Terrain Tutorial using Shade-C

By padmalcom (February 2011)

www.jofre.de

1 Introduction

If you want your exterior levels to look good you will not come around the usage of terrains. Terrains are basically square 3D models which are generated from a height map…

Height map:

A square gray scale bitmap. The darker the color the lower is the terrain. The example shows a scale reduced height map. Normally, you have height maps with a size from 512x512 to 4096x4096 pixels. The bigger your height map is the more detailed / the bigger is your level.

Hint: The level of detail does not only depend on the size of the height map bitmap but also on the number of triangles which will be set later in our modeling tool.

The best way to create a height map is to use a tool like Earth Sculptor. You can directly see your results and adjust your level on the fly. Modern level editors (i.e. Unreal Editor) include a height map editor.

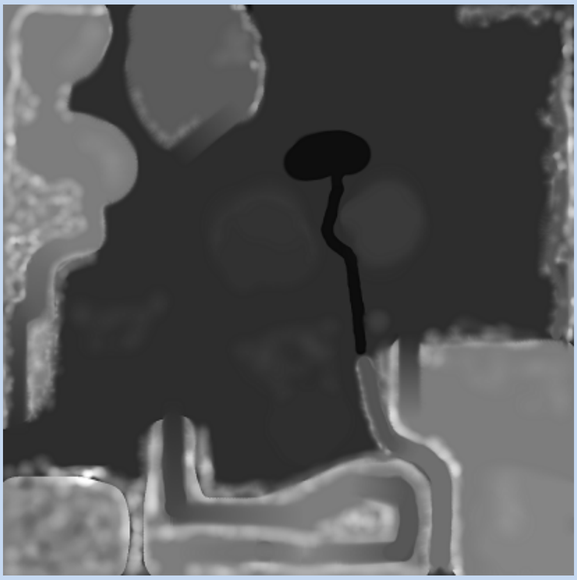


Illustration 1: Height map

… and textured by either a color map of a mask map – shader – textures combination. Let’s get to know these two possibilities:

1 One simple Color map:

A color map is an image file which represents the texture of the terrain. The **advantage** of a color map is that you see what your level looks like and you even see the shadows if your terrain editor supports the generation of shadows. Furthermore you can paint your terrain as you like. The shader method we use only supports three different textures. The color map approach needs no programming at all.
The **disadvantage** is that your color map gets extremely huge if your terrain is big (>30mb). Even with this size you will have washy textures which do not look good at all. So color maps should only be used for small terrains or for low quality / low performance games.

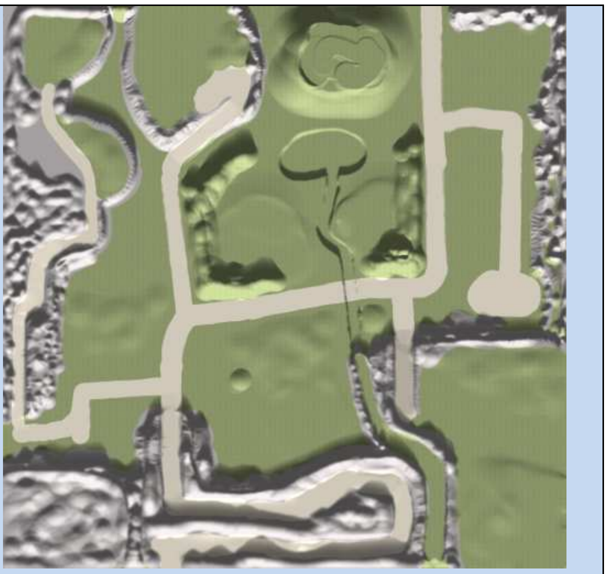


Illustration 2: Color map

2 Mask map, shader and textures:

This approach is not only based on design but also contains a shader which means: coding! Fortunately, BoH_Havoc took this part for us and designed a nice small shader to be used with Lite-C.

To use this shader we need three things:

1. The shader
2. A mask map
3. Textures

The mask contains three colors which are replaced by the textures while the game is running. Illustration 3 shows an example mask map. The red color will be replaced with a grass texture, the blue color with a sand texture and the green color with a rock texture.

The three textures will be put into one DDS file (A quad texture = an image consisting of 4 smaller images) with mip maps enabled. A mip map is a series of images with the same motive but with decreasing details.

The shader will later on take each part of the texture file and replace the three colors in the mask map by the corresponding texture.

Hint: The shader from Shade-C uses some more maps (Color map, normal + specular map of the color map) but this is *theoretically* unnecessary.

