FVD++ Documentation

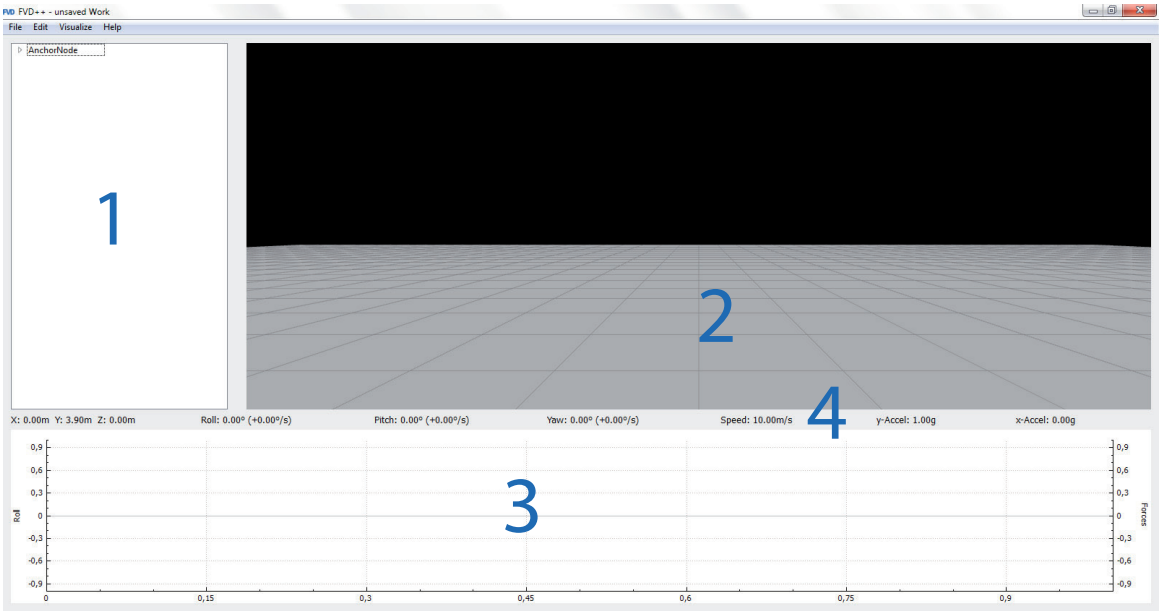
v0.31 beta

Table of Contents

1 User Interface

1.1 Basic layout

After having started FVD++, the main window appears which is splitted up into four areas:



The sections list on the left hand (1) shows a list of all the sections that the coaster consists of. Each list item can be expanded to show further options. At the beginning, the *AnchorNode* is the only section on the list and can't be deleted. Each section can be renamed by clicking on the name itself.

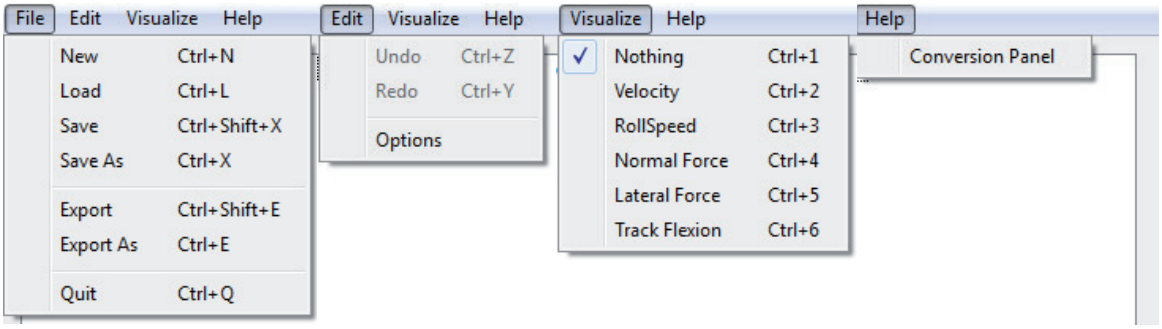
The 3D viewport (2) will show a 3D-rendered model of the coaster. The *AnchorNode* being only a point isn't can't be seen here. In v0.31 orthogonal views aren't implemented yet, which will likely be done in later versions. More on moving the camera and the POV mode see **Viewport and camera controls**.

Next is the graph panel on the bottom of the window (3). It shows the transition graphs laying on top of each other and representing different components which can be manipulated in order to shape the track.

Lastly, the statistics panel (4) which shows all kinds of different values that will be explained later. All of these values refer to the end point of the transition being currently selected in the graph panel or the current position if the POV mode is active.

1.2 The main menu

At the very top of the main window you'll find the menu which consists of the following sub-menus:



In *File*, the project files (extension: *.fvd) can be loaded, saved and transfered to NoLimits which will open up the export panel (see **Exporting**). *Export As* will do that on every click, whereas *Export* will just save the nlelem-file with the exact same settings which were used in your last exporting action without asking.

The *Edit* panel allows to undo and redo all actions and clicking on *Options* will open up a new window to define the color settings, the measure and the OpenGL version in case of problems with the program or when working on a Mac.

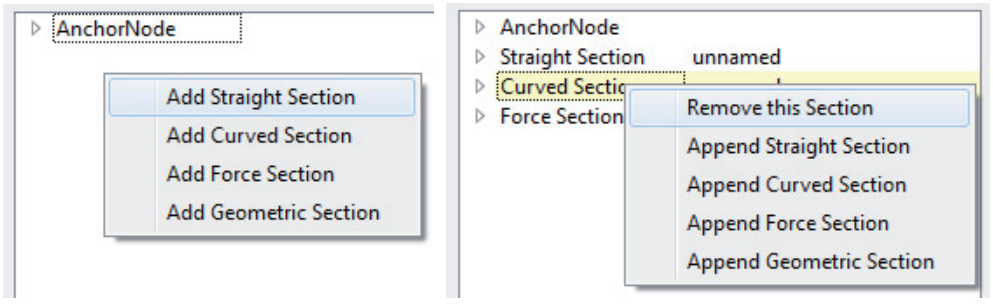
The visualisations have the effect of coloring the track dependently on the attribute being chosen to be kept in check. Having it set to *Nothing* will result in a blue track model without any visualisations.

The *Conversions* panel in *Help* stores all the heartline values for every coaster type of NoLimits 1.8 and performs some basic measure conversions (currently speed and length).

1.3 Using the sections list

Some aspects regarding the usage need to be explained first:

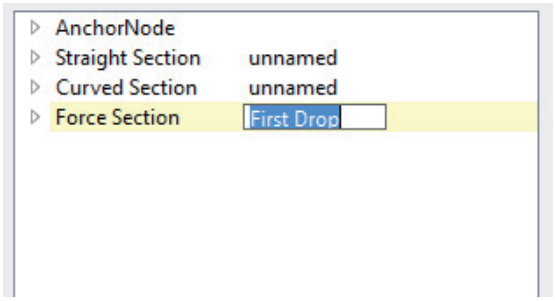
Adding a new section is done by right-clicking on the blank area below the *AnchorNode* and selecting the type of the new section:



Adding and removing sections

Left picture on page 2: right-click on existent sections to delete the one selected or append a new section of any desired type. Double-click on the name of the section (preset: "unnamed") allows typing in a new name, which helps a lot when exporting different parts of your coaster.

