

ORCF User Guide

Written for ORCF **2011.07p**

July 14, 2011

Contents

1	Getting started	4
1.1	What is ORCF?	4
1.2	System requirements	4
1.2.1	Minimum	4
1.2.2	Recommended	4
1.3	Get ORCF	4
1.3.1	Binary core packages	4
1.3.2	Source core packages	4
1.3.3	GIT version	4
1.3.4	Data packages	5
1.4	Compiling from source	5
1.4.1	Linux	5
1.4.2	Mac OS X	5
1.4.3	Windows	5
2	Settings	6
2.1	Language	6
2.2	Screen	6
2.3	Graphics	6
2.3.1	General	6
2.3.2	Effects	7
2.3.3	Water	8
2.3.4	Reflections	9
2.3.5	Presets	10
3	Gameplay	11
3.1	Terrain editing	11
3.1.1	Creating marks	11
3.1.2	Selecting height	12
3.1.3	Modifying the landscape	12
3.1.4	Resizing	12
3.1.5	Water	12
3.1.6	Changing textures	13
3.1.7	Changing the texture collection	13
3.1.8	Other stuff	13
3.2	Building objects	14
3.2.1	Selecting the object to build...	14
3.2.2	Placing an object	15
3.2.3	Coloring an object	15
3.2.4	Modifying built objects	16
3.2.5	Deleting objects	16
3.3	Building pathes	16
3.4	Building flatrides	16
3.5	Building tracked rides	17
3.6	Taking screenshots	17
3.7	Recording videos	17

3.7.1	Encoding with ffmpeg	17
3.8	Creating cameras and camera routes	17
3.9	Saving a park	17
4	Troubleshooting	18

1 Getting started

1.1 What is ORCF?

Open RollerCoaster Factory aims to be a free open-source theme park simulation game running on Linux, Mac OS X and Windows. Other than e.g. Roller Coaster Tycoon, this is not about economy and money but about building and enjoying good-looking parks and extending the game with custom objects and rides.

Although development started in November 2009, the game could not even reach its beta phase yet, but it is actively developed so that the first stable version shall be released September 2011.

1.2 System requirements

ORCF is a resource-hungry game relying on modern 3D technology and does therefore not run on every computer.

1.2.1 Minimum

- Any x86 or x86_64 CPU
- Video card with Shader Model 4.0 support (NVidia GeForce 8000 series or ATI Radeon HD 3000 series)

1.2.2 Recommended

- Dual-Core CPU with 2 GHz
- Very fast video card (NVidia GeForce 9800 GT, ATI Radeon HD 5450) with 512MB of dedicated VRam
- 2 GB of RAM
- A few GB of free HDD space for additional data packages

1.3 Get ORCF

At the moment there is no stable version of ORCF. There are development snapshots available as well as a continuously update GIT repository, but all these sources have in common that they ship an unstable game that still lacks lot of features.

1.3.1 Binary core packages

Not available yet.

1.3.2 Source core packages

Not available yet.

1.3.3 GIT version

To clone the GIT repository, install GIT and do the following:

```
1 | $ git clone git://github.com/ireyon/Open-Rollercoaster-Factory.git orcf
```

Be warned: This contains the current development snapshot which is, in most cases, highly unstable and not ready for release. If you want a working game, use the source or binary packages from above.

1.3.4 Data packages

In order to have something to build a park with, you need data packages which you can find ... nowhere yet.

1.4 Compiling from source

If you want to compile ORCF from source, which is needed when you downloaded the GIT version or a source package, you need at least Version 2.2 of the FreePascal compiler which you can get either from your Linux distribution's package repositories or from the official website¹.

1.4.1 Linux

You need to install OpenAL and SDL packages for ORCF to link properly. Development headers should not be needed.

Compile and start the game using the following commands:

```
1 | $ cd orcf
2 | $ ./configure
3 | $ make
4 | $ make tools
5 | $ ./orcf
```

There is currently no supported way of “installing” the game to `/usr/local`. Instead, you might move the entire ORCF directory to `/opt` if you want multiple users to be able to use it.

1.4.2 Mac OS X

You need to install GLFW and OpenAL for ORCF to link properly.

