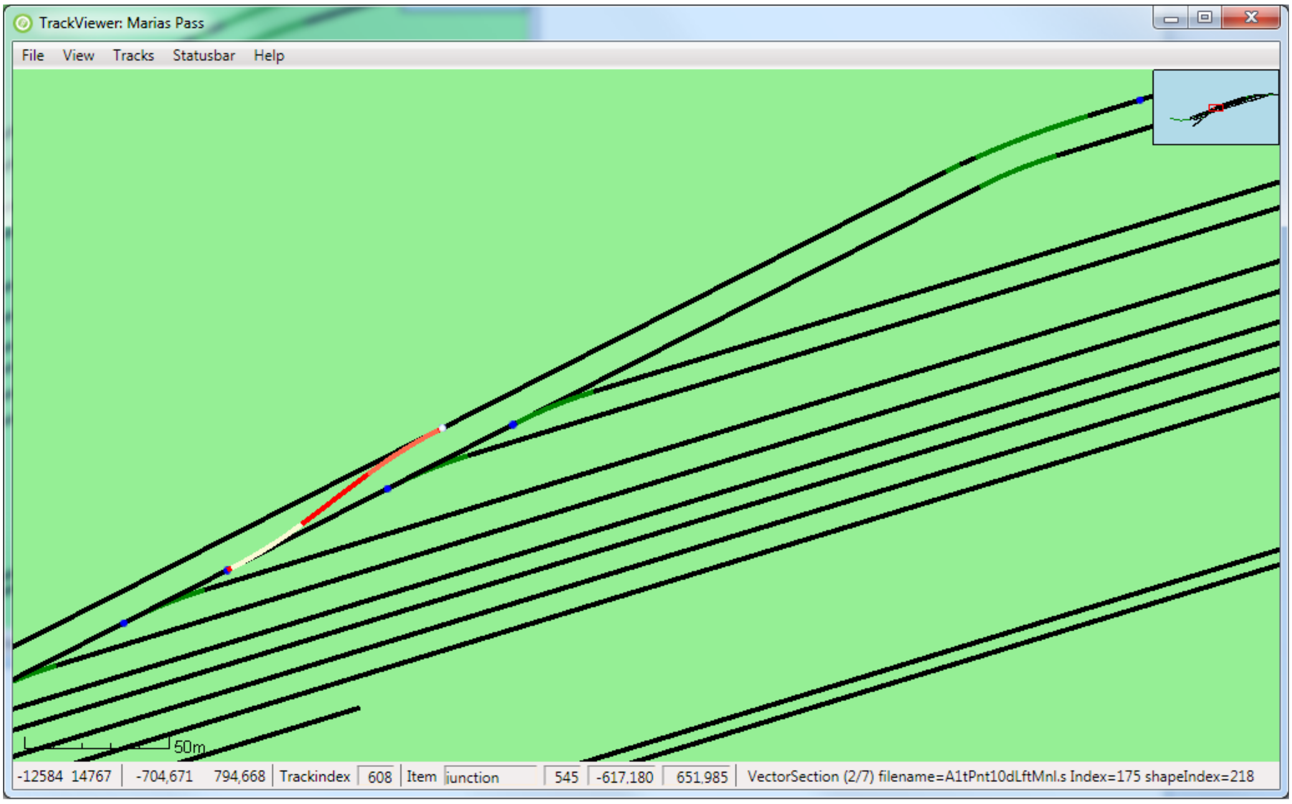
ORTS TrackViewer

Viewing MSTS tracks and editing MSTS paths



-12584 14767 | -704,671 794,668 | Trackindex 608 | Item junction 545 -617,180 651,985 | VectorSection (2/7) filename=A1tPnt10dLftMnl.s Index=175 shapeIndex=218

Contents

1. Introduction

ORTS TrackViewer is an open source program to view tracks and all track items from a MSTS (Microsoft Trains Simulator) route and to edit MSTS paths (as used in activities). The viewing part of TrackViewer is very similar to the program MSTS TrackViewer, which is no longer developed. The ability to edit paths is new.