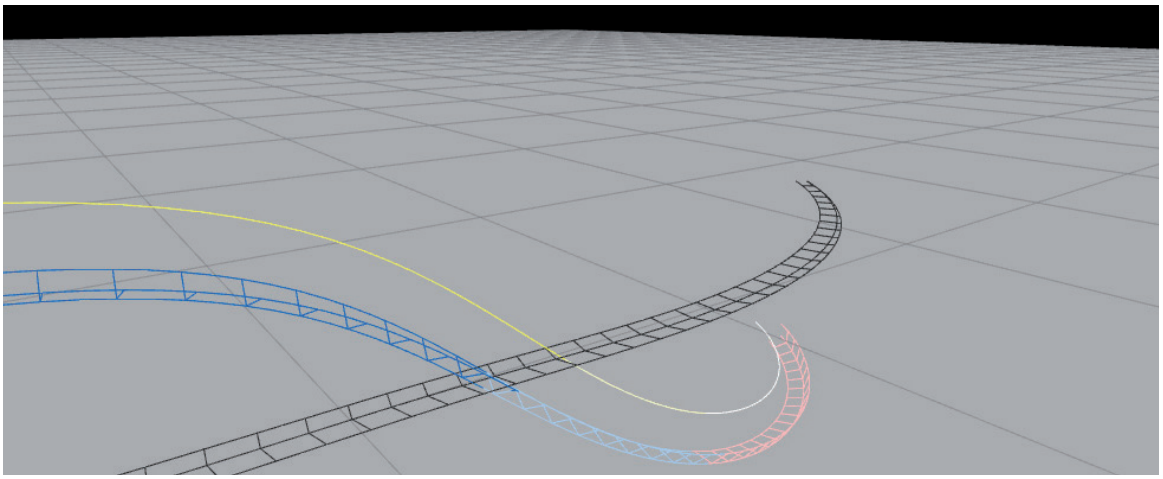
The different section types will be explained in the corresponding chapter.



Renaming sections

1.4 Viewport and camera controls

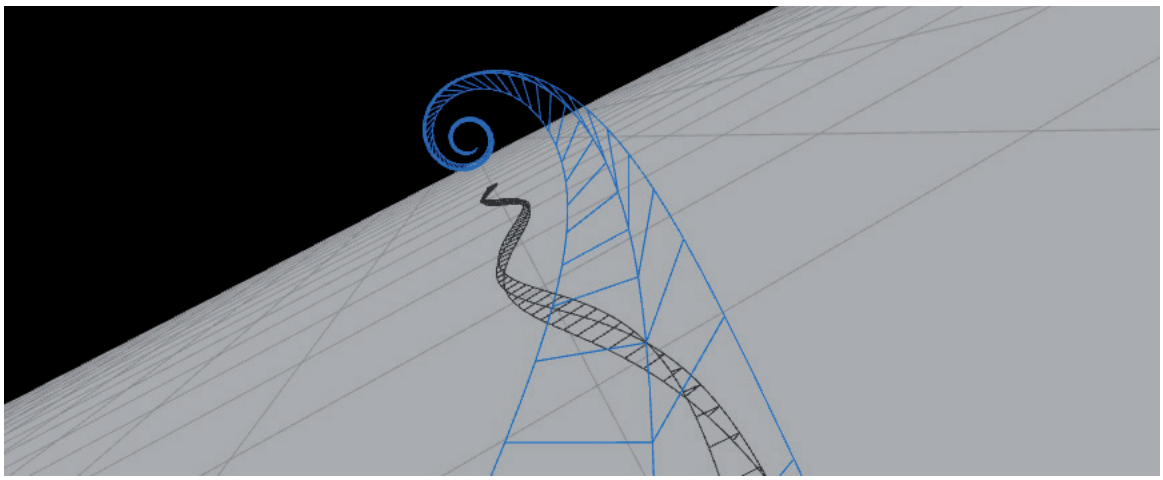
Right next to the section list there is the viewport which shows a simplified model of the coaster inside a virtual room. Unlike in NoLimits, the track can exist beneath the terrain level. The grid consists of squares of 10 by 10 meters.



As being configurable in the *Options* window mentioned before, the track has different colors, even if the visualisiations are turned off. As demonstrated in the screenshot, any track existing in negative heights gets displayed different than normal. The part of the track which is effected by the transition you are working on is always displayed green.

All of the following controls demand a right-click inside the viewport so that the mouse pointer disappears. Now the camera can be turned into any direction by moving the mouse. Flying forwards/backwards is done by pressing W or S, paning to the side by pressing A or D.

Entering / leaving the POV mode is done by pressing the space bar (again provided your mouse pointer isn't displayed):



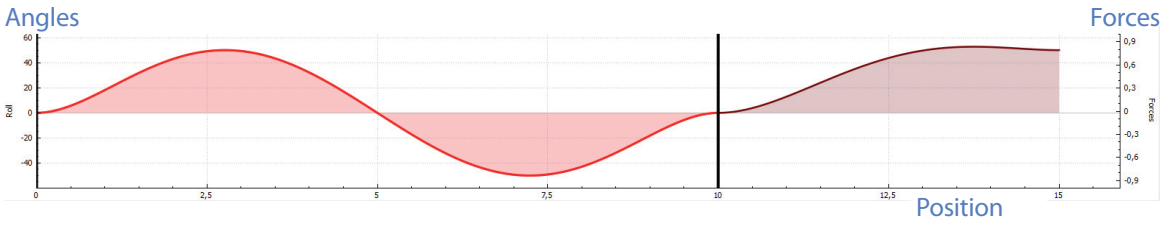
The speed the camera has when riding the coaster in the POV mode is the calculated speed of the coaster which allows in-program checking of the smoothness and pacing of the track. For some section types, a constant speed can be set up which then will also be used in the POV mode. Using a constant speed for the whole track isn't possible.

In each one of the two viewing modes (normal, POV), the speed can be either halved by pressing Ctrl or doubled by pressing Shift while moving.

1.5 Using the graph panel

The graph panel located at the bottom of the window displays mathematical graphs which are representing dynamic values to form the track itself. Besides any detailed explanation (see corresponding chapters), only the handling of the transition themselves will be explained.

The panel has two axes where the one on the bottom represents time (or distance) and the two on both sides angular and force components. This separation is needed because the graph panel usually shows more than one graph so there will be graphs with different meanings therefore needing two different y-axes.



In this example, the red graph consists of two transitions, where the first one of them is selected (lighter color). A transition can be selected by left-clicking on the line. The thicker vertical line represents the connection of two transitions.

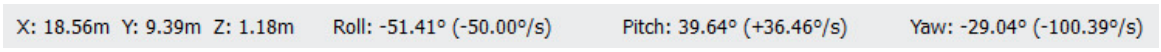
The most basic operation is appending a new transition after the selected one by right-clicking. The selected transition also can be deleted that way (both left picture).

The same technique can be used to insert a new transition between two existing ones, but there are two ways to achieve this (right picture): Either by using *append transition* on the left transition or using *prepend transtion* on the right one.

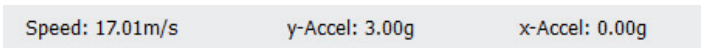
Zooming in and out works by scrolling and paning by left-clicking and dragging the curves horizontally into position. Any vertical adjustments are done automatically.

1.6 The statistics panel

This panel shows different values which are all refering to the endpoint of the currently selected section or the current position in the POV mode.



The first half gives information about the position and the direction of the end point. After the angles there's an additional number in parantheses which is the change rate of that angle (see **Pitch and Yaw Change**).