Compile the game using the following commands:

```
1 | $ cd orcf
2 | $ ./configure
3 | $ make
4 | $ make tools
5 | $ ./bundle-orcf.sh
```

This should result in a working application bundle.

1.4.3 Windows

Compiling ORCF on Windows is somewhat trickier than on other systems. First, open `compile.bat` file in a text editor. You will see three lines starting like

```
1 | fpc.exe -MObjFPC -g -Sghi -vewnhi -Fu. -Fi. -Fu.\main
```

¹<http://www.freepascal.org/download.var>

Prepend the installation directory of your FreePascal compiler to every of these lines so that your computer can find it.

Now, run the compile.bat file either by double-clicking it from a file manager or from a CMD window. ORCF ships windows DLLs for SDL and OpenAL so you don't need to install these.

If you used the GIT repository, create a directory named `config` in the orcf root directory before running the game.

2 Settings

When you first run the game, it will load some default settings: A 800 by 600 window, English language and poor graphics. If you want to change it, click the SETTINGS button in the main menu.

2.1 Language

Currently there is only a text field where you can enter `en` or `de`, but there is no complete German translation yet.

You must restart the game for the changes to take effect.

2.2 Screen

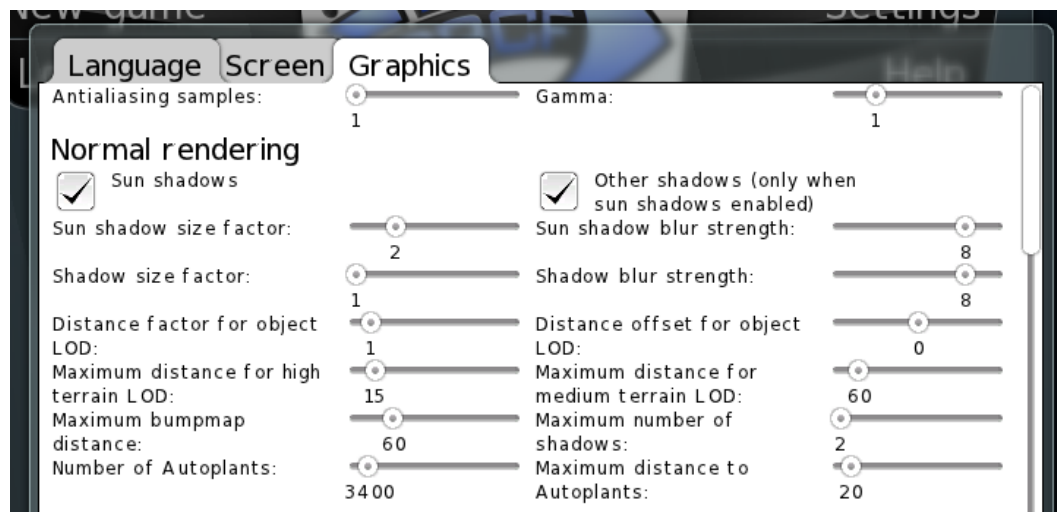
Currently there are two text fields for the window dimensions.

You must restart the game for the changes to take effect.

2.3 Graphics

You can almost control everything related to rendering. The options are divided into the following four groups.

2.3.1 General



Antialiasing samples: Specifies how many samples should be rendered per pixel. Higher values result in smoother edges, but also in significantly higher VRAM consumption and poor performance. Values of 2 and more are good for videos.

Gamma: Inverse exponent of each pixel. Does not affect user interface. Usually, 1 is the best value.

Sun shadows: When checked, the sun casts a shadow map.

Other shadows: When checked, all light sources can cast shadows. However, it doesn't mean that every light source *does*.

Sun shadow size factor: Multitplied by 1024, this value specifies the size of the sun shadow texture.

Shadow size factor: Multitplied by 256, this value specifies the size of the other shadow textures.

Sun shadow blur strength: The blur radius for soft shadows casted by the sun. Higher values do *not* mean stronger blur, but they *do* mean nicer appearance of blurred shadows.

Shadow blur strength: The blur radius for soft shadows casted by normal light sources.

Distance factor for Object LOD: The minimum and maximum visibility ranges for objects will be multiplied by this value. The higher the value, the more details distant objects will have as long as they use LODs.

Distance offset for Object LOD: Value to be added to visibility ranges. Normally, 0 should do.

