

GP2 Scenery Editing Tutorial v2.0



This is version 2.0, not indicating that the tutorial has made a big step forward (it has not) but it seemed like a nice number to end my involvement in the scenery tutorial. That means, that unless some major breakthrough is discovered, you are looking at the final version of this document. If you don't like that, or want to notify me of something concerning this tutorial, contact me at m.p.j.keizer@sepa.tudelft.nl

Index

[Introduction](#)

[1\) Scenery Basics](#)

[2\) Commands Overview](#)

[3\) Guideline](#)

[4\) Basic Structures](#)

[5\) addie's special-request section](#)

[Appendix](#)

Introduction

By Martijn Keizer, m.p.j.keizer@sepa.tudelft.nl

In this document, I will try to share the knowledge I accumulated over the last months about the editing of GP2-scenery structures.

Therefore I would like to start with some theoretical basics you have to know about when trying to understand what I try to make clear. Then, an overview of the commands used while editing scenery, from Addie's Command-library, supplemented by some practical tips concerning the use of these commands. This is followed by a step- by -step guideline how to build the scenery. In chapter 4 I made some drawings of basic structures and formations as examples of scenery-editing. The appendix contains some things I couldn't place elsewhere, like a FAQ and version updates. But first:

Credits

Martijn's Scenery Tutorial

A lot of thanks go to Paul Hoad, for the editor and lots of support. When I heard there was gonna be a Trackeditor, I started making drawings of circuits etc, and finally I decided that it was time to replace the 486 anyhow. This editor has had a lot to do with that! So I guess the registration fee was only a very marginal addition to the costs of making tracks.

Also credits to addie, for finding the basics about scenery editing (amongst other subjects as pilanes etc.), and taking the time to write it down in the command-library, pitlane-tutorial and GP2-trackediting tutorial. Without him, command 0xB8 would still be unknown!

I would also like to thank everyone who found out things for GP2, it becomes even more clear you can't do it all yourself. The funny thing is, you don't have to! So I use sometimes pieces of information that are not originally my own finding, but rather combinations of different sources and quick notes by someone, that I tried to take together in a comprehensive and easy-to-read form. So when you think your name should be explicitly called in this tutorial, let me know, and I'll add you to the following list of contributors!

John Verheijen, Swervin' Irvin', Snqqpy.dog, Glenn Durden, Michael Hompus, Babel, Bob Culver, Jason Sinnbeck,

And of course to all trackeditors that keep making even-better-looking tracks. On one hand I really hate that, because it costs so much more time to make a decent track relative to others! But I really hope this tutorial helps all trackeditors to enhance their tracks to a next "generation".

Tracks:

In this tutorial I will sometimes point to a track, where there can be found a good example of the effects I wanted to describe. These tracks are mainly the original tracks of Monaco and Spa-Francorchamps, together with two tracks I build myself as an exercise and experiment for using scenery. These tracks are StuwLake Track, and Tunneltrack. These two tracks are available for download at GP2.com and at Sjons GP2 site. You may wish to take a look at these tracks, in order to get some idea of the possibilities of scenery.

Chapter 1

Scenery Basics

In this chapter I will guide you through the different sets of coordinates GP2 uses, and the basic structure of any scenery-piece. Therefore, first a description of the "ribbon"-concept will be given, then some beautiful drawings by Addie to explain the coordination system along with some notes.

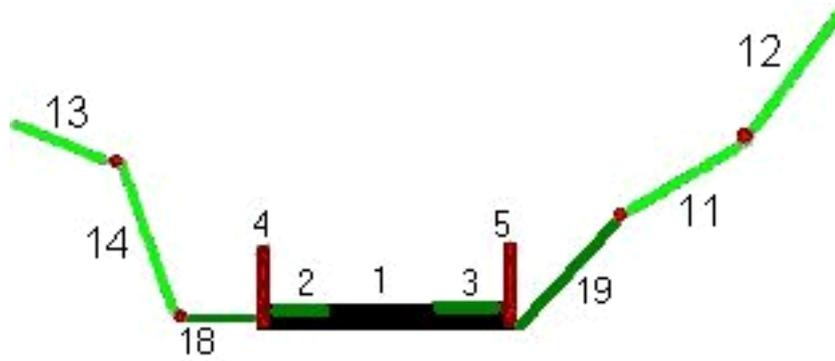
1.1 Ribbons

Have you ever build a mountain for your little model-train? One way to do this is to build a framework of iron-wire, and then glue some heavy paper over the framework. Well, in GP2 the scenery is build in a similar way. The Scenery1 (0xB8) command defines the basic framework. It consists of four places where the paper can be attached to. These arms are always in the plane perpendicular to the axis of the track. Now these strips of paper are called "RIBBONS" in GP2-terminology.

1.2 Banks

The space between the fences and the lower ribbons is also filled with a texture. We will call these textures "BANKS" in GP2. If you press alt-D two times in the game, you will see that all "ribbons" disappear, but the "Banks" stay. This way, you can clearly see what is "bank" and what is "ribbon".

Martijn's Scenery Tutorial



1-road
2/3 verges
4/5 Fences
18/19 l/r bank
11 ribbon1
12 ribbon2
13 - ribbon3
14 - ribbon4

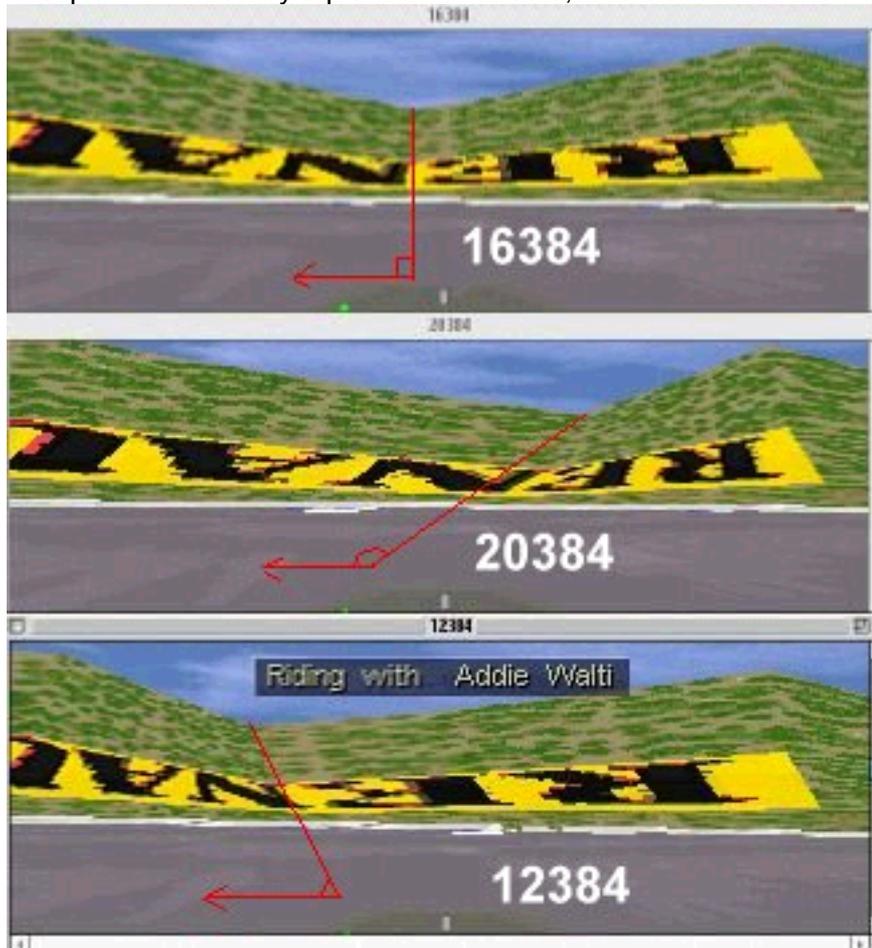
Martijn
1998

1.3 Coordination System

The precise places of where the ribbons are, is defined by the Scenery1 (0xB8) command. The arguments of this command are the coordinations of the connection places relative to the local tracksection. Thus, every tracksection has its own relative system, and a ribbon at a height of 1000 in two consecutive tracksections will be parallel to the track, regardless of the height-change of the actual tracksection.

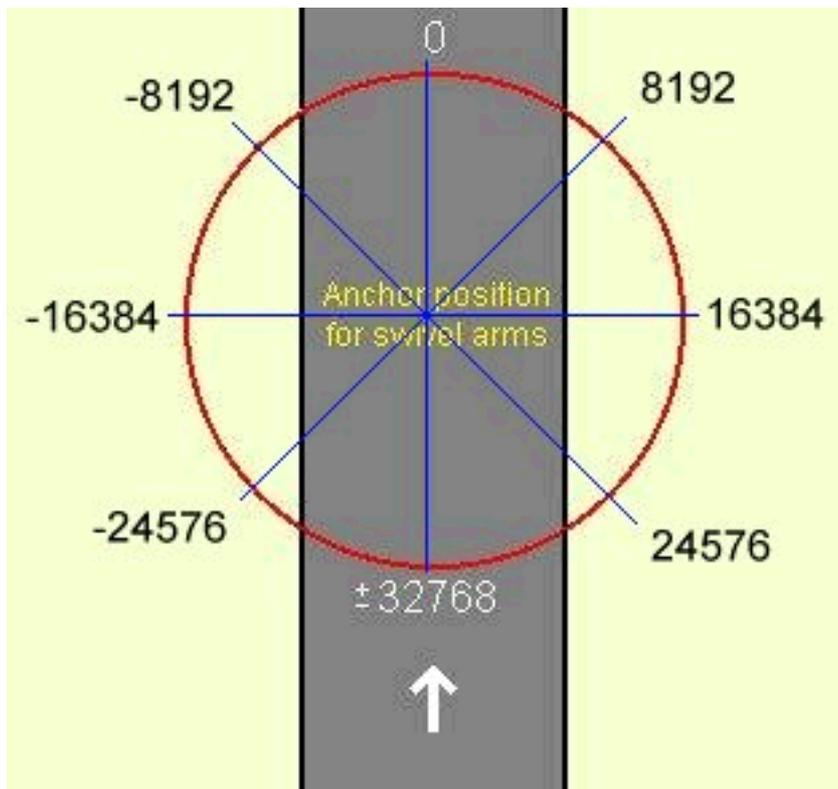
1.4 Swivel Arms

The "arms" that define the positions of the ribbons along the track, do not have to be perpendicular to the track. The 0xAF command defines the angle that these arms make relative to the track. How better to explain that than by a picture from Addie, who discovered the true meaning of 0xAF:



All three screenshots feature exactly the same clip of a certain part of a track. There are three "arms" visible. One at the left side, one at the right side and one in the middle. The only difference between the three images is the angle value of the middle "arm".