The second half shows the speed and the forces in both vertical and horizontal direction.

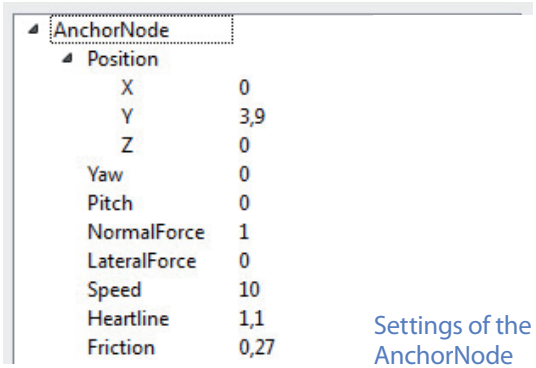
If any of these values shows "nan" (not a number) followed by the unit, the speed to complete the selected section / transition might has reached zero. As a consequence, any following track can't be rendered and the exported track surely is unusable.

2 Section Types

2.1 The AnchorNode

The *AnchorNode* is always the first item in the sections list and can't be deleted or moved. It's used to specify important settings of both the starting point and the rest of the track.

By clicking on the small arrow icon, the list of possible options expands. The first item is the *Position* of the starting point, which also needs to be expanded to change the coordinates themselves. *Y* stands for the height.



Settings of the AnchorNode

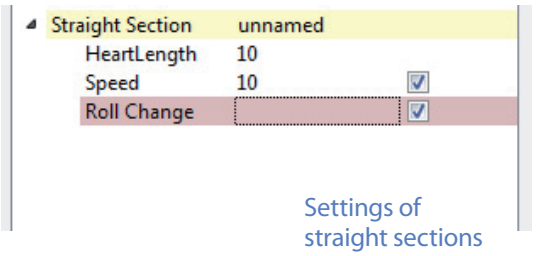
Please be aware that FVD++ uses the center point between the track rails as a reference for any coordinates concerning position, in contrast to heartline-based coordinates of newton, for example. Any component's function is calculated from the heartline position.

After the starting position coordinates, the settings for the *AnchorNode* include the yaw and pitch angles together with the starting forces and speed. Only the heartline and friction value at the end will have an effect on the whole coaster. To convert friction coefficients from newton to FVD++, multiplying by 9.81 is a good start for further experimentation.

2.2 Straight Sections

These are the simplest section types in FVD++, having only three different settings:

HearthLength defines the length of the heartline path of the section.



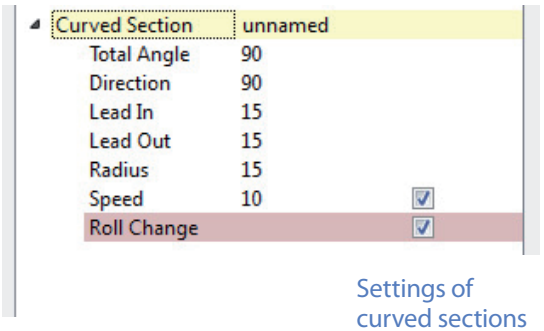
Settings of straight sections

If *Speed* is checked (clickbox on the right), FVD++ will work with the value being set there, otherwise the section will behave exactly the same as a blank piece of track including friction and accelereation due to gravity.

RollChange is a transition type controling the banking of this section and it will be explained in greater detail later. When the graphs are getting ignored by now, the section won't have an effect on the track banking at all (like the station and transfer tracks, for example).

2.3 Curved Sections

These sections are parts of a circle with further configurations, which are the following:



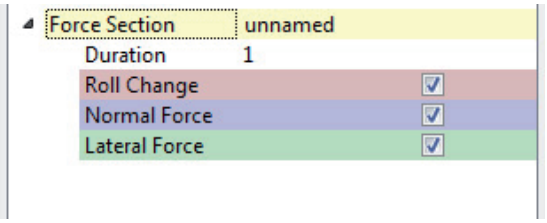
Settings of curved sections

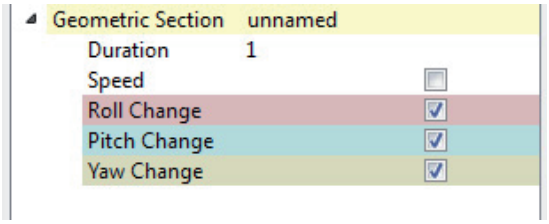
The *TotalAngle* defines at which total angle the track has to bend, no matter at which *Direction* the bend goes. The *Direction* is pre-set at 90 degrees which results in a track bending to the left. At 0 degrees, the curve would point upwards, downwards at 180 degrees.

Lead-in and *Lead-out* define which angles of the *TotalAngle* will be used in order to have a smooth blend from completely straight track to track that has the specified *Radius*. Please note that the Sum of the Lead-in and Lead-out can't exceed the *TotalAngle*.

Speed and *Roll Change* are serving the exact same functions as they do in *Straight Sections*.

2.4 Force and Geometric Sections





These two are by far the most sophisticated section types FVD++ has to offer. They are quite similiar in use and, most importantly, transition graphs have to be used in order to shape the track using these section types:

The only information that is displayed here is the *Duration*, which represents the time it takes to complete the section. On *Geometric Sections*, there's also the already discussed option to lock the speed to a fixed value.

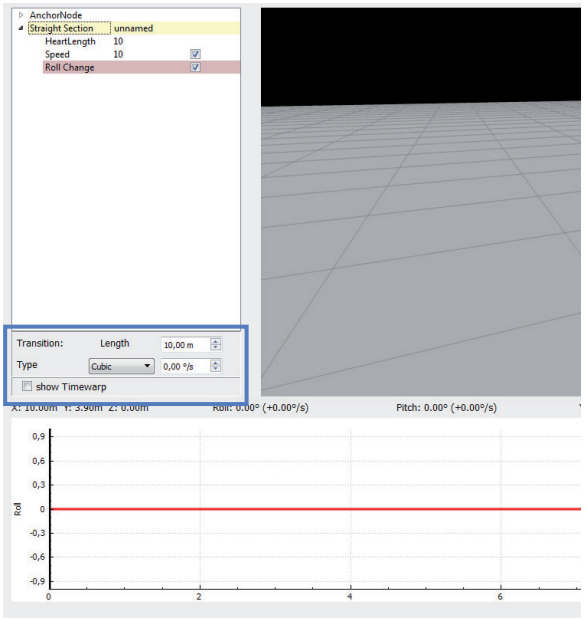
To take a look at the transition graphs themselves and how to use them, see the next chapter.

3 Transition Graphs

3.1 The transition panel

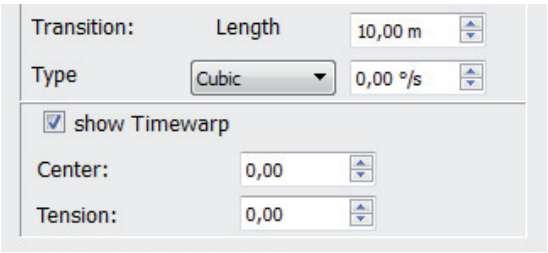
Every time a new section that supports transition graphs gets added, a new panel will appear on the left side of the program window. Taking a *Straight Section* as example:

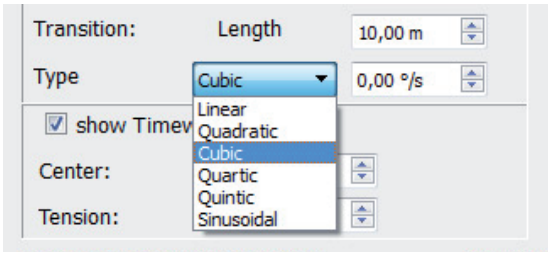
The red line is the *Roll Change* graph which is shown in the sections list as well. As already explained, it's possible to append new transitions on this one and deleting transitions.