Maximum distance for high terrain LOD: Size of the terrain quad which contains the smallest sub blocks.

Maximum distance for medium terrain LOD: Maximum distance for macro blocks with medium sized sub blocks.

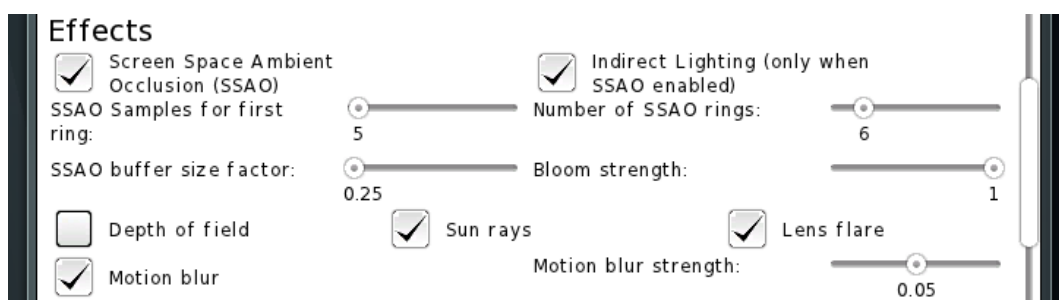
Maximum bump map distance: Distance to which the terrain appearance will be improved by normal maps.

Maximum number of shadows: The number of normal light sources that will cast shadows. Only the light sources that affect the viewer's position most will get a shadow map.

Number of autoplants: Number of quads that will be rendered for dynamic grass.

Maximum autoplant distance: Distance after which auto plants will be removed and put elsewhere.

2.3.2 Effects



Screen Space Ambient Occlusion (SSAO): Effect that is used to gather details in shaded areas.

Indirect Lighting: Effect that causes objects to be illuminated by light-emitting materials.

SSAO samples for first ring: The number of pixels to be used for SSAO calculation in the first SSAO ring. If it is set to 5, it will use 10 samples for the second ring, 15 for the third and so on. The higher the value, the more accurate the effect will look.

Number of SSAO rings: Number of SSAO rings. The higher the value, the more accurate the effect will look, but it has huge impact on overall performance.

SSAO buffer size factor: Multiplied by the screen resolution, this will give the size of the SSAO buffer.

Bloom strength: Strength of HDR bloom. Set to 0 or 1, anything else will look strange.

Depth of field: When enabled, the object under the mouse cursor will be focussed, anything more or less distant will be blurred.

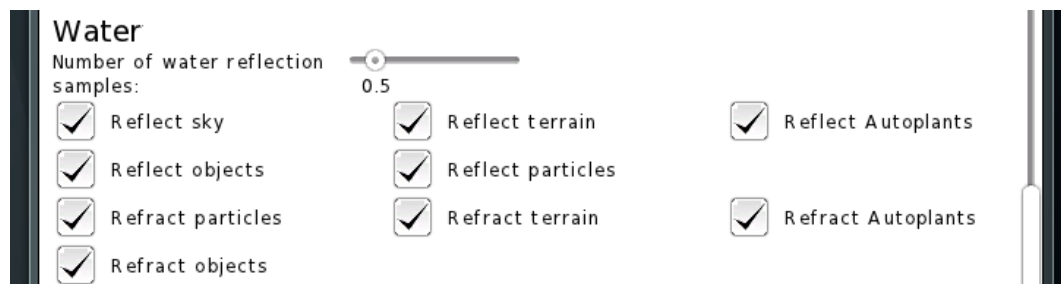
Sun rays: Simulates visible sun rays that are especially nice on sunsets.

Lens flare: Simulates camera lens effects.

Motion blur: Moving parts of the scene will look a little blurry.

Motion blur strength: The higher the value, the more blurry a moving scene will look.