You can determine where the arms will end up by looking at this compass. You see a full 360-degree turn is from 0 - 65535, or, if you like, from -32767 to 32768. Using non-perpendicular swivel-arms can be useful if you want to make sure that arms of scenery-commands of different tracksections do not overlap one-another, for instance in a sweeping section or a tight hairpin.



Unfortunately, the swivel-arms do not (yet) show up correctly in the trackeditor. They are all drawn as if perpendicular to the track. So you'll have to use your imagination for now. Alternately, you can use the trackeditor of Iso-Hannula to view the correct location of the ribbons.

1.5 Number of ribbons to a side

Normally in a standard situation, there are two ribbons to the right and two to the left. It is however possible to change this situation. In some cases it is useful to have three ribbons to one side, or even four. Because the number of total ribbons is maximized to four, a ribbon added to the right is subtracted from the left side. The a2 of the Scenery1 (0xB8) command determines the "number of ribbons to the right".

Note that if you change the Scenery1 (0xB8) a2 value in the middle of your track, that a ribbon has to cross the track. That gives sometimes an ugly scenery flash. Therefore you have to switch the ribbons off temporarily. How to do that will be explained later.

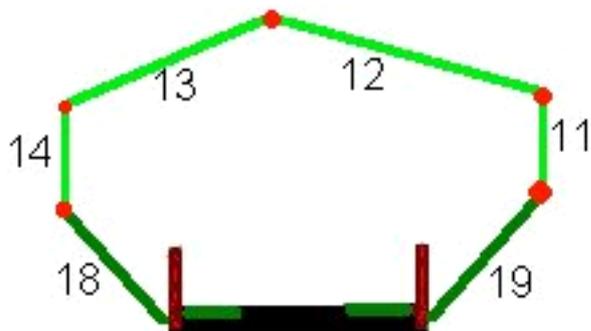
In this document I normally presume, for most cases, that the ribbons are equally distributed. Otherwise it will be explicitly announced.

1.6 Naming the ribbons

A ribbon is totally useless without a texture applied to it. It is important to know where ribbons are if you want to select textures for the ribbons. So I have taken the terms used in the 0xBC command as a guideline to name the ribbons.

Because the ribbons can show up on any side, I will try to explain the numbering with the following drawing.

Martijn's Scenery Tutorial



11 - ribbon1
12- ribbon2
13- ribbon3
14- ribbon4
18 - bank left
19 - bank right

Martijn
1998

Imagine a tunnel. You see that the ribbons are numbered consecutive. Note that it is NOT necessary to know in this case how many ribbons come from the right, and how many from the left. So look at a2, number of ribbons to the right, as a determination of WHERE these ribbons are cut into a number to the left and a number to the right. If you remember this picture, you will also be able to see easily why you have to rotate the texture on ribbons to the left, and not on the right.

So from right-down to left-down we find the texture-location values: "19,11,12,13,14,18".

If there's a "cut" in the middle ("2") then the right side constitutes of (from bottom) rightbank (19), ribbon1 (11) and ribbon2 (12). The left side constitutes of (from bottom) leftbank (18), ribbon4 (14), ribbon3 (13).

In the case there's a value (Scenery1 (0xB8) a2=3), then the rightside consists of rightbank (19); ribbon1 (11); ribbon2 (12); ribbon3 (13). The leftside is (f.b.) leftbank (18); ribbon4 (14);

I expect you will see the logic in this, and work the other situations out yourself.

1.7 Pitlane Scenery

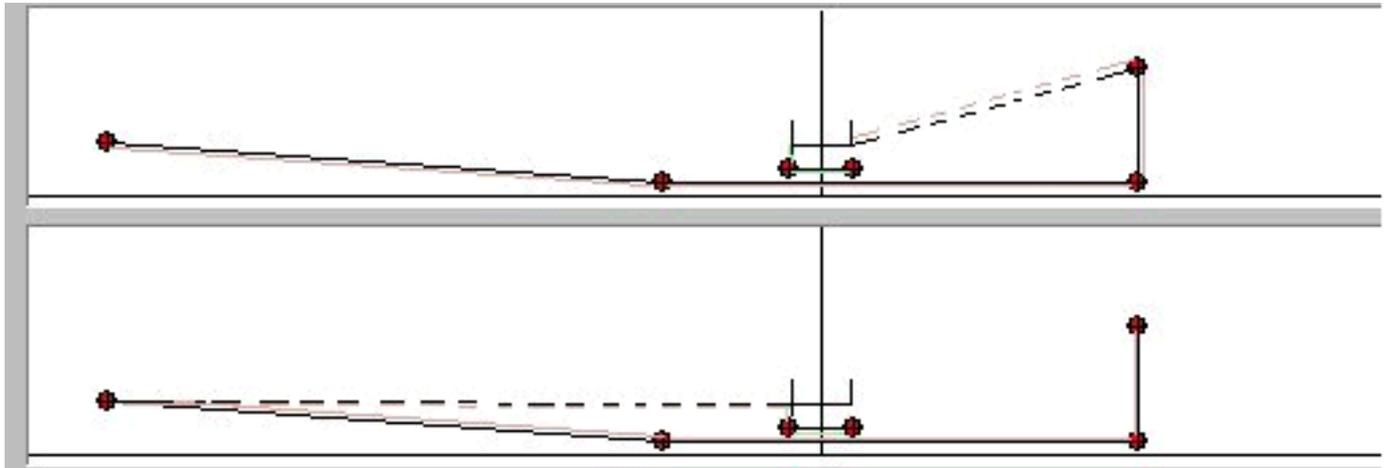
As you may have noticed in browsing through some tracks, all tracks have Pitlane-scenery commands. However, they are in most tracks disabled and thus not visible in the game. Ofcourse there is a exception to this rule, it being Monaco. The pitbuilding there is constructed from a ribbon, with a garage-texture applied to it. As you probably have NOT noticed, these pit-ribbons are INvisible when you are driving on the main road. Therefore, the usefulness of this pitscenery is reduced to some very special cases. I can think of, for instance, a way to make those pitwall-tents where Ron Dennis and Jean Todt watch the race. You know, the kind of tents that Jos Verstappen likes to smash [France 1994]. You may think of other uses yourself, if you can...

If you want to know, the pitlane-scenery is enabled by placing a 0xD9 command with a2=63, at the first pit-scenery command.

You could however also use the main scenery-ribbons to make a pitbuilding that is visible from the pitlane as well as from the main straight. Therefore, you must enable the ribbons near the start/finish-area, and make the ribbons start at the edge of the pitlane-road. You can (for instance) make pitboxes that seem to be carved out of a high rock. An example of the use of pitlane scenery can be found on the "StuwLake" track.

1.8 Two sides to all Ribbons

Note that a ribbon has two sides: a front and a back site. Looking onto the front, you see a texture. Looking onto the back of a ribbon, it is completely transparent. Bear that in mind when trying to develop complex scenery structures, as sometimes the ribbons are bended in such a way that they become totally invisible. Imagine a tunnel, like in section 1.6. Now, driving from the track, you can see all ribbons. But when you have a camera outside of the tube, you won't see any ribbons at all.



Here you see two different solutions to the same problem. The aim is to build a bridge. Both setups are with three ribbons to the right. On top you see that the visible side of the ribbons (marked in a light pink here) is faced down on the side where it matters, the water and the trees to the right. That means that you won't see any of this scenery; it's all transparent. On the bottom-drawing, you see that basically the structure is the same, but that the direction in which the ribbons to the right are defined, is flipped. All of a sudden, the visible side of the ribbons is faced up, and thus you can now see the water and the trees. Note that the right bank is to be removed here, and is therefore dotted.

Chapter 2

Commands-Overview

In this chapter I will give an overview of some commands that are used during Scenery-editing. We try to get these descriptions identical in the commands-tutorial and this document. I refer sometimes to a "location code", see Chapter 2.2 for the types and the description.

2.1 Scenery Commands

These descriptions are directly stolen from Addie's command library (v2.7), with some permission.

0xaf (175) Pair Of Swivel Arms, 3 args

This cmd defines position and angle of a pair of swivel-arms for the scenery-ribbons, one for the left side and one for the right side of the track. The cmd 0xaf (175) is always followed by a cmd 0xb8 (184) that defines the x- and z- coordinates where the scenery-ribbons are attached to the swivel arms.

a1: Offset Into Sector (for the 0xaf/0xb8 pair)

a2: Angle Of Left Arm; (useful: negative value); -16384 (90° to the left)

Martijn's Scenery Tutorial

a3: Angle Of Right Arm; (useful: positive value); e.g. 16384 18432 20480 24576

In a single track-sector there can also be several pairs of 0xaf/0xb8 them with different offsets in cmd 0xaf(175).