Note that also this documentation is still under development.

2. Installation and support

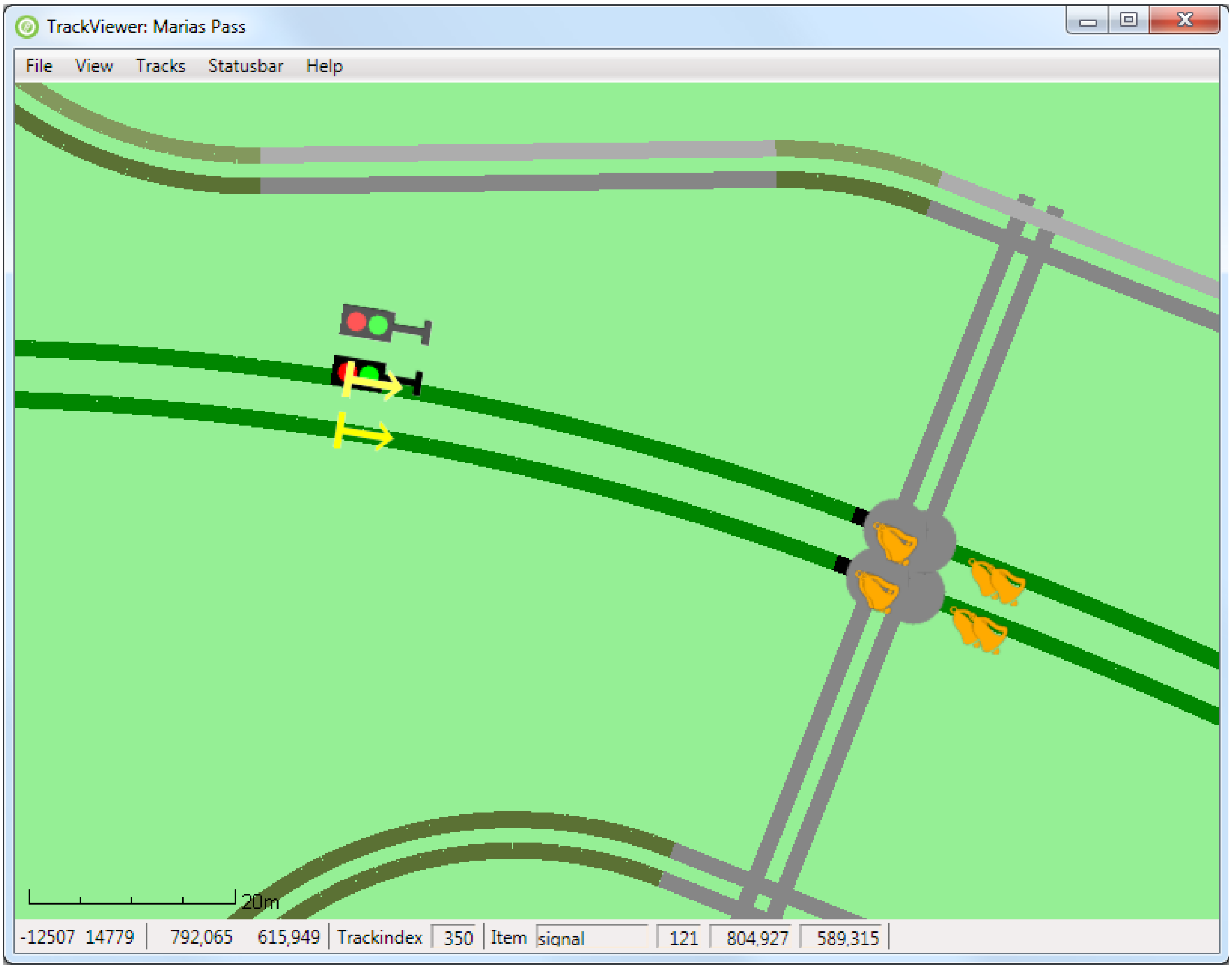
TrackViewer is current part of the Open Rails Transport Simulator (ORTS, see www.openrails.org). It is available both in source code and as a pre-compiled binary as part of the ORTS distribution. This means that if you installed a recent (experimental) version of ORTS, you will have TrackViewer as well. Its executable is called Contrib.Trackviewer.exe. TrachViewer will not run independently of ORTS, since it reuses parts of the code of ORTS. For more information on installing an experimental version of ORTS, see its web-site http://www.openrails.org/experimental.html.