2.3.3 Water

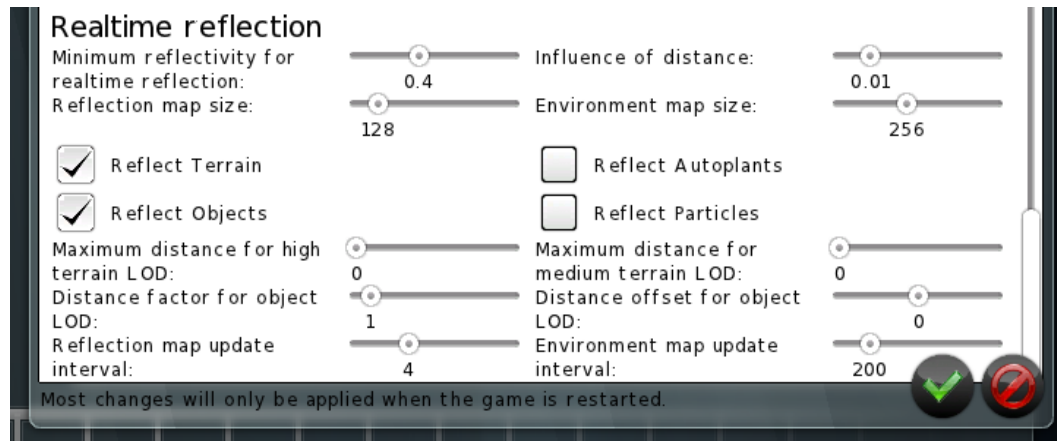


Number of water reflection samples: Multiplied by the screen resolution, this represents the resolution of the water reflection and refraction buffers.

Reflect ...: Specifies which parts of the scene will be reflected in water.

Refract ...: Specifies which parts of the scene will be visible when looking from above into the water.

2.3.4 Reflections



Minimum reflectivity: Reflectivity value a material must have to be able to use realtime reflection.

Influence of distance: The higher this is, the faster the minimum reflectivity value increases for distant objects. When too far away, even very reflective materials will only show the environment map to improve performance.

Reflection map size: Edge length in pixels of the reflection maps.

Environment map size: Edge length in pixels of the environment map.

Reflect ...: Specifies which parts of the scene shall be reflected in real time. The environment map only reflects the sky and the terrain.

LOD stuff: See above.

Reflection map update interval: Number of frames that shall pass before any of the dynamic reflection maps will be updated.

Environment map update interval: Number of frames that shall pass before the static environment map will be re-rendered.

2.3.5 Presets

Later, there will be some presets that makes the engine configuration much easier and faster.

	Default	Performance	Quality	High Quality	Video
Sun shadows		✓	✓	✓	✓
Other shadows			✓	✓	✓
Sun shadow size factor		2	2	2	3
Shadow size factor			1	2	2
Sun shadow blur radius		0	2	4	7
Shadow blur radius			2	3	6
Distance factor for objects	0.6	0.8	1	1.5	2
Distance offset for objects	0	0	0	0	0
High Terrain LOD distance	10	12	15	20	25
Medium Terrain LOD distance	40	50	60	90	120
Bumpmap Distance	40	50	60	90	120
Number of shadows			5	10	20
Number of autoplants	0	2000	3400	7700	13600
Autoplant distance		15	20	30	40
Ambient Occlusion			✓	✓	✓
Indirect Lighting				✓	✓
SSAO Samples			5	7	9
SSAO Rings			6	9	12
SSAO buffer size			0.25	0.5	1.0
Bloom strength	0	1	1	1	1
Depth of field				✓	✓
Sun rays			✓	✓	✓
Lens flare		✓	✓	✓	✓
Motion blur		✓	✓	✓	✓
Reflection samples	0.25	0.33	0.5	0.75	1
Reflect sky	✓	✓	✓	✓	✓
Reflect terrain		✓	✓	✓	✓
Reflect autoplants				✓	✓
Reflect objects			✓	✓	✓
Reflect particles				✓	✓
Refract terrain		✓	✓	✓	✓
Refract autoplants				✓	✓
Refract objects			✓	✓	✓
Refract particles				✓	✓

Minimum reflectivity	1.0	1.0	0.7	0.4	0.0
Influence of distance			0.03	0.01	0.00
Reflection map size			128	192	384
Environment map size	128	192	256	384	
Reflect terrain	✓	✓	✓	✓	✓
Reflect objects			✓	✓	✓
Reflect autoplants					✓
Reflect particles					✓
High terrain LOD distance			0	0	0
Medium terrain LOD distance			0	0	25
Distance factor for objects			1	1	2
Distance offset for objects			0	0	0
Reflection map update interval			4	2	1
Environment map update interval	1000	500	200	100	

3 Gameplay

At the moment, this is under heavy development. Although you can already modify the terrain and place simple objects, almost nothing has the desired functionality yet.