See paragraph 1.4 for an example.

In the top image the value is 16384 (its a right arm; left arm would be -16384). The value in the middle image is 20384. The value in the bottom image is 12384.

for more details on the angle value, see compass chapter in glossary.

There are also the cmds 0xc0 (192) and 0xc1 (193). They act basically as cmd 0xaf (175) but they define just a single swivel arm. see following sketch:

0xb0 (176) Turn Ribbon Off, 2 arg

This command is to switch off a ribbons and/or banks

a1: Offset Into Sector

a2: Location Code Type B

Default: Ribbons/Banks off

For switching them on, use 0xb9 or 0xd9.

Do ONLY insert this cmd at the same position as a 0xaf/0xb8, 0xc0/0xb8 or 0xc1/0xb8 pair. (see original tracks for reference). NEVER have it alone. If you do, gfx bugs of the strange kind may happen.

When you see it, you will recognise it. Its a floating line starting somewhere around the border of the iamge leading into the "helmet of the driver".

0xb8 (184) Scenery Structure, 14 args

This is the famous ribbon-cmd. It defines the basic scenery, together with the swivel-arm-cmds 0xaf (175), 0xc0 (192) or 0xc1(193). There are a lot of additional scenery cmds, but 0xb8 is the basic scenery cmd. As mentioned you never see the cmd 0xb8 standing alone, there always is another cmd BEFORE it, a cmd 0xaf(175), a cmd 0xc0 or cmd 0xc1. Very often cmd 0xb8 is also followed by support cmds, e.g. 0xb0, 0xba etc.

Before going on you may first want to have a look at the cmd 0xaf(175), 0xc0(192) and 0xc1(193). Because these cmds do define the offset into the sector and number and angle of the swivel-arms.

If number (1 or 2) and angle of the swivel-arms is defined, you can set up the structure of the swivel arms with 0xb8 (184).

Arguments:

a1: unk; ?always 0

a2: values 0..4 basically it gives the location of the track within the 6 ribbons. You can also refer to it as the number of ribbons to the right of the track

a3 .. a14: coordinates of ribbons, as can be seen in the sketches

Origin of x- and z-coordinate is the middle of the track.

When double-clicking on a cmd 0xb8 in the track tree, the scenery editor opens and you see a cross section of the choosen scenery structure cmd. There you can set the several arguments and immediately see what happens. If you are new to it, you may want to check out a2 "ribbons to the right" at first.

PH: Scenery is determined as a series of RIBBONS that run along the side of the track these ribbons are defined by the scenery commands along the tracks, all this

Martijn's Scenery Tutorial

defines the actual shape of the scenery, you then have to map a texture to the scenery in the same way as you map a fence or gravel trap using the (0xbc commands) by changing the location code in the 0xbc commands from left/right verge to fence or ribbon you can change the look of the scenery

it seems that the 0xc8 commands at track section 0 also change the default scenery used for the ribbons.

You should think of the ribbons as being continuations of the grass verge only the height about the road can be defined as can the distance from the road hence giving the ribbon a low x(horizontal) and a high z (vertical is going to make a vertical wall (outside the track fence, hence monaco) now map a brick texture to it and you have a real wall. map a set of trees to it and you have a forest a la monaco or hockenheim, map grass to it and you get a grassy hill ala Brazil.... getting the picture yet!

You should think of the ribbons there are six of them (not always 3 each side (this is defined by a2) as providing you with extra surfaces simply to map textures.

[actually there are 4 of them; the very first ribbon to the right and the very first to the left of the track is called "bank" in GP2]

addie: the maximum number of cmds 0xb8 seem to be 128. if you insert another one, the very last of them in the track gets lost. MK experienced even crashes here.

0xb9 (185) Turn On Ribbons, 2 args

MK: "With this cmd you can "turn on" individual ribbons and banks depending on the value in argument a2. The ribbons are showing up no matter what detail level is set in the game (in opposite to 0xd9)"

a1: Offset Into Sector

a2: Location Code Type B

Do ONLY insert this cmd at the same position as a 0xaf/0xb8, 0xc0/0xb8 or 0xc1/0xb8 pair. (see original tracks for reference). See 0xb0 for an example of a gfx-bug that can happen when not following this rule.

To switch them off again, use 0xb0

Default: Ribbons/Banks off

0xba (186) Bridge Scenery, 3 args

a1: ?Offset Into Sector; mostly 0; in f1ct04 there is a value=2

a2: Location Code Type B; typical values : 0 1 2 3 4 5 6 8 10 15 (40 f1ct09)

a3: ?Location Code Type B; typical: 0 1 2 3 4 5 6 8 10 15 16 (20 30 80 f1ct09)

most of the time a2 is equal a3; exeptions e.g. in f1ct03 f1ct05

MB: take a look at the 0xba cmd. it does about the same as connect [bridge] fences...

MK: This command is used to force the scenery to "shortcut" the track. When scenery is made in a curved tracksection, normally the scenery follows the curvature of the track. When this command is used, the scenery makes a shortcut, thus following a STRAIGHT line between two scenery-connection points (0xB8 cmds).

Note that the scenery5 command is a SWITCH that can be set for some part of the track, and some ribbons using the location code.

The a2 value (and/or a3 value) indicates to which ribbon- locations the command is applied, you can thus force individual ribbons to shortcut.

Martijn's Scenery Tutorial

Normally, you use this only on the INside of a corner.

See also section 3.5 [in scenery tutorial]

0xc0 (192) Single Swivel Arm Left, 2 args

0xc1 (193) Single Swivel Arm Right, 2 args

These cmds define position and angle of a single swivel-arm for the scenery-ribbons. The x- and z-coordinates of the scenery-ribbons are defined in the cmd 0xb8 (184).

a1: Offset Into Sector

a2: Angle Of Arm; (0xc0: useful: negative value; 0xc1: useful: positive value)

For extensive description of the meaning of swivel arms, see cmd 0xaf (175).

0xcd (205) Adjust Shadow, 4 args

Lights up or darkens the textures.

a1: Offset Into Sector

a2: Location Code Type A

a3: Length; (of the darkend or light up zone)

a4: Light Up Or Darken Factor; MK: seems to be between 0 (dark) and 15 (bright).

Example in Monace t22-t24

In this example the cmd 0xcd (205) is used to lighten up the frontwindows of the massenet-hotel. Or in technically terms: to lighten up the texture on ribbon 4.

0xd9 (217) Turn Ribbons On, 2 args

MK: With this cmd you can "turn on" individual ribbons and banks depending on the value in argument a2. The banks are showing up no matter what detail level is set in the game. The ribbons only show up in medium and high detail level. If you want them showing up no matter what detail level is set, you have to use 0xb9 instead)

a1: Offset Into Sector

a2: Location Code Type B

Do ONLY insert this cmd at the same position as a 0xaf/0xb8, 0xc0/0xb8 or 0xc1/0xb8 pair. (see original tracks for reference). See 0xb0 for an example of a gfx-bug that can happen when not following this rule.

To switch them off again, use 0xb0

Default: Ribbons/Banks off

2.2 Texture Location Codes

Often, you will find a "texture location code" in a scenery command. This means, that you should see the value as a digital checkbox, where the bits (and thus value) stand for the exact places it's applied to.

Type <A> Location Codes

used for scenery7, 0xD0?, 0xC7?

Martijn's Scenery Tutorial

Type location code
used for Scenery4, Scenery2
Type <0xBC>
used for 0xBC, 0xC8

name of place	Type <A>	Type 	<0xBC> value		on/off (example)	digital	value
Fence left	1	*	5		off	0	0
Fence right	2	*	4		on	1	2
?kerb left?	?4	*	?17			0	
?kerb right?	?8	*	?16			0	
Verge left	16	*	9		off	0	0
Verge right	32	*	8		on	1	32
Bank Left	64	16	18		off	0	0
Bank Right	128	32	19		on	1	128
Ribbon 1	256	1	11		on	1	256
Ribbon 2	512	2	12		on	1	512
Ribbon 3	1024	4	13		off	0	0
Ribbon 4	2048	8	14		off	0	0
?	4096	*	?			0	
?	8192	*	?			0	
?	16384	*	?			0	
All places	32767	63	N/A		total	001110100010	930

In the right part of the table, you'll see an example of the calculation of an situation, where the goal is to change the lightlevel of the entire right side of the track. Then, you'll have to insert a Scenery7 (0xCD) command, which should be applied to the right fence, the right verge, the right bank, and the right ribbons 11 and 12 (in case of [0xB8;a2=2]). So you add the type <A> values of these places, and add thus $2+32+128+256+512 = 930$. This value is placed in the a2 of (0xCD).

Chapter 3

Guideline to Using the Commands

In this section you may find the missing connection between the easy-to understand, fresh-looking and inviting chapter 1 and the dry, boring, dusty and abstract chapter 2.

3.1 Quick-and-dirty Scenery-setup

This is a quick description how to enable ribbons along your circuit and make sure that they stay there. After you have done this, there will be ribbons for the WHOLE length of the track. If you want to switch them off and later on again, you will have to do some things that are way more complex. But

Martijn's Scenery Tutorial

to switch them off and later on again, you will have to do some things that are way more complex. But for the time, this will do.

- Make sure that every Scenery1 (0xB8) command has an accompanying Scenery3 (0xAF) command.
- Remove ALL 0xB0; 0xB9; 0xBA; 0xC0, 0xC1, 0xD9 commands.
- Insert a Scenery4 (0xD9) command in the first tracksection with a2=63.