The new panel right above the graph panel is called transition panel. From here, further settings for the transtion that is currently selected in the graph panel can be specified. In this case, transition *Length* and *Type* including the transition value next to it.

Length specifies the expansion of the selected transition in the track, this time being measured in meters (see **Function Argument**). The transition *Type* allows





selecting different kinds of curves from a drop down menu (right picture) so even more control over the transition itself is possible (see **Transition Types**). The field right next to it is used for the transition value which defines how much the value will change over the course of the transition.

The timewarping panel allows further adjustments of the form of the selected transition. How timewarping behaves in paricular is dependent on the transition type (see **Timewarping**).

3.2 Manipulating transitions

In each one of the number fields that can be found in FVD++, any numerical value containing 2 decimals at a maximum can be entered. Either a comma or a full stop as a decimal mark is allowed (this depends on your operating system). Here's an example of using custom values for the selected transition:

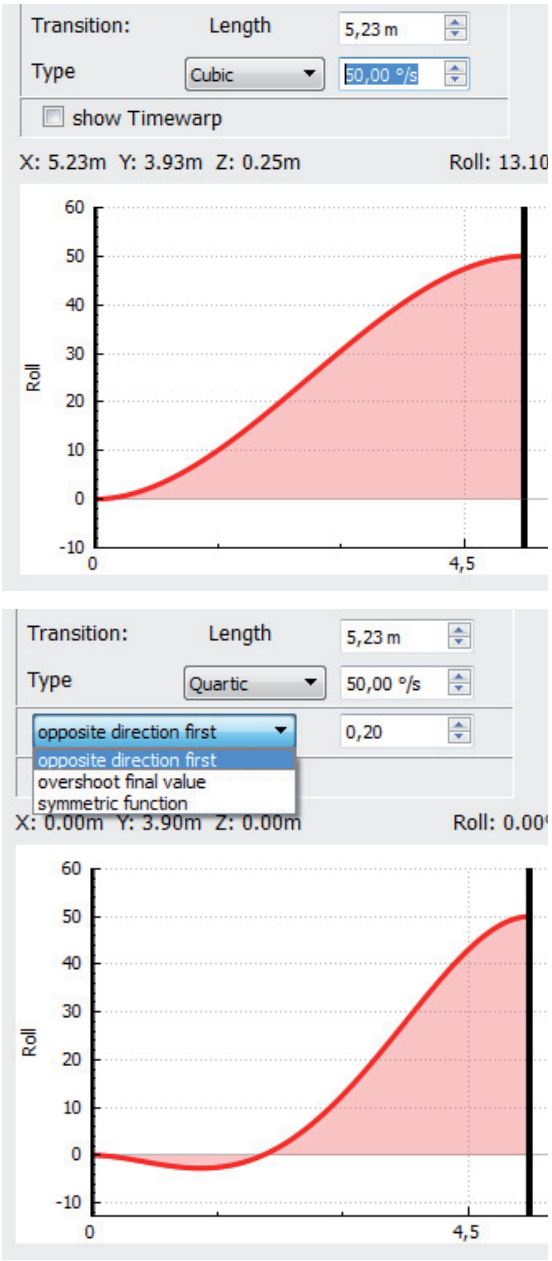
The transition shows a bend upwards as it changes its value by +50 °/s (from 0 to 50 °/s). The form is defined by the *Length*, transition *Type* and timewarping settings.

Transitions can be edited that way in any situation, even if other transitions are following or different graph types are being shown in the graph panel at the same time.

The transition you can see here is a cubic polyomal function which blends in and out. It's the default as being used most often. Because it's a simple transition type, only one value is needed to define its shape

When choosing a more complex transition type such as *Quartic*, a new menu appears allowing to choose different variants of this type. For example, now it's possible to specify how much the transition goes into the *opposite direction* before the final change value is reached.

Each transition type has its own specialised variants allowing a great variety of different transition forms.

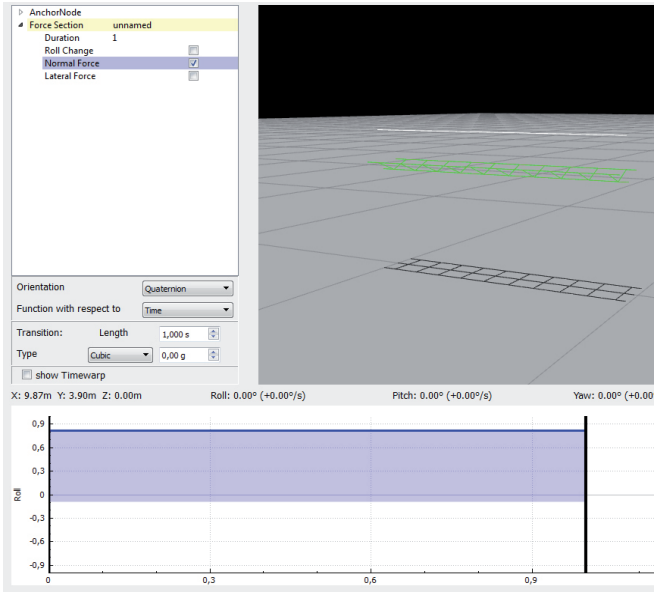


Transition variants

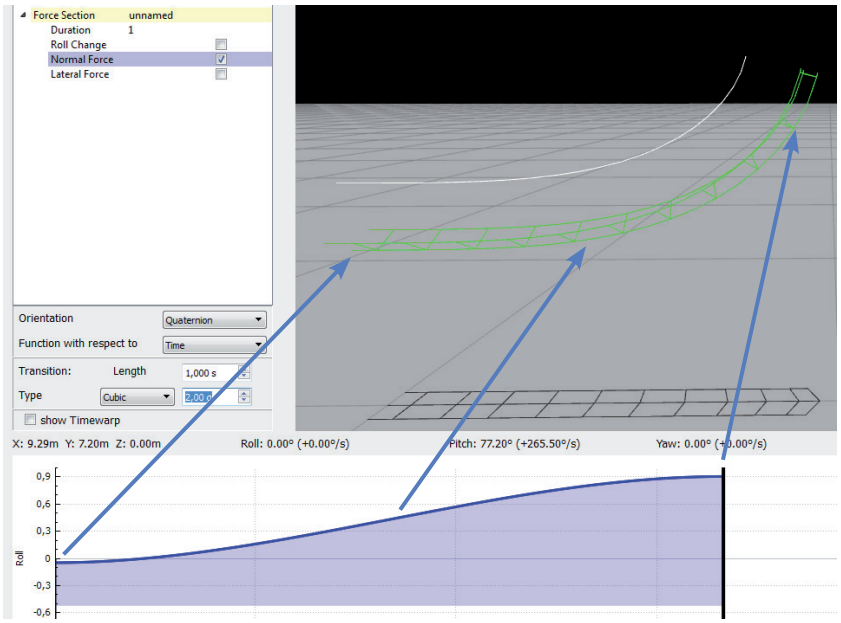
3.3 Normal Force

As FVD++ is a force vector design tool, the track can be formed based on forces the user has specified in the transition graphs including real-time adjustments on them. The most important component defining the final shape of a *Force Section* is the *Normal Force* graph.