3.1 Terrain editing

Due to the little block size, ORCF uses a marked area based terrain editor. Open it with a click on the terrain symbol and you will get this window:



Figure 1: The terrain editor

Note that the area which does *not* belong to your park will be darkened.

3.1.1 Creating marks

The first step of every terrain editing action is to create a marked area. To add marks, click the large button with the ADD MARK icon.



Figure 2: The ADD MARK and REMOVE MARK buttons

Now you can click on the terrain as much as you want to. Right-click an existing mark to delete it and left-click it to insert a new mark after it. The currently placed mark appears in red color.







Note that the larger the marked area is, the longer the terrain modifications will take. Creating a huge water layer sometimes takes up to 30 seconds in which the game seems to do nothing, although it is still running.

3.1.2 Selecting height

The terrain editor has a text area with two arrows next to it. With that you can select the height level you want to raise a hill or cut a valley to—but you can also pick the height level of the last mark you placed by pressing the MARK button or select a height level with the mouse by clicking PICK.



3.1.3 Modifying the landscape

On the left of the terrain editor, you see six buttons. From left to right, these do the following:

-  creates a (kinda) smooth hill. Does not work perfectly yet on complex selections.
-  creates a smooth valley with the same problems.
-  sets the maximum height of an existing hill.
-  sets the maximum height of an existing valley.
-  sets the height of the marked area exactly to the selected value.
-  tries to smooth the marked area. This is very slow.

Note that objects that are placed in the modified area will *not* be moved!

3.1.4 Resizing

With the little  and  you can resize the terrain along the X and Z axes. The larger your building area is, the more you can do, but the slower further modifications will be. The file size of a stored park will also increase by a lot. **Note** that the numbers (1024, ..) are the number of sub blocks in either direction, not meters. If you want meters, divide it by 5.

Note that the default building area is quite small. It has only 204.8x204.8 meters, whereas an empty park in RCT3 has something like 512x512 meters.

Note that a feature to move the entire park along the axes is planned, but not implemented yet.

3.1.5 Water

Water is one of the best-working features, although stuff like waterfalls is still missing. To create a new water surface, click the SET WATER button. You will get a height line under which the water will be. However this doesn't set the water level for the entire park, it rather fills the connected

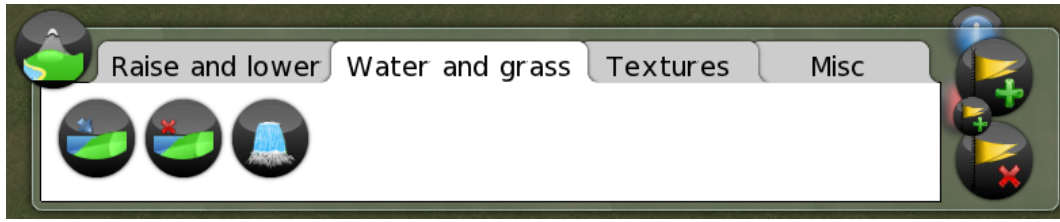


Figure 3: The water and grass tab

region under your mouse. So, if you want to fill a region and the height line is interrupted, only a smaller part will be filled. **Note** that this again is very, very slow.

If you want to delete a water surface, click the DELETE WATER button and select a surface. No height lines here.



Figure 4: SET WATER and DELETE WATER

3.1.6 Changing textures



Figure 5: Everything about the terrain texture collection

If you want an area to look sandy, dirty, rocky or snowy, you can choose between eight textures supplied by a so-called texture collection. Click on one of the eight round buttons and the area you marked will be changed. This is usually relatively fast.

3.1.7 Changing the texture collection

If you don't like the default eight textures, you can load other terrain texture collections with the LOAD NEW button. It will also replace the autoplant appearance.

3.1.8 Other stuff

The left part is used for conditional texture changes. If you, let's say, have a mountain and you only want the top of it look like snow, you can enter the height level you like in the text area (or just pick it with the functions in the RAISE AND LOWER tab), then select HIGHER THAN... and the desired snow texture. **Note** that for STEEPER THAN..., the compared values are usually somewhere between 0 and 2. No picking here (yet), sorry.