With these simple actions you are able to make simple scenery-tracks like the Tunneltrack I built.

Now, you can build some basic structures by adjusting the Scenery1 (0xB8) values. The way to make it look like a specific thing, like trees, is to map a texture on the ribbons using the 0xBC command. But that's a story that will be told elsewhere...

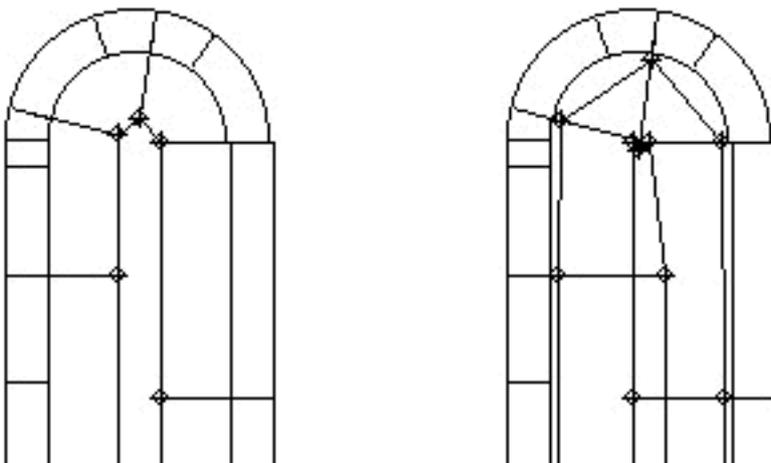
3.2 Normal Scenery Structure

Normally, the scenery of a track consists of one 0xAF command, followed by a 0xB8 command. These two commands form a PAIR, and the order in which these two appear is fixed, and should be maintained. First comes the 0xAF, and then the 0xB8. Note that the first value of the 0xAF is an offset into sector, and this value defines where this PAIR of commands will appear on the track.

You should look at this pair as being practically one command, split in two parts. They should always remain together, else you will get errors. Note that you can also form pairs 0xC0 - 0xB8 and pairs 0xC1 - 0xB8. These pairs define the scenery on only one side of the track. More on that in section 3.5.

3.3 Scenery in the inside of corners

The problem with scenery in corners, is that the scenery can be placed so far from the track *before* the corner, that it is visible *after* the corner. This is caused by a leak-through. Problems like these occur mostly in 130-180 degree, tight turns, such as hairpins. Unfortunately, the Trackeditor doesn't yet show the proper locations of the ribbons, they are much further from the track in (virtual) reality. The next drawing shows what I mean. Note that the track-pies give a clear indication of the maximum distance you should let your scenery drift away from the track.



In the left drawing, you see that the ribbons on the way towards the hairpin, do not overlap with the ribbons coming from the hairpin. On the right drawing, you can imagine that in mid-corner, an external camera doesn't know what to do anymore, and this becomes clearly visible as some mixed-up scenery parts. Therefore, you should avoid using scenery like in the right drawing, unless you know very well what you're doing.

In the TE 1.6.7 there is a good representation of the distance a ribbon has to the track. So this will certainly point you to situations where the ribbons make the error described here.

3.4 Scenery on the outside of corners

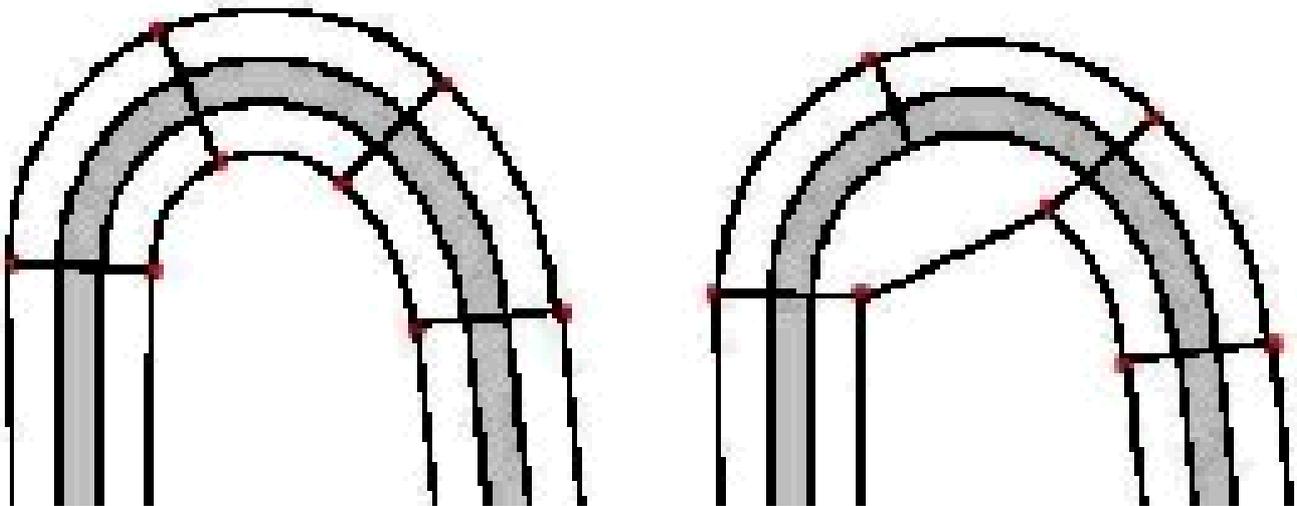
On the outside of corners, scenery is usually not that much of a problem. Some things to note, though. First of all, ribbons on the outside of a corner are much longer than on a straight section. That means, that the textures applied to ribbons in a corner, appear very stretched. Therefore, you should always define textures for straight sections and curves separately. This problem is best visible with trees, on straights you can have one texture per 4 length-units, while in the corner you need twice as much to get the same type of trees.

Secondly, a graveltrap on the outside of a corner means that the verges are a lot further from the track than elsewhere. That means, you also have to move your scenery backwards, as otherwise, you will have some scenery overhead when you accidentally drive through the graveltrap.

3.5 Scenery Shortcutting using Scenery5 and 0xC0/0xC1

By default, the scenery ribbons follow the corners in the track. You can see that clearly on, for example, the Hockenheim track where the grandstands have the same curvature as the track. Now sometimes, you want to let the scenery follow a straight line from one point to the other. This can be done by adding a Scenery5 command. You can see a close analogy between adding a scenery5 command in a certain tracksection, and clicking the "bridge fence" checkbox. One counts for the ribbons, the other one for the fences. Note that you have to specify in the Scen5 cmd to which ribbons this effect should be applied.

Another way to create a similar (yet slightly different) effect is by using an 0xC0 or 0xC1 command, instead of the 0xAF command. By doing that, you refrain from defining the scenery on both sides of the track, and choose to define only one. I will make a drawing to clarify this:



I would like to emphasize I drew the ribbons curved, like they appear in the game, but NOT like they appear in the Trackeditor.

On the left, you see a normal defined scenery, with 0xAFs and 0xB8s. In the middle, you see the real situation when one of the 0xAFs is replaced with a 0xC0. This command defines only the left side, making the game connect the rightside ribbons without taking this 0xC0 and its accompanying 0xB8 into account.

On the right side, you see that the first curved section has a scenery5, 0xBA command. That causes the ribbons to go straight WITHIN that particular tracksection.

Note that the scenery5 command is a SWITCH that can be set for some part of the track, and some ribbons using the location code. There is also another command that influences this setting in some way, I believe 0xC7 or 0xC6. You are warned!

The work on Circuit Ultima set me on this. In the Casino corner, there is a building to the right, made of ribbons. You should examine this building closely, it's extremely complex, and cost me a few hours

Martijn's Scenery Tutorial

of ribbons. You should examine this building closely, it's extremely complex, and cost me a few hours to understand.

In t25, the second scen1 (offset 1) defines the right side of the facade of the building. After that, there follow in t27 three scenery-lefts. Then in t29, there is a scenery-left (0xC0) , followed by a 0xAF. At this moment, 4 Ts later and over 4 scenery-lefts, the building-definition continues. In the meantime we have rounded a 120 degree righthander. Yet, the building is flat. That's quite logical, as I can't make a guess WHICH curvature should be taken, if you have crossed four tracksections. That let me to the conclusion that it would make more sense if it was a straight line from one definition to the other.