After having added a new *Force Section* in the sections list and expanding it, three transition graphs appear in the graph panel below the transition panel. Uncheck the other two components so there's only the blue graph left.



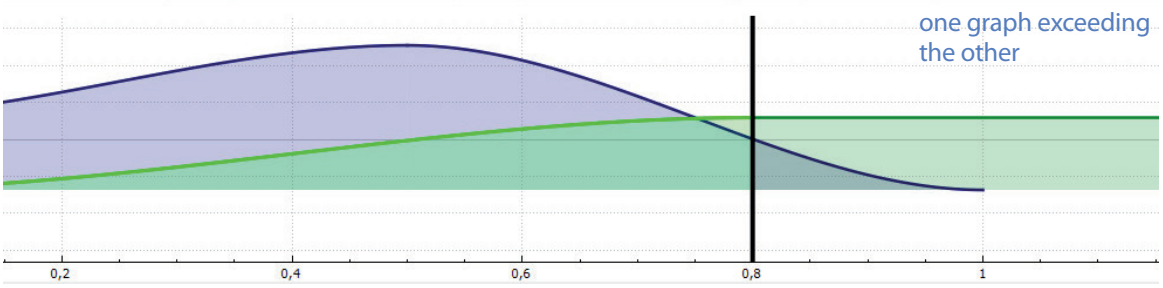
Now the transition can be altered only concerning the vertical forces on the track. Changing the transition value to a positive number (e.g. 2g) will result in a track bending upwards so it produces the increasing g-force specified. As the section starts with 1g (see the *AnchorNode*) and it has one transition of +2g, the force at the endpoint of this section will be 3g. Therefore, the transition defines a change of the absolute value of the normal force over time. When entering negative numbers, the track will bend downwards producing lower forces.



Points of the same time at both the transition and the rendered track

3.4 Lateral Force

Graphs concerning the *Lateral Forces* are following the very same principles like the *Normal Force* ones. FVD++ is capable of showing multiple graphs in the graph panel at once while each transition is still selectable on its own.



The picture shows two graphs lying on top of each other, with one *Lateral Force* transition being selected, thus editable. At this point it's worth mentioning what happens if one graph is shorter than the others (*Normal Force* ends after 1 second in the example). As the section is only defined where all active graphs exist together, it will end at this point even if transitions are getting cut off then. This effect needs to be handled carefully because it might cause bumps.

3.5 Absolute value vs. changing value components

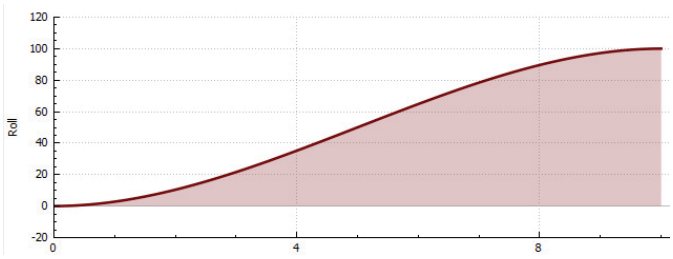
In newton, for example, the roll can be treated exactly the same way as normal and lateral forces: The transitions stand for changes to the absolute value (see page 10).

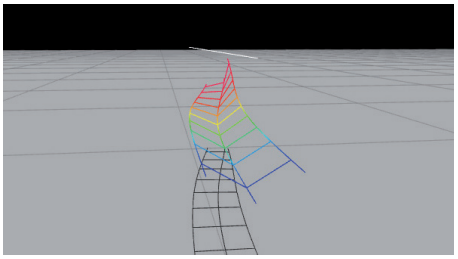
A completely different principle stands behind *Roll Change* and other changing value components: As the names already imply, the graphs of this components are representing a change rate, so in other words, how much the component in question changes in an increment of time or distance. *Roll Change* can also be called roll speed or roll rate.

Roll Change is not the only component working in this way: The other two are *Pitch Change* and *Yaw Change*. All three come together in *Geometric Sections*, whereas *Roll Change* is used in every section type.

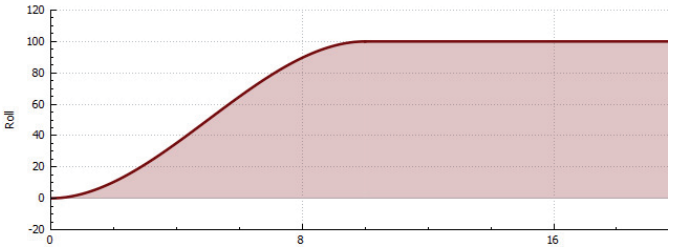
3.6 Roll Change

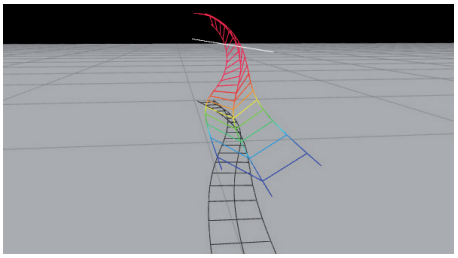
In the following different roll change graphs and their visual apperance with the *Roll-Speed visualization* turned on:



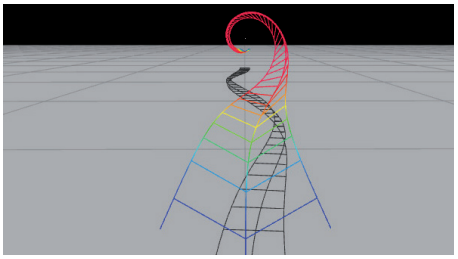


A basic Quintic transition of +100 °/s results in the rolling to happen faster as the section progresses. Positive roll values will always tilt to the left (counter-clockwise).





Appending a zero-transition will result in a constant roll speed after 10m, thus creating the first part of a heartline roll containing a smooth entrance.



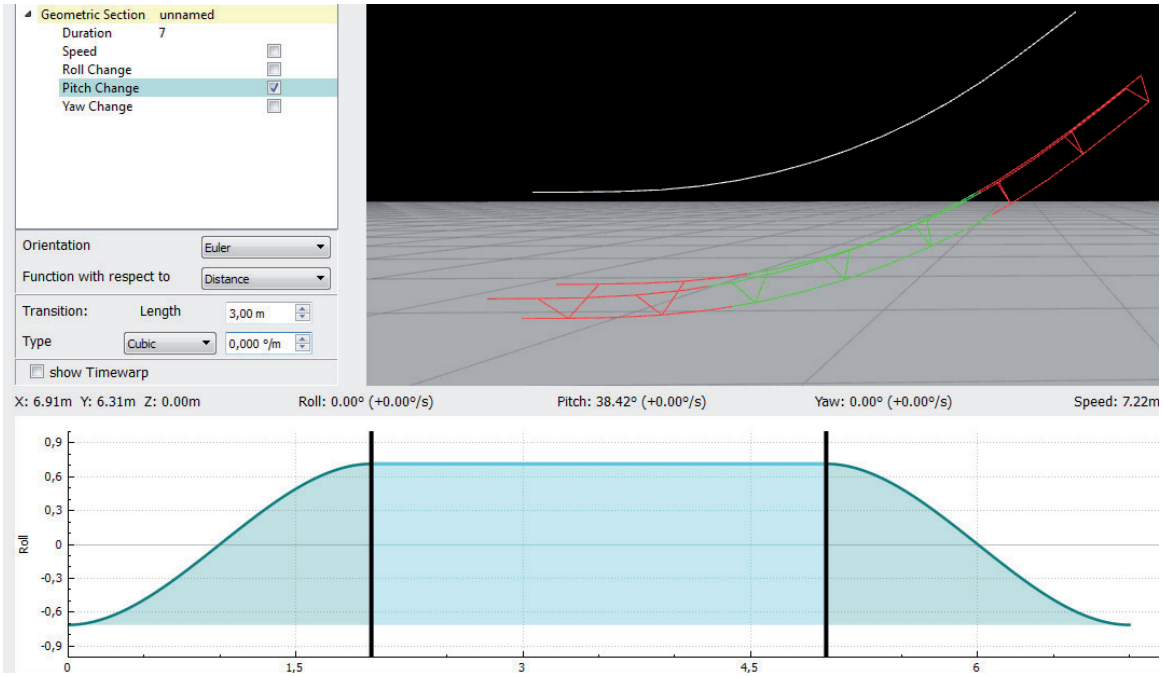
Returning to both 0 °/s and 0° roll by another transition completes the heartline roll.