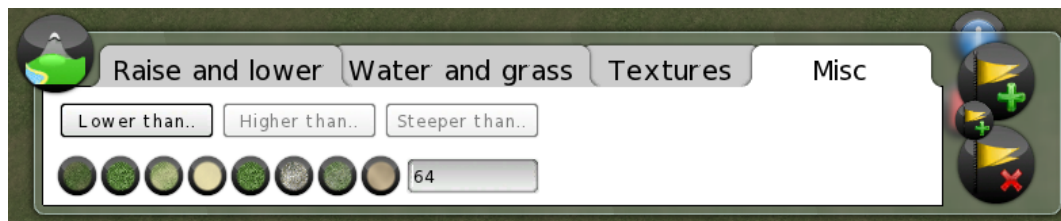


Figure 6: Some other functions

3.2 Building objects

In ORCF, you are almost free where and how to place objects. The only limitations are the building area of the park, the accuracy of 32 bit floating point numbers and a height of about 250 meters.

3.2.1 Selecting the object to build...

All objects, rides and pathes can be built using the object selector. Click on the OBJECTS button.



Figure 7: The OBJECTS button

You will get a large window looking like this:



Figure 8: The object selector

On the left part, you can select or deselect groups. Objects that are in any of the selected groups will be shown in the middle park. So, if you only want medieval lamps, I have to disappoint

you, you will most probably have to search them in the list of lamps or in the list of medieval objects.

The search bar on the top of the window behaves differently, only objects and sets that match every word in their name, their author's name or their description.

Once you select an object from the middle part, metadata such as groups the object belongs to will be displayed. A big ADD button will appear in the bottom right corner when the object is loaded—that might take a while, especially scripted objects might take a second more to get ready.

3.2.2 Placing an object

When you click that button, the object builder window will open:

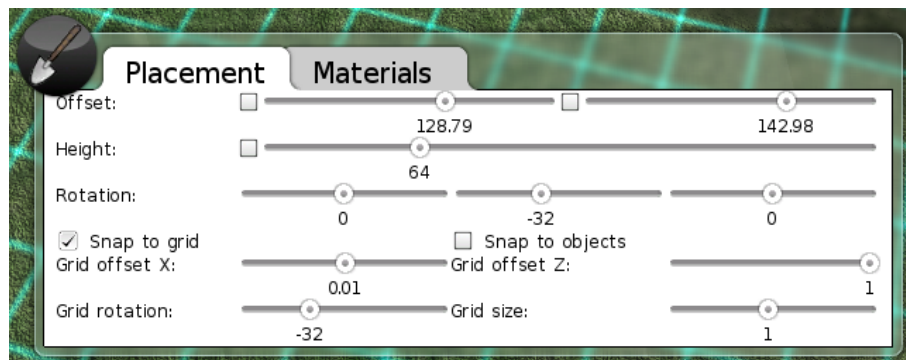


Figure 9: The object builder

That's a lot of sliders, I agree. But you don't need to change any of them normally: If you move the mouse over the park, you'll see that the object adjusts to your mouse position and the POSITION and HEIGHT sliders change as well.

If you want to lock any of the position axes, tick the check box next to the sliders. They won't change anymore.

If you press CTRL and scroll your vertical mouse wheel, the object will go up and down, the Y position (HEIGHT) will be locked.

If you press CTRL and your middle mouse button and move the mouse to the left or right, you rotate the object in 5 degree steps along the Y axis. To rotate around the other axes, you must adjust the sliders.

When the grid is enabled, you might want to rotate it the same way as the object. To do that, press CTRL and your right mouse button, you can rotate the grid just like the object.

Play around with the other options to see what they do.

3.2.3 Coloring an object

There is this nice MATERIAL tab in the object builder. When you click that, you will see a list of materials on the left and the properties of the currently selected material on the right. You can change the color, the level of light emission and the reflectivity of every material. The names of the materials are stored in the objects themselves, which makes the object's author responsible for them.

If you accidentally changed something you didn't want to change, just click the REVERT button next to the property you want to reset.