3.6 Multiple Scenery Commands in one tracksection

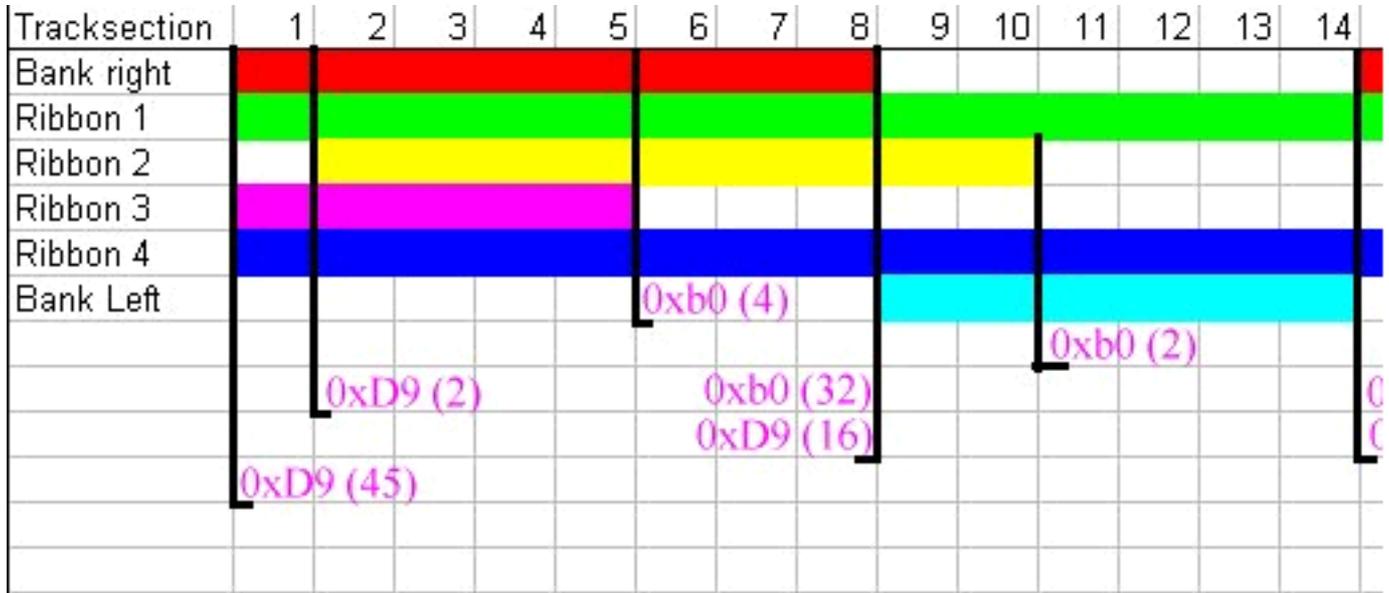
Note that this is not well supported in the TE. The TE only shows the first scen1 cmd of a section, and the following ones not. This will be amended (and partially already is), in newer verions of the TE.

In the meantime, from TE1.6.9 or so, the multiple-scenery cmds is very well implemented in the trackeditor, and this should no longer lead to any problems. It is just good to know that it is possible to use more scen1's in one TS. They all must have a 0xAF or 0xC0/1 before them ofcourse, otherwise things will get mixed up. The offset that you put in the 0xAF/0xC0/1 is used by the 0xB8 to define where it's located. This particular thing is not yet implemented in the TE, and all scenery definitions appear to have taken place at the start of the TS.

3.7 Switching Ribbons On and Off

Just defining the basic structures is not enough to see nice scenery. The ribbons can be switched on and off using different commands. This way, you can generate different effects.

The main way to switch ribbons on is using the Scenery4 (0xD9) and Scenery 2 (0xB9) commands. The way to switch an individual ribbon off, is using the 0xB0 command. The values in the arguments of these commands indicate a texture location, of type .



In this picture you see how it works. The bars indicate whether a ribbons should be switched on or off at a particular tracksection. You should have a picture like this in your head, when you're making the scenery. Now, you see on this picture where you have to insert the commands in order to get the commands switched on and off. Between brackets, I have placed the appropriate a2 value of the particular command. On the right you see the type location codes, directly from the [location-table](#) in 2.2. Note that every command has to be inserted at the first tracksection in which it should work. For example, the first command should be inserted in t1, and the last one in t19 (not drawn in the picture)

Martijn's Scenery Tutorial

Paul Hoad has made a drawing in the TE just like this, so you can monitor where your ribbons go in case you miss one. It is accesible by clicking a 0xB8 in the pitlane, or on the track.

When you think of switching a ribbon to another side, you have to switch a ribbon off, and on again a little later. Then, you have an 0xb0 for that particular ribbon, and an 0xD9 again in (for instance) the following tracksection.

Switching a ribbon to another side therefore usually implies to switch a ribbon off, and on again. To know which ribbon you have to switch off can be figured out, but as an extra service I put a table here. Note that the back-switching implies the temporarily switch-off of the same ribbon as the switch-forward.

ribbons to right	0	1	2	3	4
0	X	1	1,2	1,2,3	1,2,3,4
1	1	X	2	2,3	2,3,4
2	1,2	2	X	3	3,4
3	1,2,3	2,3	3	X	4
4	1,2,3,4	2,3,4	3,4	4	X

On the X-axis you see the number of ribbons-to-the-right in $ts(x)$, and on the Y-axis the number you want to switch to in $ts(x+1)$. In the table you can now find which ribbons are transported to the other side, and have therefore to be switched off.

By the way, this method does not work always. It's partially an unknown section of scenery-editing, and I can't really help you much here. Trial and error is the thing, unfortunately. One way to help you is this: Apply in the trouble-zone (temporarily) a good recognizable, and different texture to all ribbons, like water or gravel. This way, you can easily spot in the game which ribbon# gives the problem.

3.8 Lighting and Darkening of Ribbons

This is done with the scenery7, 0xCD command. It defines the lightlevel for a particular ribbon (or fence or verge). This can be used for instance to darken the inside of a tunnel. Another use is to lighten the ribbons of a house. Usually the houses are quite brown due to the sun being at the other side, therefore it becomes too brown. Lighting it gives it a fresh, beautiful look.

This command does not apply to objects. For objects, there are other commands that define the lightlevel, but these are still unknown.

3.9 Transitions

This is by far the most difficult part of making extensive scenery-structures, such as can be found on Tunneltrack and StuwLake. For every other structure, you have to come up with a different solution to make the transition from a mountainous part to a forest of trees look good and natural. But some things can be said about it, and may give you a hint of the path to follow.

Normally, if you just change the number of ribbons to a side, and adapt the 0xB8 values, there will be a piece of scenery crossing the track. Sometimes, though, you are lucky and are bug-free. But if you're not, you obviously have to switch off a ribbon for one TS. The problem is, to determine which one. A method that works is to put a different texture to each ribbon, and then you can easily spot which one is crossing the track. Then you know which one has to be switched off. A bit of a trial and error, but I can't think of anything else right now.

Chapter 4

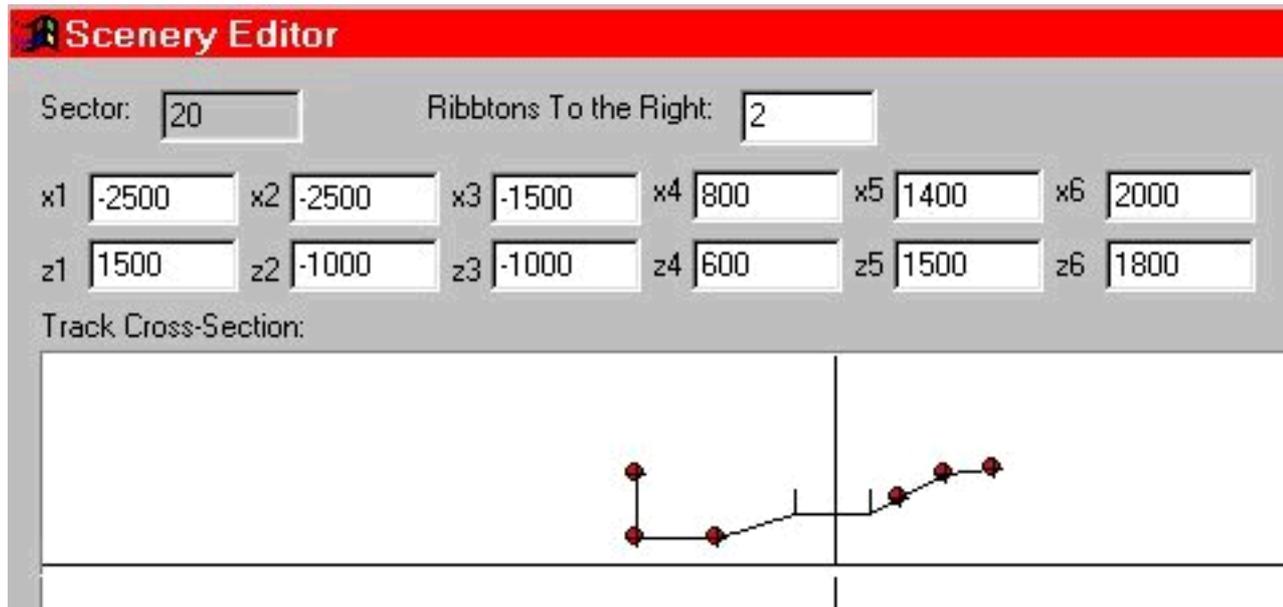
Structures and Formations.

The objective of this chapter is to make clear how certain features on the track are build of banks and ribbons. This may not be too obvious on first look, but after reading this chapter it should be clear.

If you come up to a certain structure that you wonder about, let me know, and I can include it.

