the decimal system we count beyond 9 by switching to 10 and again increasing the right-most digit from 0 to 9 (10, 11, 12, …, 19), before switching to 20, etc. We do something very similar in the hexadecimal system. We first count until we exhaust the 16 characters available:
>
> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f

Then we switch to "10", and start increasing the right-most digit as far as it can go. So after "f", we count:

10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 1a, 1b, 1c, 1d, 1e, 1f

Next we switch to 20, and continue counting by increasing the right-most digit:

20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 2a, 2b, 2c, 2d, 2e, 2f

Then we switch to "30". Next we switch to "40", then to "50", etc.

After we reach "90" and count up to "9f", we switch not to "100", but to "a0", because "a" is the next available digit after "9". Thus we continue counting:

a0, a1, a2, a3, a4, a5, a6, a7, a8, a9, aa, ab, ac, ad, ae, af

followed by a switch to "b0":

b0, b1, b2, b3, b4, b5, b6, b7, b8, b9, ba, bb, bc, bd, be, bf

We switch to "c0", then "d0", "e0" and finally "f0", and count:

f0, f1, f2, f3, f4, f5, f6, f7, f8, f9, fa, fb, fc, fd, fe, ff

Only at this point would we switch to "100", because we have now exhausted the available digits. However, we don't need to count further.

If we had been counting in the decimal system instead of the hexadecimal system, we would have reached 255 by now: "ff" in the hexadecimal system is the same as 255 in the decimal system. Including the initial 0, we have thus defined 256 numbers (256 = 16 x 16).

This "ff" is as far as we need to count to define colors: we can use these 256 hexadecimal numbers to define 256 different "shades" or "intensities" of a color, by using two hexadecimal digits: the shade "ff" is the strongest, most intense shade of a color; the shade "0", usually written as "00", is the weakest shade of a color. So, if we take red, the shade "ff" will give us the most intense version of red (full red), while the shade "00" will give us the weakest version (no red).

As a result, we can define 256 intensities of red, 256 intensities of green and 256 intensities of blue (as well as 256 intensities of transparency). By combining red, green and blue, we can define a very wide variety of colors: counting the different combinations of 256 reds, 256 greens, and 256 blues, we get 256 x 256 x 256 = 16,777,216 different colors (nearly 17 million).

That is exactly how the hexadecimal color code works. A color code "ttrrggbb" defines a color that has a transparency level "tt", a red intensity "rr", a green intensity "gg", and a blue intensity "bb". See the examples below.

As far as I am aware, the transparency "tt" is actually not used in MSTS. So we can basically forget it: it will always have the value "ff".]

Let us consider a few examples of colors. First, suppose we choose

`colour ( ff000000 )` [**black**]

This means that tt = ff (ignored in MSTS), rr = 00 (no red), gg = 00 (no green), and bb = 00 (no blue): this color has zero intensities of red, green and blue, meaning it is black. At the other extreme, we can choose

`colour ( ffffffff )` [**white**]

This means that tt = ff (ignored in MSTS), rr = ff (full red), gg = ff (full green), and bb = ff (full blue). This color has full intensities of pure red, green and blue: they combine to produce pure white. Obviously, black and white are important colors.

We can also define different shades of gray: gray is an equal mix of red, green and blue. So

`colour ( ff202020 )` [**dark gray**]

contains a bit of red, a bit of green, and a bit of blue ("20" is much closer to "00" than to "ff", so it represents a weak intensity), giving a weak or dark gray. To get a light gray, you need more intensity of each color, for instance:

`colour ( ffb3b3b3 )` [**light gray**]

NOTE: **You produce major changes to a pure color intensity by changing the first of its two hexadecimal digits**: a change by 1 gives a relative intensity change of about 6% (1 / 16). **You can fine-tune a pure color intensity by changing the second of its two hexadecimal digits**: a change by 1 gives a relative intensity change of about 0.4% (1 / 256). So the contrast between "b3" and "b0", for example, is about 1% (3 / 256).

We can define pure red, green and blue colors in the following way. If we choose

`colour ( ffff0000 )` [**intense red**]

we have rr = ff (full red), gg = 00 (no green), and bb = 00 (no blue): this color is therefore pure intense red.

If we define

`colour ( ff800000 )` [**half-intense red**]

we have the same except for rr = 80 (about half-intense red): this gives half-intense pure red ("80" is in the middle of the range from "00" to "ff", so gives an intensity in the middle of the range).

Similarly, `colour ( ff00ff00 )` gives **pure intense green**, and `colour ( ff0000ff )` gives **pure intense blue**.

We can combine pairs of pure colors. Thus `colour ( ffff00ff )` includes both full intense red and full intense blue, giving **intense violet**. And `colour ( ff00ffff )` combines full intense green and full intense blue, giving intense **green-blue**. Also, `colour ( ffffff00 )` combines full intense red and full intense green, giving intense **yellow**.

For less intensity, just reduce the individual color intensities proportionally: to get half-intense yellow (namely orange), use something like `colour ( ff808000 )`.