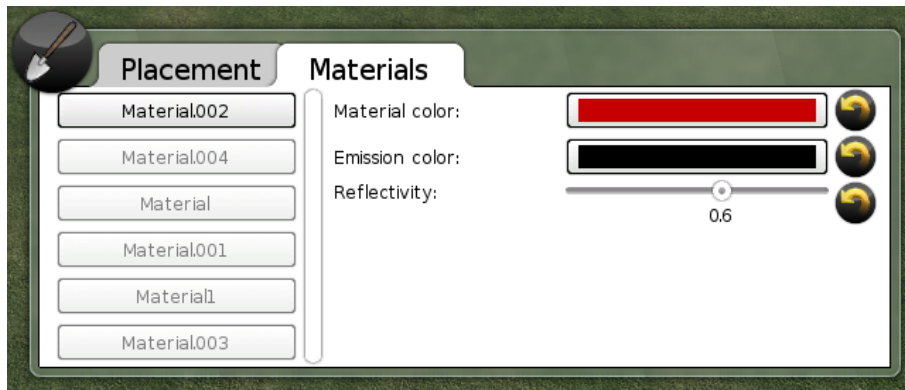


Figure 10: The material editor



Figure 11: The REVERT button

3.2.4 Modifying built objects

Under normal circumstances, you only have to click on an object to edit its materials or to move it around.

3.2.5 Deleting objects

To delete an object, you first of all have to change the selection mode. Click the DELETE button



Figure 12: Three selection modes: NORMAL, DISABLED and DELETE.

and then select the object you want to delete. **Note** that the game does not ask you whether you really want to remove an object or not. Be careful with transparent objects as well.

You can also use the keys 1, 2, 3 on your keyboard (not on the numblock!) to change the selection mode. Don't forget to reset it when you don't want to delete anything anymore.

3.3 Building pathes

Not implemented yet. Will be something using Bézier curves.

3.4 Building flatrides

Not implemented yet. Will almost be the same as objects.

3.5 Building tracked rides

Not implemented yet. Again, something like objects with Bézier curves.

3.6 Taking screenshots

When you press F10, a screenshot will be recorded and saved in the screenshots directory somewhere in the ORCF home data path. When pressing CTRL+F10, the user interface will be included in the screenshot.

3.7 Recording videos

When you press F11, a video will be recorded and saved in the screenshots directory somewhere in the ORCF home data path. When pressing CTRL+F11, the user interface will be included in the video.

ORCF stores every single frame as a TGA file. Therefore you must encode it to a format your favorite video editing software can handle. This can be done for example using `ffmpeg`². If you prefer applications with a graphical user interface, I still have to disappoint you, but ORCF will get an interface for `ffmpeg` later. But that's something for a stable version.

3.7.1 Encoding with `ffmpeg`

`ffmpeg` is a cross-platform command line utility with which you can transcode video and audio between a vast number of codecs and container formats and even has the ability to read input from single images like TGA files.

To encode a video, do the following:

```
1 | $ cd ~/orcf-data/Videos/Video_0
2 | $ ffmpeg -r 25 -i frame_08d.tga -vcodec libtheora -vb 8000k video.ogg
```

Note that not every video editing software can handle OGG/Theora video. Replace `libtheora` by `libx264` or `mpeg4` and `video.ogg` by `video.avi`, `video.mp4` or `video.mkv` in order to get more compatible files.

Note that you must use `\` instead `/` of on Windows.

Note that the bitrate of 8000k is good for 1280x720 videos. Increase it to 16000k for 1920x1080 and use lower bitrates for lower resolutions

3.8 Creating cameras and camera routes

Not implemented yet.

3.9 Saving a park

Click on the quit icon and then save the park file using the save button.



Figure 13: The quit and save icons.

²<http://www.ffmpeg.org/download.html>

4 Troubleshooting

If you have problems, make sure that:

- your computer matches the recommended system specifications as listed in 1.2.2
- you use the latest official ORCF release
- you did not mess up your ORCF install by violently deleting or moving important files or directories.

To report a bug, contact me via mail, ICQ, XMPP, whatever way you like, describe the problem as good as possible and also tell me which data files you used before the game crashed or behaved an unexpected way. In some cases it might be needed to attach the park file as well, bugs can only be fixed when the developer can reproduce them.