4.1 Basic hill

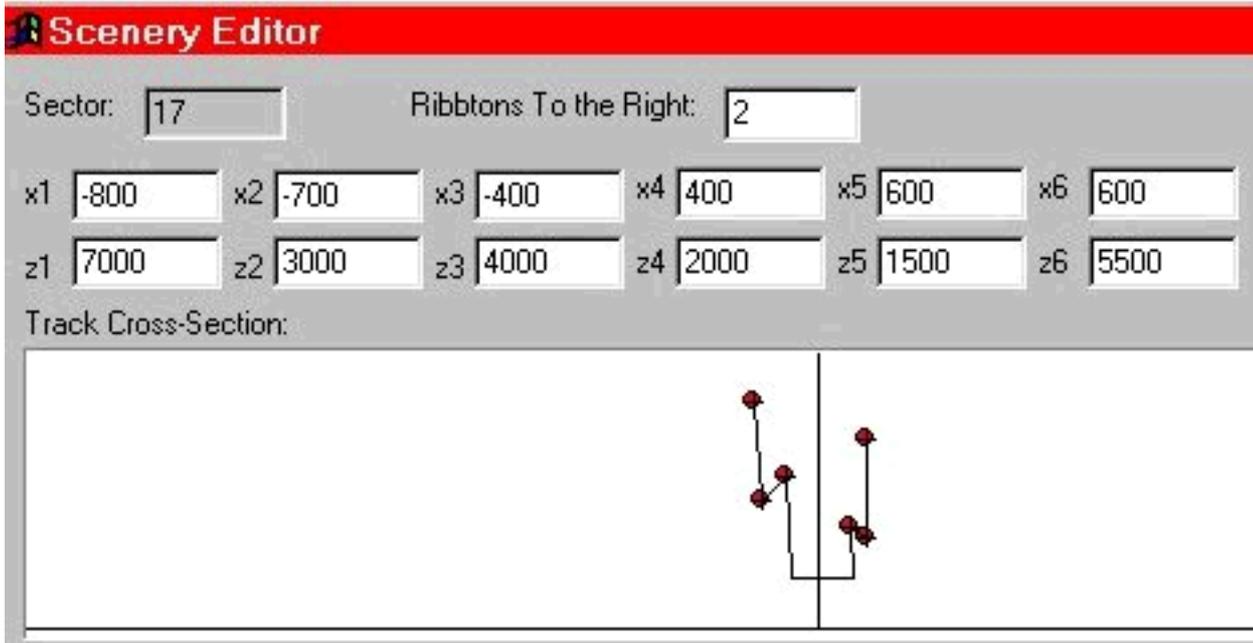
The basic hill is made out of ribbons that have a decreasing slope, to get the effect of depth. Most of the time, one ribbon and the bank is enough to make it look good.



The hill is the right side of the track. Note that the slopes of the bank and the first ribbon are equal, the split is made because of texturing- use.

4.2 Walls

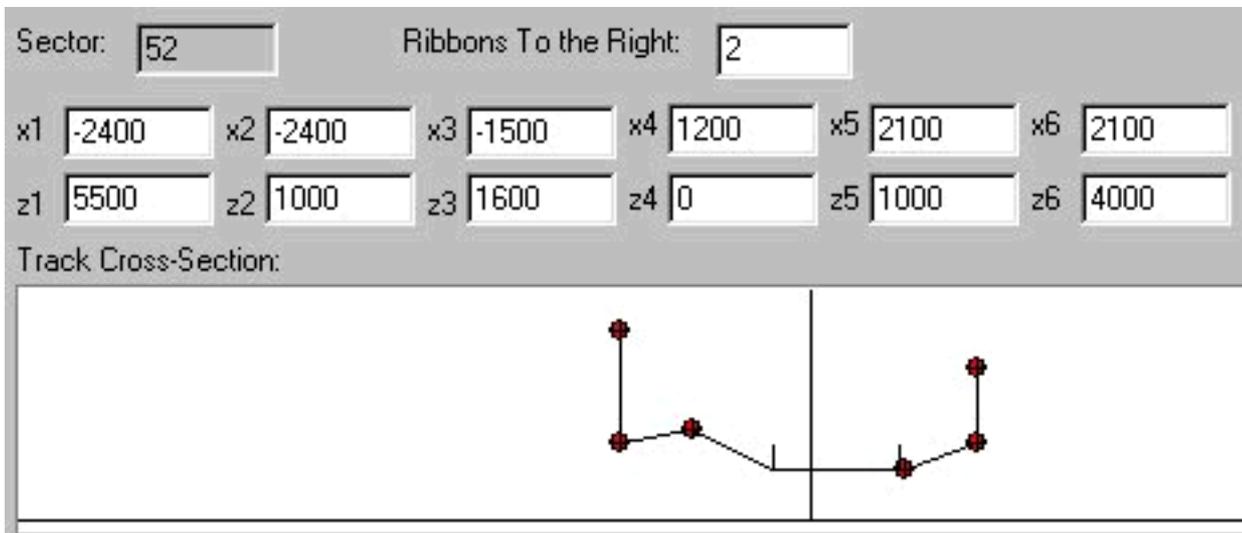
Walls are made in the same way as hills, although now the end of the bank (usually) has the same X-value as the start of the bank. Because the start of the bank is determined by the width of the track and the verges, it is sometimes difficult to get the wall totally straight up. For a "typical tracksection", a Xvalue of 500 might work. If you drive past it once in the game, you'll notice immediately if it's vertical or not.



Both sides have a wall here. Note the vertical banks, they form the walls here, but that is not necessary. Also, the right side seems to hang over the track a little. This may be just the picture drawn in the TE, as it's also difficult for Paul to see the relation between trackwidth, verge-width and the scenery-coordinates.

The thing that makes a wall look like a real wall is ofcourse the texture. But I initially decided that texturing commands and uses are outside the scope of this tutorial. If anyone feels it should be included, let me know.

4.3 Basic trees



Here you see trees on both sides of the track, the vertical ribbons (12 and 13). The right side has a small hill, on top of which the trees are build. On the left side, this hill recedes a little near the trees, giving an increased idea of depth.

4.4 Two lines of trees

Martijn's Scenery Tutorial

Sector: Ribbons To the Right:

x1	<input type="text" value="-2160"/>	x2	<input type="text" value="-2160"/>	x3	<input type="text" value="-1104"/>	x4	<input type="text" value="-1104"/>	x5	<input type="text" value="2000"/>	x6	<input type="text" value="2000"/>
z1	<input type="text" value="6112"/>	z2	<input type="text" value="1392"/>	z3	<input type="text" value="3072"/>	z4	<input type="text" value="832"/>	z5	<input type="text" value="0"/>	z6	<input type="text" value="1024"/>

Track Cross-Section:

Here you see the Fagnes curves on the original Spa-Francorchamps. There are three vertical ribbons, all carrying tree-textures. The two to the left are somewhat special; you can see the large, outer tree-line above, and behind the near, inner line of trees. This gives an impression of a tree-covered hill. Note that the ribbon 13 here is purely a connection-ribbon between the tree-lines, that is best switched off. Else, you will see some texture on top of the trees which is usually very ugly.

4.5 Cliffs

Sector: Ribbons To the Right:

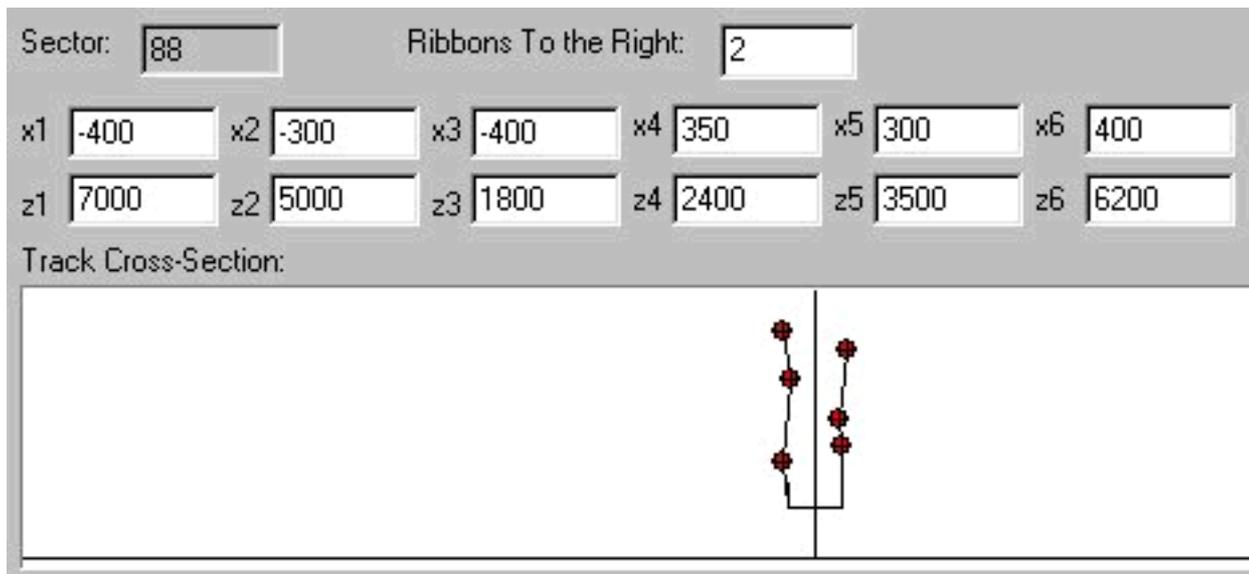
x1	<input type="text" value="-3200"/>	x2	<input type="text" value="-1200"/>	x3	<input type="text" value="-800"/>	x4	<input type="text" value="800"/>	x5	<input type="text" value="900"/>	x6	<input type="text" value="9000"/>
z1	<input type="text" value="3000"/>	z2	<input type="text" value="1200"/>	z3	<input type="text" value="600"/>	z4	<input type="text" value="100"/>	z5	<input type="text" value="-2000"/>	z6	<input type="text" value="0"/>

Track Cross-Section:

I think it's nice to drive along a lake with a F1 car, (it's however not very realistic). But, here you see how that lake could be build. On the right side, you go from the fence first a little horizontal, giving some room to place trees or adverts. Then, the scenery drops to the level of the lake, with an almost-vertical ribbon. This one is best covered with a rocky texture. Now, the lake is build very wide, and rises from the level of the cliff (z5 indicates the height of the cliff) again to zero, to enable a good attachment of the lake-ribbon to the horizon. If the lake is horizontal, there will be a gap between the level of the lake and the horizon, while you may want to make it look like it's never-ending.