Although TrackViewer is part of the ORTS release, it does not have the same status. There is no promise from the ORTS team to fix all issues. Since this is open source and based on spare-time contributions of one or a few coders, there are no guarantees. Support can be found in the same way as for ORTS itself. That is mainly via the forums on Elvas Tower (http://elvastower.com/, under Open Rails you can use the sub-forums *Discussion* or *Maybe it's a bug*).

3. Viewer manual

When running TrackViewer for the first time, it will try to find your MSTS installation. If it does not find it (or in case you did not install MSTS), you can select the install directory under the File menu. You can then load a route, again via the File menu. Selecting a different route can also be done via the File menu. After the first use, you can reload your previous route (this was saved).

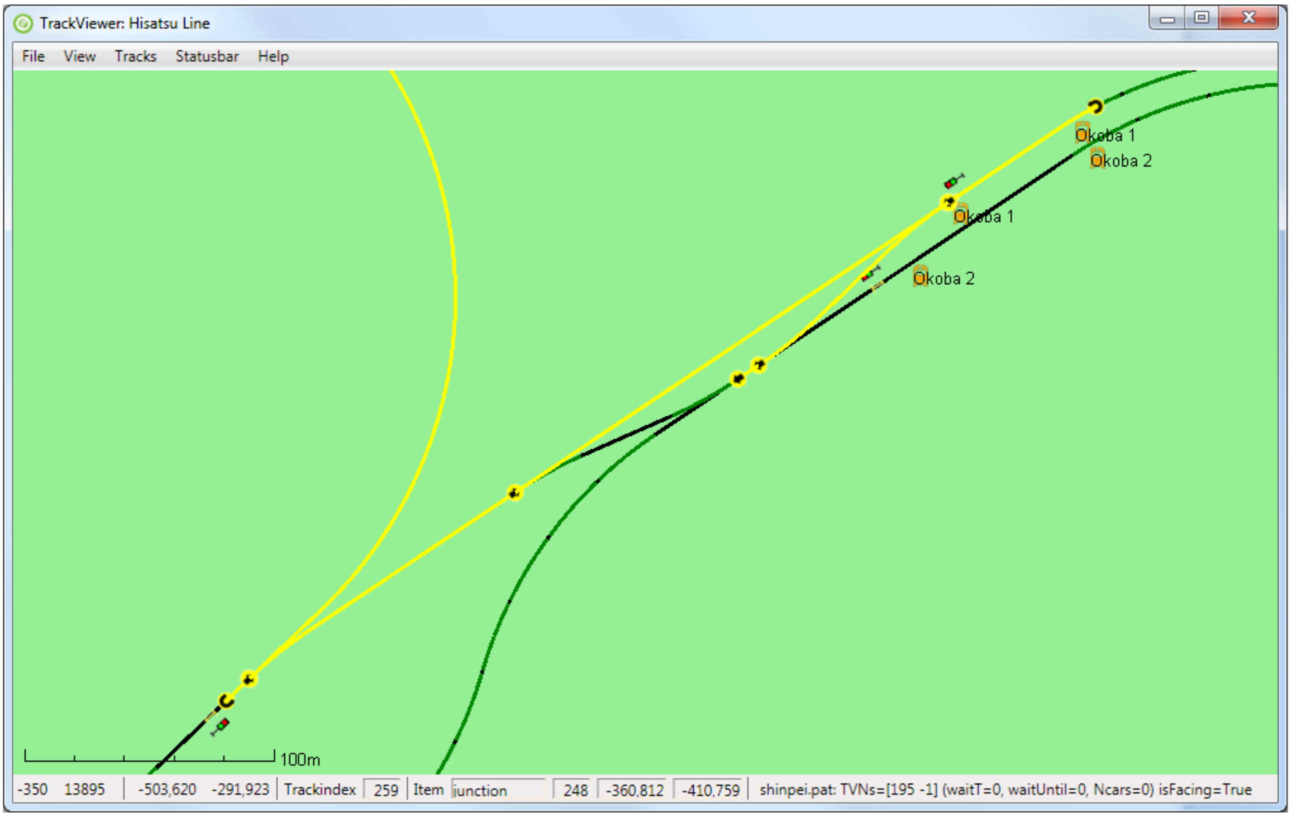
Most of the menu items are pretty self-explaining. You can select which items and which tracks to draw.