It is often difficult to imagine what color corresponds to a given color code: what does `colour ( ff5f9ea0 )` look like? And it is also often difficult to figure out what color code is needed to create a particular color: what is the code for "cadet blue"? The answer is that `colour ( ff5f9ea0 )` is "cadet blue"! But how do I know this?

To help you with this, **you can find samples of colors with their color codes on the internet**. For instance, http://yath.mine.nu/hexbycolor.html shows a wide variety of colors with their color codes. At http://www.two4u.com/cgi-bin/color/compose/, you can compose your own color and see it as the background of the web page (just use its "BGCOLOR" bar and the "gray scale" square to the right): it will produce the hexadecimal code in the left-most box marked "H".

Note that the transparency part ("tt") is usually left out on web sites, so you only read about the "color" part "rrggbb" (the reason is that the part "rrggbb" is used to define colors for web pages in the html language). That is not a problem: just add "ff" in front of the "rrggbb" to form the "ttrrggbb" expected by MSTS.

[NEW SINCE V1.106]

## Appendix J. SCREENSHOTS AND CONTROLLING THE CAMERAS IN MSTS

**This Appendix describes how to take screenshots of your route, and some options for controlling the view camera in MSTS.**

Screenshots are useful for your own records, but especially to show aspects of your route to others. You can use them in your route's documentation, to advertise your route on the web, and also to make the screens that show up when you select and load your route in MSTS.

When taking screenshots, you need control over the "cameras" through which everything is seen in RE and MSTS. One of the more interesting changes is the field of view: it gives the effect of a "fish-eye lens" or a "zoom lens". Another aspect is the freedom of rotation and motion that the camera has: you can increase that freedom to give unusual angles of view.

### TAKING SCREENSHOTS

**In the Route Editor, you can take a screenshot by pressing Alt-PrintScreen: that captures the view window of RE** (including its border), but not the other small windows surrounding that view window. **If you press only PrintScreen, you capture the entire screen of your monitor**, including the RE view window, the small RE windows, the desktop behind that, and the Windows control bar at the bottom.

Alt-PrintScreen and PrintScreen put the captured image into the computer's temporary memory. **You must open a graphics program and paste the image into it. Then you can save the image in a convenient format** (see below). Or you may first modify the image as you wish, for instance by removing the edges, adding comments or changing the colors, etc. This method is slow because, after each screen capture, you have to open the graphics program, paste the image into it and save it to a file. (Also, the views in RE are not as good as those in MSTS.)

**In MSTS**, you can do the same thing as described above for the RE. However, there is a faster and more convenient method: if you **press PrintScreen, then MSTS automatically saves the full-screen view into a graphics file** in its main folder. The first time you do this during a session in MSTS, the file is called scrgrb0.pcx; the second time you press PrintScreen, the image is saved in scrgrb1.pcx; etc.

CAUTION: If you exit from MSTS and start it up again, it will save images again to scrgrb0.pcx, etc., overwriting any old versions. Therefore, after exiting MSTS, **you should rename those scrgrb*.pcx files that you want to keep**.

CAUTION: If you make many screenshots, you can saturate your computer's memory and cause it to crash.

A few words about **graphic file formats**. The most common graphic format is **Bitmap** (BMP, file names *.bmp); it stores every pixel of the image in a universally understood form.

However, this **requires very much space**, and should not be used for uploading to a web site, or for transmission to others (unless compressed in a zip file). The **PCX** format (file names *.pcx) used by MSTS **is similar, but a bit less widely used**: it also takes a lot of space, and is less easily viewed by others.

**The preferred graphic formats for storage, distribution and web sites are GIF and JPG** (file names *.gif and *.jpg). Both are widely used and viewable. Both use **very efficient compression** methods to minimize the file size. However, the compression methods are very different, so that in some cases a GIF file is smaller, while in other cases a JPG file is smaller: it all depends on the kind of picture. GIF is "non-lossy", meaning that it faithfully includes all details (so that you can completely regenerate the original image from a GIF version). JPG is "lossy", meaning that it simplifies the image somewhat for efficient compression. What is lost in a JPG image? Mainly sharp lines and boundaries: for example, in a JPG image you will see blurred lines and edges (especially if there are letters in the picture), as well as strange ghost lines near sharp edges. **So, you should use the JPG format for images that have slowly varying colors without sharp lines, boundaries and lettering. The GIF format is more efficient for pictures with lines, lettering, diagrams, and uniform colors.**

MSTS also uses another graphic format for textures: ACE. You would not use this to store screenshots. This format is not much used, so most graphics software cannot handle it. See Appendix H for a discussion of this special format.

## CONTROLLING THE CAMERAS

NOTE: **The camera changes described here do not affect the camera view in RE!** That view can apparently not be changed.

**The MSTS camera settings are found in the file camcfg.dat in the MSTS GLOBAL folder**. You will have to edit this file to change the camera views. The changes will apply to all routes.

CAUTION: Before editing camcfg.dat, make and save a backup copy.

CAUTION: Edit MSTS files only with a Unicode-capable word processor, such as WordPad (excepting very old versions) and Word.