4.6 Rocks

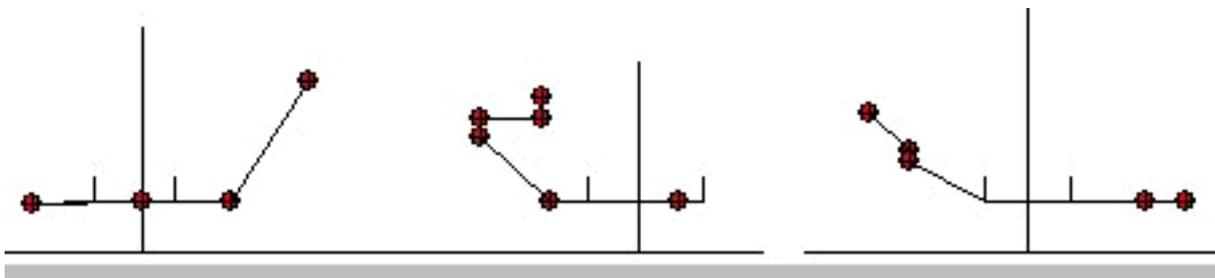
Martijn's Scenery Tutorial



To make a nice canyon, bend the ribbons a little like this. Make a new scenery1 cmd frequently, every 25 trackunits minimum. Also, make it a little erratic, to give the raw rocks-image. A rocky texture is also a good idea.

4.7 Grandstands

Grandstands are actually quite like hills, although with usually a different texture, and much more straight. The idea is, like the house in the next chapter, to make the sides perpendicular to the track. In this case however, the "roof" of the house is made larger, and the structure is made much longer. In the drawing you'll find some basic cut-throughs of stands, with or without roof, or multiple layers. Ofcourse, it's most fun to make one up yourself.



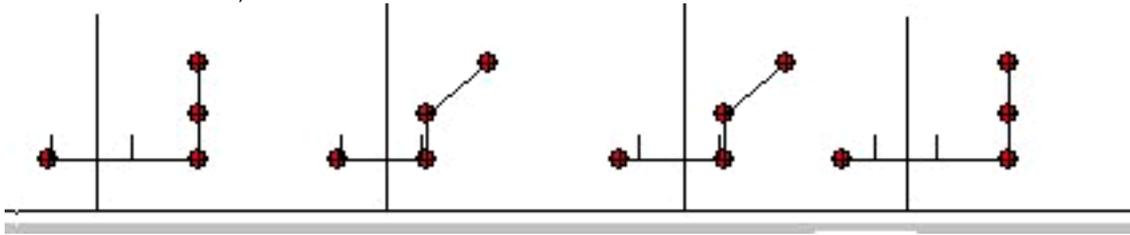
Left you see the grandstand used in the original Brazil track, before start/finish. It's just a plain slope, covered with crowd-texture. The main advantage of using ribbons for grandstands, over objects, is that you can easily curve the grandstands. That can be seen too on the original HockenheimRing, where the grandstands are curved for a long way. In the middle you see the grandstands that are build there, where there is a little roof build. The dimensions are a bit exaggerated, for clarity reasons. On the right you see a made-up grandstand, which you can find on the Elkhart Lake track (by Romeo and me), where the stands consists of two rows, clearly set apart by a vertical ribbon with another texture. These places are ideal for some advertisements.

4.8 Basic house

An impression of a house consists of two side-planes and a front-plane. These planes should be perpendicular to each other. In GP2 that's impossible to make using ribbons, and therefore we will use an approximation. As an example, I will use the house that's build in Tunneltrack. This house is build, using a particular order in which the scenery1 cmds are formed. First, you have a basic rock which is a little bend back at the bottom. Then, you bend the lower ribbon toward the track, and vertical, within one tracklength unit. That makes this plane almost perpendicular to the track. Now, you make another scen1 cmd some 15 units away, with the same values. This makes the

Martijn's Scenery Tutorial

track. Now, you make another scen1 cmd some 15 units away, with the same values. This makes the front-facade of the house. Finally, you make it bend backwards again, within one length-unit, to get the second side-plane. I will make drawings of this, and you should read it as a follow-up, so try to follow each ribbon, and see what it does.



4.9 Gallery

Not to be done.

4.10 Tunnels

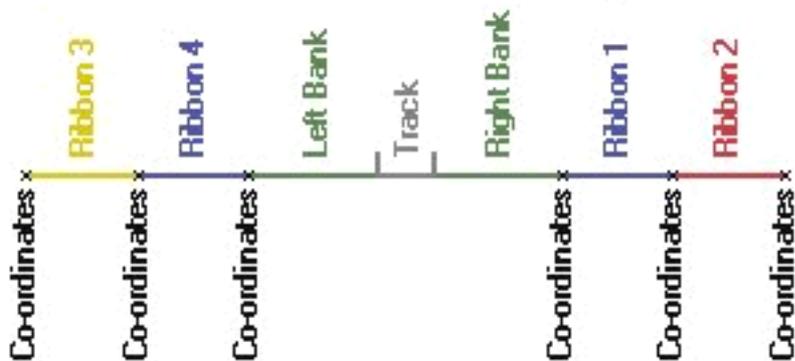
Jason Sinnbeck describes it like this:

2 RIBBON-PER-SIDE TUNNEL TUTORIAL

Having difficulties figuring out how to create a nice tunnel entrance & exit? Perhaps this short tutorial can help you. The following is a basic description of some values & commands that can be employed as a base to make a nice tunnel inside a low-ish hill.

The illustrations used in this tutorial show the ribbons and banks colour coded so as to make it simpler to understand which ribbon is doing what. Here is the key to the colour coding.

Colour Coding



< tunnel_0.jpg >

OK. Lets get started. In the diagram below we are locating the ribbons to form the outer bases of our hill. Hence the large X values to take them out a long way out from the track, and the absence of height (all Y values = 0) since our hill will naturally be starting from ground level.

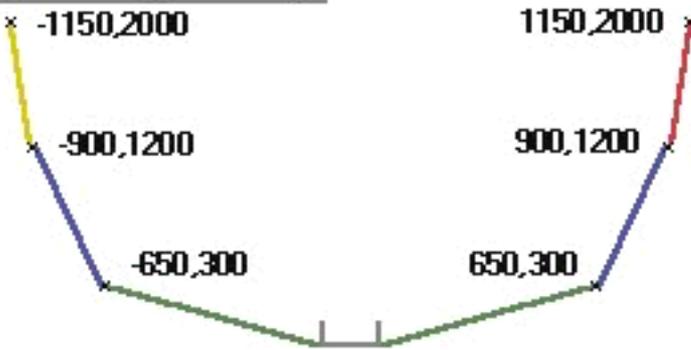
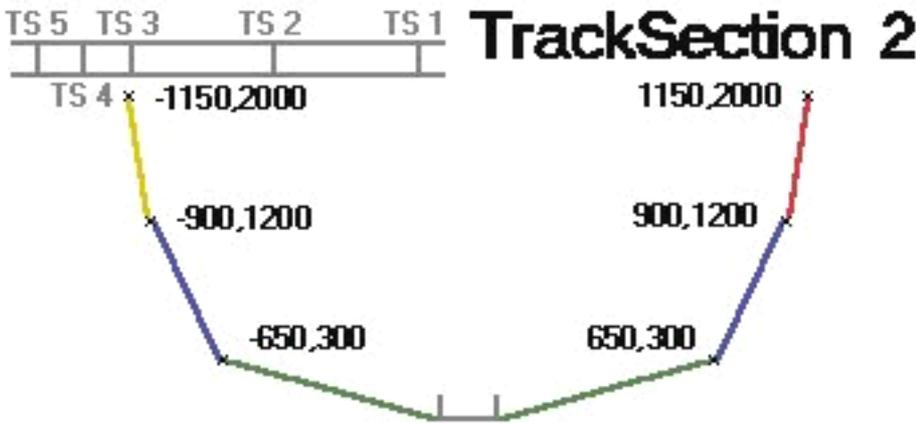
Martijn's Scenery Tutorial



< tunnel_1.jpg >

With the track section 2 scenery command we start to raise the ribbons and bring them back towards track, thus creating the start of our hill.

Note: For the values used in this tutorial, track sections 1 & 2 should be in the order of 10 units length to give a nice result.

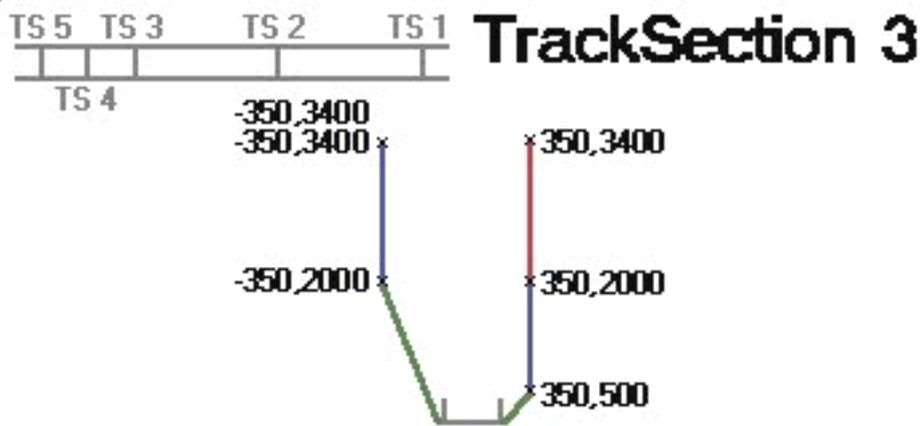


< tunnel_2.jpg >

The scenery command at the start of TS#3 is really just a continuation of the trend set by TS#2 scenery values. Remember that the rate of rise (gradient) of the hill through TS #1 is controlled by the values at the start of TS#2, and that TS#2's gradient is controlled by TS#3's values. The reason that this had been pointed out is that if you study the Y values between the TS#2 command and the TS #3 command you will notice that about 55-60% of the rise of the hill occurs DURING TS#1, and TS#2 is left with about 40-45% of the remaining height that makes up the Y co-ordinates total value at TS#3. This gives our hill a "rounding off" effect.