TrackViewer: Marias Pass
File View Tracks Statusbar Help
20m
-12507 14779 | 792,065 615,949 | Trackindex | 350 | Item | signal | 121 | 804,927 | 589,315

TrackViewer will keep most of your settings. This includes the install directory, the last route, and what you selected to be drawn on screen and what not. In other words, almost all selections work as preferences at the same time.

A number of menu items have keyboard shortcuts (also mentioned on the menu items). Other shortcuts can be found under Help. Zooming and shifting works best using plus and minus keys ('=' and '-') or the mouse wheel. Pressing shift during zooming with the mouse wheel gives you finer control. Pressing shift Zooming is centered around the mouse position normally. Shifting the view window can be done using the keys a-s-d-w, or with the mouse (left-mouse button pressed).

TrackViewer also supports the drawing of paths, as used by activities for both player train and AI trains. These paths are defined in the PATHS directory of the route and have extension .pat. Drawing can be done in two ways. Drawing the raw .pat file itself will draw a very crude path, containing only straight lines. Drawing 'processed path' is the more advanced feature and uses the ORTS AIPath code. Initially only the first part of the path is drawn. The length of the drawn path can be decreased/increased with the PageDown and PageUp keys. Shift-PageUp will draw the full path, Shift-PageDown will only draw the starting point. Using the key 'c' will center the view window on the last drawn point of the path. It can be kept pressed during zooming or during the decreasing/increasing the path length.



```
-350   13895 | -503,620   -291,923 | Trackindex  259 | Item  junction   248  -360,812  -410,759 | shinpei.pat: TVNs=[195 -1] (waitT=0, waitUntil=0, Ncars=0) isFacing=True
```

4. Path editor manual

At this point in the editor is still under construction. It is the intention to have a working version always, although it might not contain all features yet.

The Path editor is intended to be an editor of MSTS paths. It is not intended to be an editor for new ORTS-only paths and activities (because that will be build from scratch including different file formats and functionality). The editor is also not intended to be an activity editor for MSTS (it will not contain a consist editor for instance).

4.1. Path-editor menu

The path-editor menu contains various items to deal with paths.

Without editing, you can still load a .pat file. Once loaded the initial part of the path will be shown. You can center the last drawn node in the visible window with 'c'. You can also save the path. Note that the .pat file so created will have a different ordering of nodes than the original .pat file (the path editor will create a normal ordering).

To enable editing you either have to load a path from file and then enable editing (under the path-editor menu), or you have to create a new path (which will enable editing for you). It is also possible to edit the meta-data of the path (like name, ID, start location and end-location).