NOTE: A file of type *.dat opens less easily than other files in a word processor (because you may not be able to associate the file type *.dat with a program). To open it in a word processor, right-click on the file, and select "Open with…", if available. Otherwise, start WordPad and then open camcfg.dat from inside WordPad.

NOTE: The first part of the default camcfg.dat contains comments that are largely inappropriate (they may be plans for the future).

The file camcfg.dat contains a number of camera definitions: they correspond to the camera selections that you get in MSTS when you press 1, Shift-1, 2, 3, …, 7.

Look at **the first camera definition: it gives the cab view**. Its definition is like this:

```
camera ( CamTypeCab
      CamType ( CamTypeCab CamControlViewSwitch )

      CameraOffset ( -0.51231 4.28496 10.0387 )

      Fov ( 60 )
      ZClip ( 0.5 )
      WagonNum ( 0 )
      Description (Cab_Cam)
)
```

**The line with Fov ( 60 ) is probably the most interesting to you: it gives the "fish-eye" or "zoom" lens effect, as an angle covering the "field of view". The angle of 60 degrees is the angle between the left and right edges of the camera view. To get a fish-eye lens effect, increase it, for example to 90 degrees (you can choose a larger value, as long as you stay below 180). If you want a zoom effect, decrease this angle, for example to 40 degrees.**

**You can make the same or different view-angle changes for each camera.** I like to have a narrower field of view for the front-tracking camera ("2"), and a wider field of view for the rear-tracking camera ("3").

The `CamTypeCab` declares this camera to give a cab view, while `CamControlViewSwitch` defines how it may move and rotate. The line with `CameraOffset` places the camera relative to a "wagon", namely `WagonNum ( 0 )`: this is the front "wagon", normally the (front) locomotive; `WagonNum ( -1 )` represents the last wagon. You can raise the camera by increasing the middle of the 3 numbers in `CameraOffset`. The line with `ZClip` probably cuts off objects when they come too close to the camera.

Other cameras are defined in a similar way, with minor variations. For example, they may have a line such as `Direction ( 0 0 0 )`: it gives the initial camera direction (in the passenger view). And `RotationLimit ( 60 160 0 )` gives the range of rotations that the camera is allowed to make. You will need to experiment with these numbers if you want to change them.

**There are two ways of increasing the freedom of motion of the different cameras**: **they are detailed in the following table**, which lists what I know. First, while running in MSTS you can press Ctrl-Shift-9 to allow more motion of certain cameras. Second, you can increase that further for the trackside camera ("4") and the yard camera ("7") by changing the camera control method in the camcfg.dat file (before starting MSTS): for example, for the trackside camera you would replace the line `CamType ( CamTypeSpotter CamControlRotate )` by the line `CamType ( CamTypeSpotter CamControlFull )`. You would make the same change to the yard camera definition: use `CamControlFull` instead of `CamControlYard`.

| MSTS key press | View | Default motions | Ctrl-Shift-9 adds to default | CamControlFull adds to default | Ctrl-Shift-9 adds to CamControlFull |
|---|---|---|---|---|---|
| 1 | cab | left & right views | none | none | none |
| Shift-1 | head out | horizontal rotation | vertical rotation | none | none |
| 2 | front tracking | height, orientation, distance | hor. & vert. rotation, hor. & vert. shift | none | none |
| 3 | rear tracking | height, orientation, distance | hor. & vert. rotation, hor. & vert. shift | none | none |
| 4 | track side ("Spotter Cam") | none | hor. & vert. rotation | hor. & vert. rotation, height, distance | hor. & vert. shift (but freezes rotations) |
| 5 | passenger | horizontal rotation | vertical rotation | none | none |
| 6 | coupling | height | none | none | none |
| 7 | yard | x, y, z within yard | none | hor. & vert. rotation, can exit yard | none |
| none | derail | vertical rotation | none | none | none |

Notes about the table above:

- a horizontal rotation turns the camera in the horizontal plane, from left to right or right to left;

- a vertical rotation turns the camera in the vertical plane, from up to down or down to up;

- a horizontal shift is a non-rotating motion of the camera in the horizontal plane: sideways, forward or backward;

- a vertical shift is a non-rotating motion of the camera in the vertical plane: up or down;

- after pressing Ctrl-Shift-9, you can rotate the camera by dragging the mouse while keeping the right mouse button pressed.

**The increased freedom of the yard camera is particularly valuable.** If your train is in a station with a yard definition, and if you press "7", you get the normal straight-down view onto the train. With the extra freedom given by CamControlFull, you not only can turn the camera to look in any direction, but you can also **move the camera anywhere along the route, independently of where the train is!** Also you can drive the train independently of where the camera is, so you can watch one train while driving another, or you can watch distant signals as your train moves. With a fish-eye-lens view, you can see wide portions of your route from a high altitude. (The camera will not rise above a certain fixed distance over the terrain, however, so it will dip into deep valleys.) Remember that you must be in a yard definition area to activate this camera, and that as long as you don't switch to another camera, it can be used everywhere.