Speaking more mathematically, the area beneath the *Roll Change* graph represents the actual roll value, whereas negative areas cancel out positive areas (like integrating over a function). To return to 0° roll, the areas have to either cancel out or bring the roll to a multiple of 360°. To stay at 0° roll, the graph also has to be 0 °/s (= no roll speed).

If the *Roll Change* graph doesn't return to zero at the end of the section, the roll change rate will continue to the next section which allows smooth connections between sections of different types (e.g. *Force Section* after *Geometric Section*).

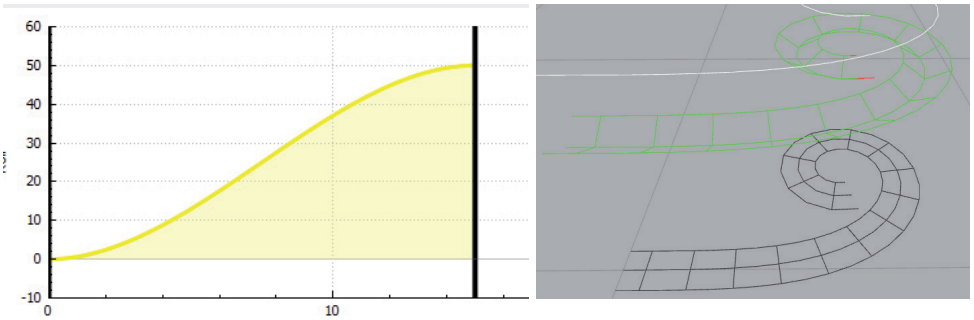
3.7 Pitch and Yaw Change

Similar to *Roll Change* these component types control the pitch and yaw angle changes in the track. This example shows how using only the *Pitch Change* graph can replace curved sections (in most cases), for example as the start of a lifthill:



The constant transition in the middle of the section behaves like a part of a circle, whereas the starting and ending transitions are leadin and leadouts of the bend. The section begins and ends with 0 °/m pitch change, therefore it creates straight endings.

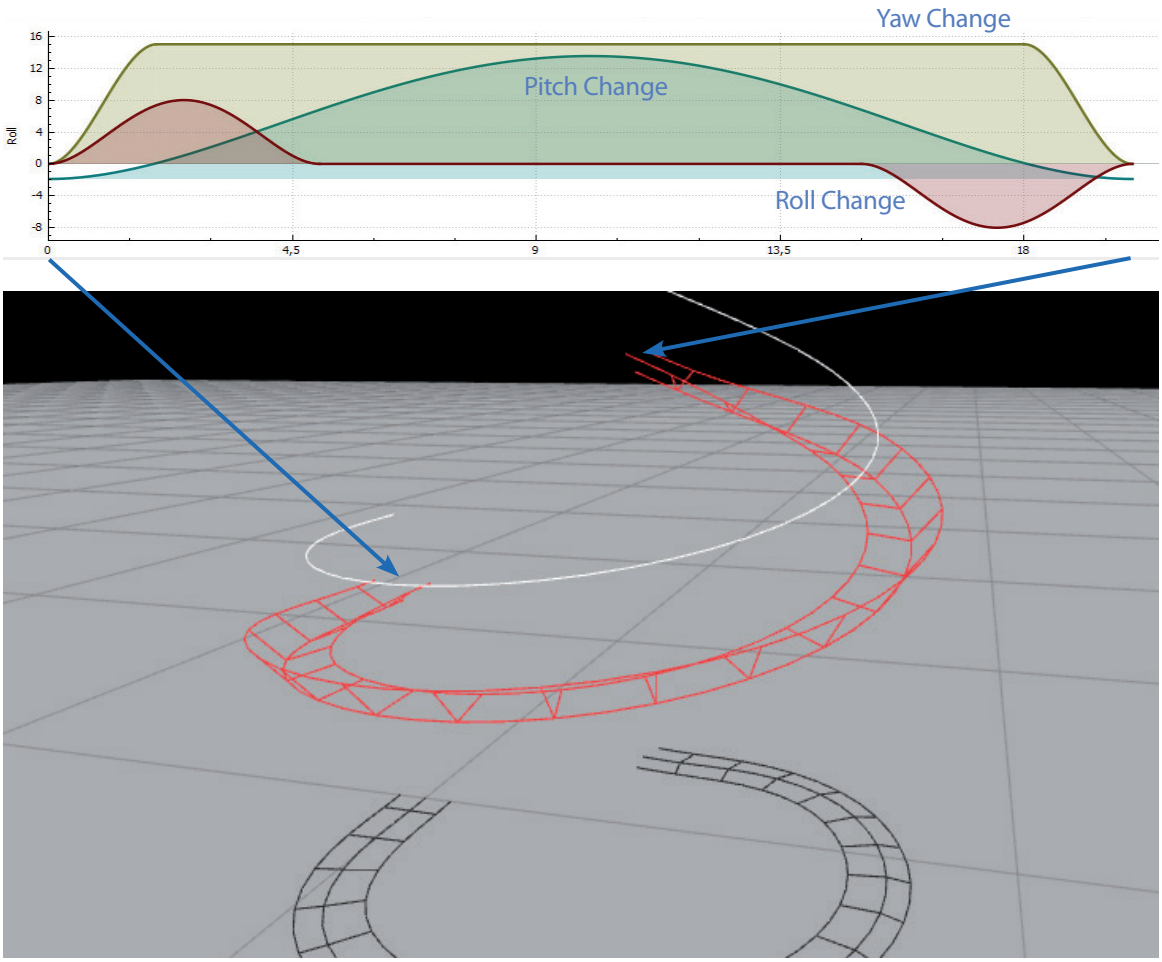
Yaw change behaves exactly the same, only to the side. This is how an increasing yaw change would look like:



In this example, the functions are in respect to *Distance*. For the reason of this, see **Function Argument**.