4.2. Context menu

The editor recognizes two kinds of 'active' locations (both identified with a small ring around the relevant location)

1. The node that is closest to the location of the mouse is called the 'active node'. Note that only actually drawn nodes can be active. The reason for this is that this makes it possible to deal with nodes that have the same location, for instance when the path has been reversed

and goes over the same junction(s) again.

2. The location on the track that is closest to the mouse is called the 'active track location'. If there are no nodes defined (so for a new path), this can be any location. If there are nodes defined, only drawn track can contain an active track location.

The context menu will popup when clicking on the right mouse while editing is enabled. It will contain actions that can be performed to the current active node or the current active track location.

4.2.1. Active track location

Actions related to the path itself

- Place start point. This will create the first point of a new path.
- Place reversal point. Place a point where the train will need to reverse. This will remove any activity-related points that have been defined further along the path.
- Place end-point. Note that we make a distinction between what happens to be the last node, and has actually been defined as an end-node. Once an end-node has been defined, actions that would invalidate this end-node are not allowed (with the exception of course of removing the end-point itself). This is to prevent accidental changes.

Actions related to activities on the path.

- Place wait point. Place a point where the player doing the activity will have to wait. A popup will appear where you can edit the meta data of this point (like the time to wait, or the time until you have to wait).
- Place (un)couple point. An (un)couple point is a point that within the MSTS editor would be part of the activity. A popup will appear where you can edit the meta data of this point (like the number of cars to couple and whether you want the cars to couple or uncouple).

4.2.2. Active node

Actions related to the path itself

- Change start direction. The initial direction of the path, once a start point has been added, depends on the default in the track database. There is about 50% chance that you would like the other direction.
- Take other exit. For default direction through a facing junction is the main route as defined in the track database (actually, tsection.dat). Taking the other exit simply changes the current exit taken from the junction. When there is no end-point defined, you have a lot of freedom to do this. When there is an end-point taking the other exit is only allowed in those situations where the editor can find a way to reconnect to the existing path itself. For more complex situations, use Cut and store tail (see below).
- Add passing path. Instead of taking the other exit, it is also possible to add a passing path (also called siding path). This is also only allowed in those situations where the editor can find a way to reconnect to the existing path itself. For more complex situations, use Start passing path (see below)
- Cut and store tail (not implemented yet, so open for change). For complex restructuring of paths (including dealing with broken paths), this will basically cut the path into two at the current location. This location and the rest of the path will be stored (we call this the 'tail'). You can then edit the first part of the path in any way you like, until it is time again to recombine the first part with the tail.
- Start passing path (not implemented yet, so open for change). Start a complex passing path

here. No other actions are allowed until the passing path is reconnected again to the main path.

Actions related to activities on the path

- Edit point data. This will popup a menu to edit the meta data of a wait or (un)couple point.

Actions related to removing a point

- Remove end point: This will remove the end-point. Note that paths loaded from file will always have an end-point when loaded.
- Remove reversal point. Simply remove the reversal point. Obviously this will impact the track quite significantly.
- Remove wait point. All meta-data will be lost.
- Remove (un)couple point. All meta-data will be lost.
- Remove start point. This obviously will lead to an empty path.
- Remove passing path. Remove the passing path that starts at this junction.

Drawing related

- Draw path until here. This does not affect the path itself, only which part of the path is drawn. When the full path is drawn, for each location only the last drawn node can be active. In situations where the path goes multiple times over the same track (e.g. due to reversal nodes), this can be an issue. It can easily be solved by reducing the number of nodes actually drawn. However, for long paths it is very inconvenient if the path can only be extended or reduced by only one node at a time. Draw path until here allows you to select until which node the path will be drawn (and reduced or extend again from that node).

4.3. Broken nodes and paths

(not completely implemented).

The MSTS .pat file stores the various nodes using the location. This makes it independent of the index a junction or track happens to have in the track database, which is good. Sometimes, however, the track database has been updated (tracks have been updated or moved, junctions have been added or removed, …). In those cases a stored .pat file might suddenly be broken: a defined path node can no longer be linked to a correct track location. Broken nodes will be indicated by a cross through the node. Furthermore, the path can no longer be drawn along the track: straight lines will be used.