Remember that both TS#1 & #2 have to be of a similar length for this to be valid.

Martijn's Scenery Tutorial



< tunnel_3.jpg >

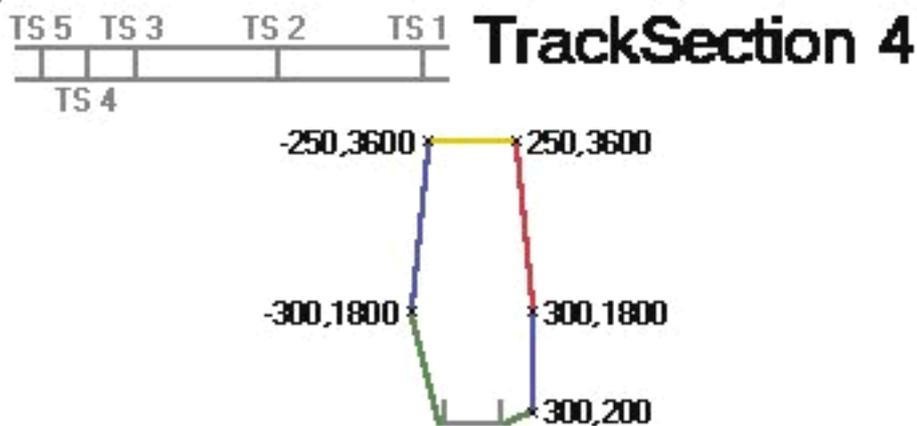
You could leave out TS#2 altogether and just have a straight hill rise to the start of TS #3. Alternately, you could put more sections in for an even nicer rounding effect on the hill side. Whatever you feel like!

You have no doubt noticed that Ribbon 3 finishes TS#2 (i.e starts TS#3) at the same co-ordinates as ribbon 4. That is because we have to turn ribbon 3 off for it's journey through TS#3 as it must cross the track. If it was still higher than Ribbon 4 we would see it suddendly stop when turned off. Not very pretty. So we piggy-back it briefly on ribbon 4 to hide it's changing of state.

To this extent, TS#3 must have an 0xb0 command to turn ribbon 3 off. The section offset a1 will be 0, and the location code for a2 will be 4. Also, TS#3 is typically very short. Just one length unit is normal.

Now to the start of TS#4. Ribbon 3 has crossed the road and is connected to the top of ribbon 2. Time to turn it back on. The simplest way is to just throw in an 0xd9 with an a1 value of zero, and with a2 = 63. This turns 'em all on.

Also notice that there has been a slight increase in the hill's height again. This is optional, but I think it looks a little better this way.



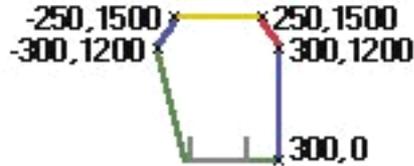
< tunnel_4.jpg >

TS#4, like TS#3, is also usually just 1 unit in length.

And now last, but by no means least, we reach TS #5; the actual start of our tunnel. The values used here are simply copied throughout the length of the tunnel.



TrackSection 5



< tunnel_5.jpg >

During TS#4, ribbons 2 & 4 have been lowered, taking the still-connected ribbon 3 with them. This gives us the front face of the hill that becomes the roof of the tunnel in TS#5 once lowered.

When you get to the exit, it is simply a reverse of the entry (except that you will still have the 0xb0 before the 0xd9 of course) and then you can adjust the values to achieve the desired result. For example, the fictitious Eagles Nest circuit (available from Sjon's) shows a nice version of an exit that becomes a hillside spectator mound on one side.

The only other thing to mention here is that the track verge values used are 15. If you go for a tight tunnel with small verge values, say about 12 or less, you may notice that you have flags being waved by a marshall's arm which is protruding from the wall on the side of your tunnel. Ghost marshalls perhaps? Kinda spooky, and not too reassuring for Schuey, Fissi, Damon & co as they blast on through....

Chapter 5

Addie's Special-Request Section

In this section some things about subjects that belong sideways to scenery will be said. I will make some notes about the use of the 0xBC command, as it is closely related to ribbons. The part about view-distances is a bit of a repetition of the commands-library, but more integrated in the scenery-stuff.

View-distances

The game determines how much of the track need to be drawn in front, by using commands. This command is NOT a definition of the "distance the beam of light shines in front of the car", but more of an one-time placement of scenery, where you drive on to. That means, you have to place these commands more than once on the track. Best bet is to place the commands in the middle, or just before the corner. The view-distance value should be put then, so that the whole straight following the corner is viewed when exiting the corner. So for a long straight, you need to give a high view-distance in front, and for a shorter one, a lower value should be alright. The trade-off here is between the distance in front you see (and need) and the processor-occupancy, because longer view-distances give the computer more to think about, and thus higher occupancy. The maximum value you can safely enter is 255, you are advised to not go over this limit.

The commands used for view-ahead are 0x81 and 0xBE. Exactly why there are two commands remains

Martijn's Scenery Tutorial

The commands used for view-ahead are 0x81 and 0xBE. Exactly why there are two commands remains unknown, but just remember to put an 0x81 AND and 0xBE in the middle or before every corner. It is advisable to use the same distances in both commands, although not necessary. For the location code, I would say use 63, but maybe 48 works just fine.

Maybe the best way to check the "view distance behind" commands is by driving the track the wrong way around. Then, you can immediately spot that for a long straight, the view-command is placed AFTER the straight, not before. So build them in exactly the same way as the view-distances in front, when thinking of driving the wrong way. Use the commands 0x82 and 0xBF.

The units of these commands is the same as the tracklength-unit so you can calculate which values you need by looking at the length of the straight following your command. Most of the time, values around 150 are enough.

Appendix

A1. Frequently Asked Questions

1) I have clicked "Remove Scenery" but now I want scenery on my track!

This really is no problem anymore!

- Insert a lot of Scenery3 (0xAF) and Scenery1 (0xB8) commands, before and after each corner at least. Make sure the Scenery3 command comes before the Scenery1 command! To do this efficiently, insert a scenery3 command in the first TS, "copy" it, and "paste" it at strategic places. Then, insert a Scenery1 command at TS0 and fill in some decent values. Now, "copy" this command, then paste it at exactly the same locations as you put the Scenery3 cmds in. Don't miss any, and don't put in too many. Next, include a scenery4 [0xD9] command [a2=63] in the first TS, like told above. Then, remove all the influencing commands as explained in Ch3, Guideline.

2) Can you make the scenery for my track?

No, of course not! Then it would become my circuit, and no longer yours! It would cost me hours and hours, and I'd rather spend those on a track of my own. Only in very special cases, like when it's a really good track, I will consider making scenery for it.

3) Where can I find the Scenery-tutorial?

Well, you already have it... But you can usually download the most recent version at <http://www.sepa.tudelft.nl/webstud/426090/index.htm>
The tracks I build are also available there.

4) Why has nobody builded the old Nurburgring yet?

You really are a newbie, and should stop reading this scenery-tutorial at once. It's way too advanced for you. Go play Toca, and do not bother us with those stupid questions. Or, if you want to become a trackditor yourself, read first Addie's general guide to Trackediting and most of your questions will be answered there. OK, I will give it here for the last time: GP2 tracklength is limited to around 7 kms. So no way the old Nurburgring fits in this. Same goes for the old Spa track, and the Le Mans track.

A2. Glossary

Banks

The space between the fence and the ribbons are called "banks" nowadays. You will find this term also in the "graphics options" screen in the GP2 game itself. Switch the texture off, and you'll see exactly

Martijn's Scenery Tutorial

in the "graphics options" screen in the GP2 game itself. Switch the texture off, and you'll see exactly what is bank and what is ribbon. Another way to tell the difference is by adjusting the detail level from within the car (alt-D). If you press it twice, you will find all the ribbons gone, but the banks are still there!

Verges

This is the space between the actual road and the fence. The car can drive over the road and the verges, but not further. The verges can be textured with anything you like (adverts, rock, grass, gravel-trap, tarmac). There are also three different types of "actions" involved with driving on the verges: The first is your normal wobbly grass (default), second is the smooth tarmac (apply TexID 33), third is the difficult-to-plough-gravel trap-action, attached to TexID [I never use this]. So the texture-ID determines which action is taken! This is unlike the actual road-floor itself, that ALWAYS functions like tarmac (surprise!)

Fences

FKA "walls", these things are the ones that prevent your car from going into the wild unknown nothingness of the unmapped and untextured outside world. You must have bumped into them once... There can be a texture on the fences, but it can also be removed. That causes your car to bump into the nothingness and bounce back, without seeing what you just hit!

A9. Version Numbers

version 2.0

corrected the "swivel arms" section (chapter 1.4)

version 1.9

new "question", slight updates here and there, added the "tunnel" section by Jason Sinnbeck.

version 1.2

Added chapter 5, and filled in some empty parts.

version 1.1

added section 1.8, 3.5 and 3.7

version 1.0

added some missing structures, rewrote some more command-descriptions (0xC0, 0xC1, etc), much more on scenery-setup and using it, changed "credits" section, added some glossary terms, placed the table again somewhere else, added 1.7, 3.2 - 3.4, 3.9, 4.3 - 4.8,

version 0.9

fixed some mistakes and typos, started rewriting commands, added a few "basic structures", new description for 0xD0, 0xD9, added the "texture location codes" section,

version 0.8

First published version although far from complete.