3.8 Independent components

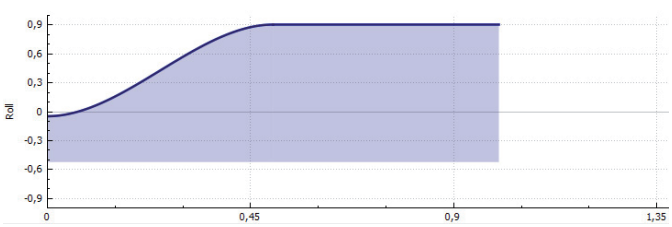
When using *Pitch* and *Yaw Change*, the individual effects of each graph are completely invariant of the direction the track is pointing at or which roll it has. This is an universal rule for the components of *Straight*, *Curved* and *Geometric Sections* to have each component independent from the others. This principle allows designing the heartline path first and altering the roll change afterwards, for example, without the shape of the heartline to be changing noticeably. Although it might look complicated, this is what using this principle can look like:

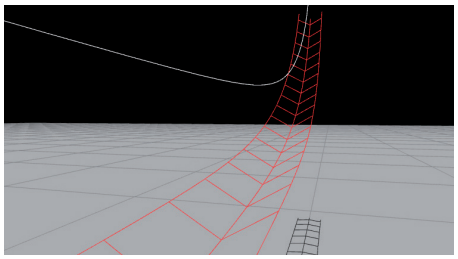


To create this, one would first set up the *Yaw Change* to create a flat curve to the left. Secondly, adding the *Pitch Change* forces the track to bend upwards and thirdly, adding roll will reduce resulting lateral forces (which can be checked using *Visualisations*).

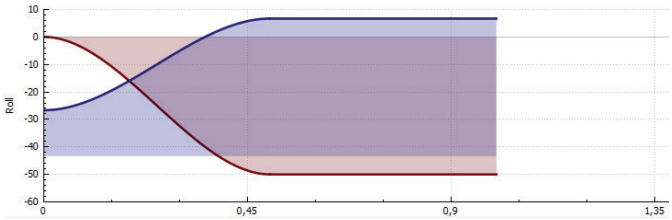
3.9 Dependent components

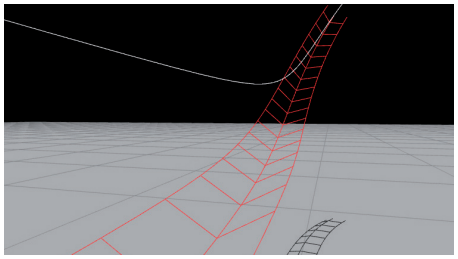
On the other hand, when using *Force Sections* each change on a component's graph will cause the heartline path to change. The reason for this behavior is the lateral force which has to stay at 0g if not further specified in its graph. So FVD++ forms the track exactly the way such that it produces all the specified forces and roll changes together.





The track bends upwards due to the increase of the *Normal Force*.





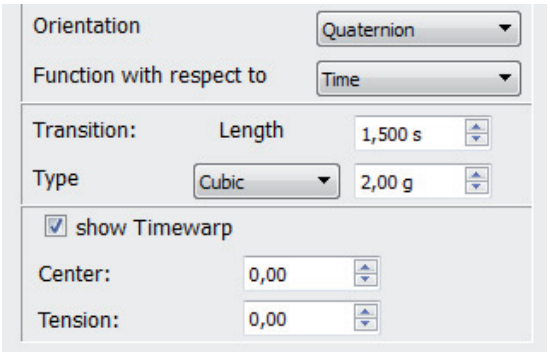
After adding the *Roll Change* graph, the track also tilts to the side to remain 0g lateral.

This leads to the conclusion that using *Force Sections* needs a different design approach than section types with independent components. Although there might be tutorials available that help on certain elements, each element needs to be built on its own as the input parameters will be different when incoporating the element in a coaster layout.

4 Advanced Features

4.1 Timewarping

The timewarping panel can be found in the transition panel by clicking on the ckeckbox *show Timewarp* at the bottom. Two number fields will appear:



Timewarping will provide more control over the selected transition which helps to improve the shaping, getting the track into the desired position or using less transitions for certain tasks.

Positive values entered in *Center* will make the transition's center shift to the right, negative values will do the contrary. Extreme values (>5) will flatten out the opposite side.

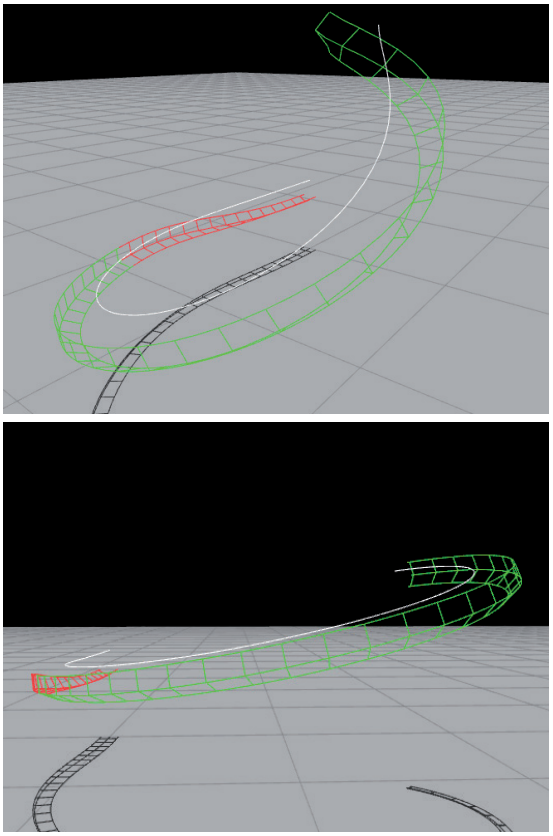
Tension values ranging from 0.01 to about 1.5 will make the transition more gradual, higher values (up to 10) will make the middle of the transition be more horizontal. Negative values up to 10 will make the transition more abrupt in the middle part and more flatten out on the endings.

In most cases, values ranging from -2 to 2 can be used without any problems, whereas more extreme values might cause unwanted bumps in the track, but there are exceptions.

4.2 Orientation

One of the most important settings can be found at the top of the Transition panel (see picture above) which is called *Orientation*. In a nutshell, this setting which can be switched between *Quaternion* and *Euler* controls in which way the program creates the final track banking based on the *Roll Change* graph. Taking the example of a basic helix will reveal the underlying principles. Especially the first example is not exactly how the helix would be built in practise, even when using the *Euler Orientation*.

The first attempt of building the helix is by using the *Euler Orientation*. In this case, the roll values of the track are always measured relative to the ground plane and as the *Roll Change* rate stays 0 °/s for the rest of the helix, the roll of the beginning of the element has to stay constant. As a consequence of this fact, the train slowing down on its ascend and the constant 3g Normal Force, the track starts to screw itself upwards with a rapidly decreasing radius. In the *Euler Orientation*, this is the path a train has to follow in order to



have constantly 65° roll relative to the ground plane. A possible solution to this would be to avoid slower speeds or manual *Roll Change* corrections..

Another issue with *Euler* is a vertical (pitch=90°) piece of track as it can't be calculated because the roll angles can't be defined properly in this case.