To correct broken paths, the following options are available:

- For situations where junctions are only changed a bit, it is possible to re-snap the broken node to the closest junction (not implemented yet)
- For other complex situations, use the 'cut-and-store-tail' functionality (see above, also not yet implemented).

4.4. Limitations

The fact that there are limitations in the path editor is related to the limitations in the file format and capabilities of MSTS itself (obviously, apart from bugs or the lack of someone willing to implement a feature). Since this editor is intended to yield MSTS-compatible paths (that can also run within ORTS), not all possible features can be implemented.

The list below might not be complete (or perhaps not even accurate)

- It is not allowed to have any activity-related points (wait or (un)-couple points) between the start end the end of a passing path, and neither is it allowed to have a reversal point. Obviously, also no start and end-point are allowed.

5. Keyboard commands and mouse behavior

Some of the keyboard comments can be seen from the menu as well. Here we try to document all of them.

5.1. Viewer

Zooming and moving the view window

- = Zoom-in. Keeping it pressed will keep on zooming pressed.
- - Zoom-out. Keeping it pressed will keep on zooming out.
- r Zoom-reset: view the whole track.
- z Zoom-to-tile: zoom to a level where exactly one MSTS tile (2048m x 2048m is visible).
- a Shift-left: Shift the view-window to the left.
- d Shift-left: Shift the view-window to the right.
- s Shift-left: Shift the view-window down.
- w Shift-left: Shift the view-window up.

Toggling what is visible/drawn

- F5 Show speed-limits.
- Shift-F5 Show mileposts.
- F7 Show signals.
- F8 Show platforms.
- shift-F8 Show platform-names.
- F9 Show sidings.
- Shift-F9 Show siding-names.
- F11 Show path (needed to be able to edit it!)
- Shift-F11 Show path from raw information in ..pat file (which does not use track database). This will not be updated during editing.

Mouse behavior

- Shift-drag-left-mouse button: shift the view-window with the mouse movement.
- Scroll-wheel: Zoom-in or out.
- Shift-scroll-wheel: Zoom-in or out, but slower (to enable more precise control).

5.2. Path Editor

Keys:

- c shift the view window such that the last drawn-node is centered. You can keep this button down while drawing more or less of the path using the next keys.

- PgUp Draw an extra node of the path (unless at the end-of-path)
- PgDn Draw one node less of the path (this does not change the path itself!)
- Shift-PgUp Draw all of the path.
- Shift-PgDn Draw only the start node of the path.
- Ctrl-z Undo the last edit.
- Ctrl-y Redo (only possible when at least one Undo has been done).

Mouse behavior

- Additional mouse button 1: Undo
- Additional mouse button 2: Redo

6. Future development

The path editor is still under construction, so that will be developed,

Any future development depends on the wishes and needs of the community. I created ORTS TrackViewer as a debugging tool initially (working on paths), and it grew into something much more.

The following items have already been requested. Some of these might end up being implemented

- Make a path editor for MSTS paths. This is under construction
  - apparently including grade crossings would be a benefit compared to MSTS.
- Currently it is possible to search for tracknodes and trackitems
  - Possibly it would be nice to have problems in the route directly available from the viewer (instead of getting this information from OpenRailsLog.txt).
- Add possibility to import and export routes. So people can have a look at a route without having it installed.
  - Currently this is already possible by copying global tsection.dat, route-specific tsection.dat, <route>.tdb (and <route>.rdb), and for the moment also <route>.trk.
  - Making an export/import routine would basically mean to make a different file format to write and read (at least part of) the information in these files. It is not clear whether this is worth the effort.
- Make TrackViewer independent of XNA. This prevents people to have install XNA. Currently this is quite a big change, and it would probably also need the code to be independent of ORTS itself.