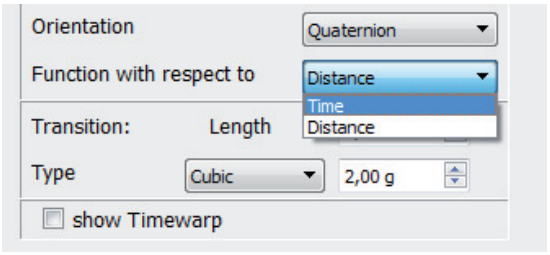
By using the *Quaternion* orientation the picture is fairly different. This time, the helix doesn't form a spiral. Instead, it stays perfectly in a slanted plane which is defined by the constant roll value of 65°. *Quaternion* does not use the global coordinate system as a reference for their roll values, it defines the roll based on the coordinate system of the track itself, which is tilted by 65° at the beginning of the helix part. But the roll reference is not fixed to this point, it will change whenever the roll of the track has to be altered due to the transition graphs. A consequence of this principle is that the (Euler-based) roll of the helix will get higher in the ascending part, being 85° at the end of the example element. This is simply due to the slanted plane the roll is based on; in the perspective of the said plane, the roll stays perfectly constant.

In conclusion, the main difference between the two orientations is the reference for the roll values which lies either on the ground plane (*Euler*) or on the track itself (*Quaternion*).

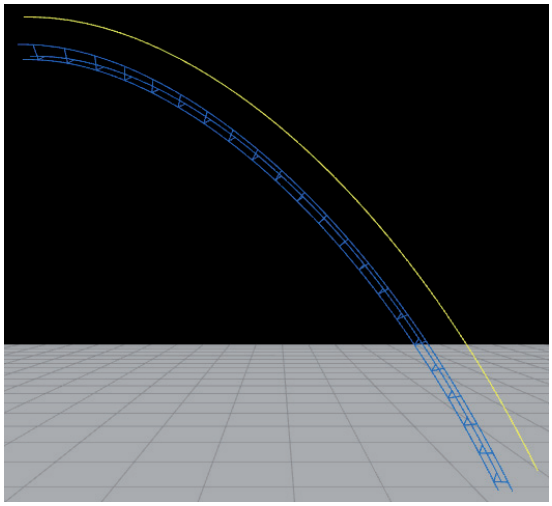
As a side note: the statistics panel uses *Euler* for the display of the roll value, whereas the *Roll Change* is always calculated using *Quaternion Orientation*.

4.3 Function Argument

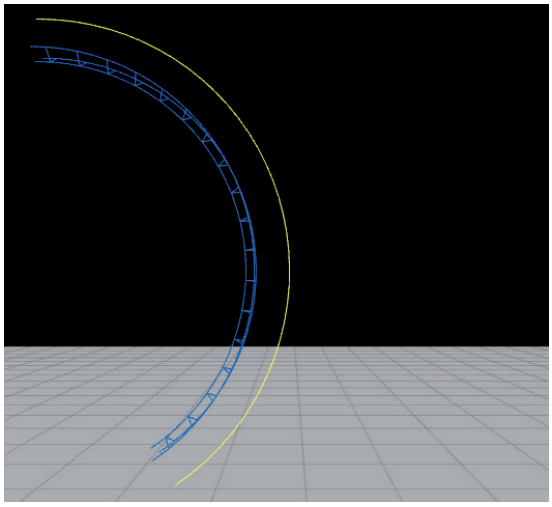
In the last pages it was already in use and finally, the whole sense of this additional setting will be made clear in this chapter.



In FVD++, each section can be set to have their component functions based on *Time* or *Distance*. This option is called the *Function Argument*. It defines on which physical dimension the program's calculations will be based on in the current section. Each setting has its own advantages.



This drop uses *Time* as the *Function Argument*. Over the whole element, the *Pitch Change* stays at constantly -21 °/s. As showed in the picture on the right, the track gets more and more streched when further going down. This effect is due to the acceleration of the trains, therefore one second of ride time needs a longer piece of track which has to bend by 21° every second. The result is the track becoming more flatten out as the speed gets higher. This effect creates bumps at slower speeds as the track flexion will have even more extreme variations.



Using the *Pitch Change* based on *Distance*, the result resembles a perfect circle instead. In this case, the rising speed has no effect on the track curvature at all. With 3 m/s starting velocity, -21 °/s will convert into -7 °/m *Pitch Change*.

Often a good strategy to avoid unwanted effects is to use both techniques together. For a first drop, the best way would be to start with a distance-based apex bend using a fixed speed, followed by a distance-based bend with calculated speed which is followed again by a time-based bend and after that, a *Force Section* to create the valley.

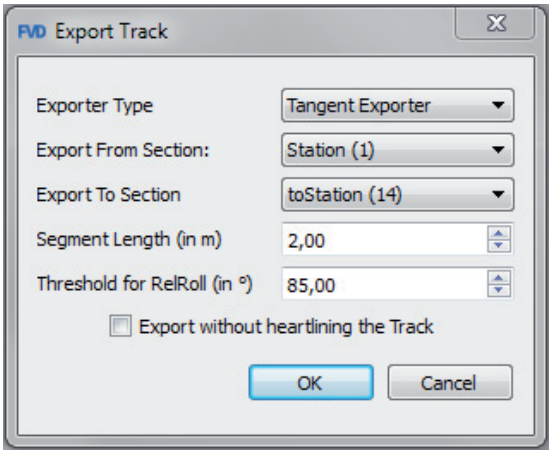
The same principles apply to all other component types as well.

5 Exporting

5.1 The Export window

When the track should finally be imported into NoLimits, it needs to be exported out of FVD++ in order to create an *.nlelem file which can be inserted in the NoLimits Editor. After clicking on *Export* (Ctrl+Shift+E) in the *File* menu, a new window will appear to specify the desired exporting configuration:

In this window, different exporting settings can be defined and by clicking on *OK*, a file manager window will open to select the directory the generated NoLimits element will be saved to. After having done the first exporting action, FVD++ will keep the settings so each time *Export* gets clicked or Ctrl+Shift+E pressed, the export will be done exactly the same way without asking about the settings. To specifiy a new exporting configuration, a click on *Export As* (Ctrl+E) will open the *Export* window again.



5.2 Export settings

The *Exporter Type* can be choosen between the *Tangent Exporter* (preset) and the *Spline Exporter*. The latter tries to create a bezier curve that follows the different sampling points calculated by FVD++ which might cause problems when exporting different sections individually. The endings of each exported section won't match as each of them only had to follow its own curve. For instance, exporting a *Curved Section* will result in a NoLimits element which has a effectively smaller *Total Angle*. In contrast to this, the *Tangent Exporter* will keep the tangents of each exported vertex point exactly how they were indended to be, therefore each angle will be perfectly matched up with the track model inside FVD++.

Export from Section and *Export To Section* defines the start and ending points, allowing to select only a part of the coaster to be exported. Next to the section names their numbers is displayed as well.

The *Segment Length* will give the exporter engine the length each exported NoLimits segment has to have. Values can be set from 0.2 to 10m but the lowest value in use should not be smaller than about 0.5m because otherwise it might create an unsmooth NoLimits track.

Threshold for RelRoll defines at which pitch angle of the track the vertices inside NoLimits should be set to relative rolling instead of normal rolling.

The clickbox *Export without heartlining the Track* will export a bezier curve that represents the heartline path (including roll) itself instead of the actual track witch rolls around the heartline.

written by Lucas van den Bosch (privat@lucasbosch.de)

FVD++ development by Stephan Alt (alt.stephan@web.de)

Further helping resources:

FVD++ Beta: Discussion Thread (v0.31):

http://forum.nolimits-exchange.com/comments.php?DiscussionID=2646

Lenny's Engineering Thread (about the early development):

http://forum.nolimits-exchange.com/comments.php?DiscussionID=2546

FVD++ Help and Tutorial Thread:

http://forum.nolimits-exchange.com/comments.php?DiscussionID=2712