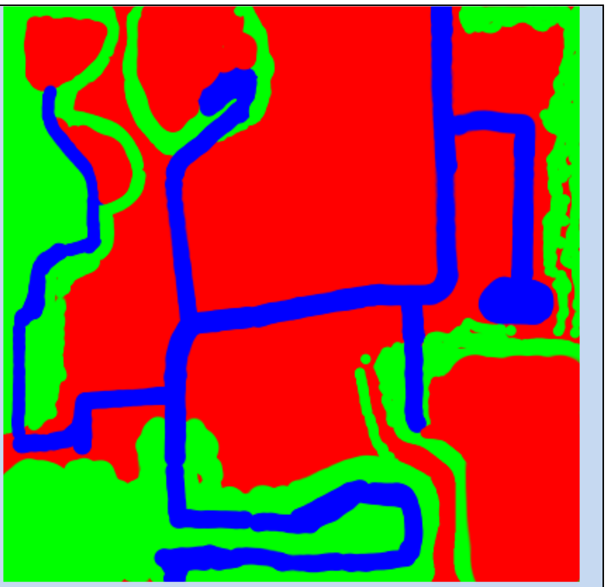


Illustration 3: Mask map

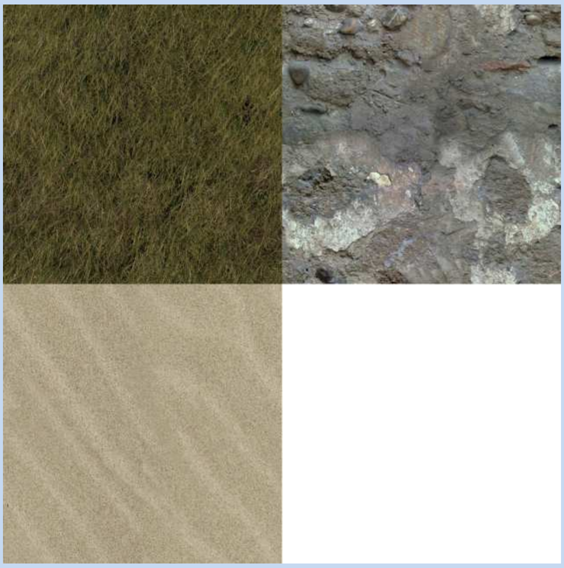


Illustration 4: Quad - Texture

We do not want to make it too easy so we pick possibility number two. Okay, the real reason is because the result looks much better, we do not need much space because we have no huge color maps and we are more flexible and can change the texture of the terrain during runtime.

In the next chapter we discuss the tools we need.

2 Tools

The tools which are marked optional are not necessary to create a terrain but are necessary for this tutorial.

- **Shade-C** – Shader library (http://shadec.project-havoc.com/)
- **Lite-C** - Commercial or Processional (http://www. gamestudio.de/)
- Optional**: The Gimp** – Image manipulation tool to create mip maps (http://www.gimp.org/)
- Optional**: The Gimp DDS plugin** – Plugin to save DDS files (http://code.google.com/p/gimp-dds/)

- **Optional**: **Earth Sculptor** – WYSIWYG terrain editor (http://www.earthsculptor.com/)
- **Optional**: **ShaderMap Pro** – Tool to create normal and specular maps (http://shadermap.com/)

  Hint: You can also use GIMP and look for the normal map plugin! Here is a tutorial to create specular maps with GIMP. So if you want you do not need to buy ShaderMap Pro: http://www.modwiki.net/wiki/Start_a_Specular_map_with_a_Normal_map

3 Creating the terrain

I do not want to go into detail of the usage of Earth Sculptor. There exist a lot of tutorials by the way, a pretty good one can be found:

http://www.reallusion.com/iclone/icnb_tutorial.asp

1. What we do is to create our terrain with heights and depths.
2. Then we create three images: One with the color red, one with the color blue and one with the color green (chose a size you want) and save them as *.tga. Now, we use these three files as detail textures. The result will look somehow similar like that:

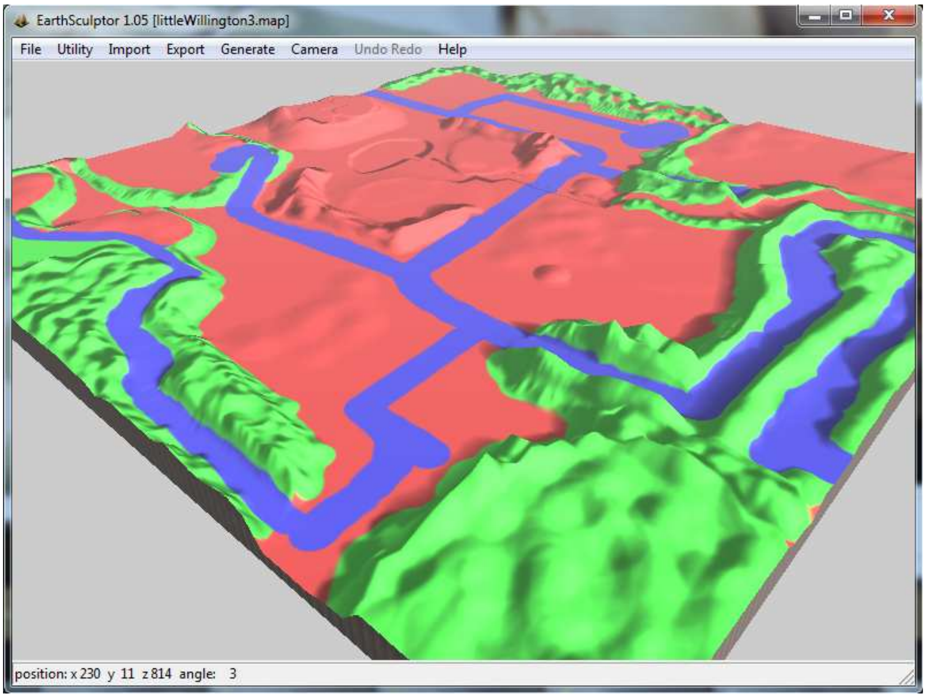

EarthSculptor 1.05 [littleWillington3.map]
File
Utility
Import
Export
Generate
Camera
Undo
Redo
Help
position: x 230 y 11 z 814 angle: 3

Illustration 5: Terrain with detail texture

3. Now export the mask map via: “Export → Terrain Textures” and select:

- Hardware Renderer → Check
- Size → 512x512
- Detail → Check
- Color → Not checked
- Light → Not checked

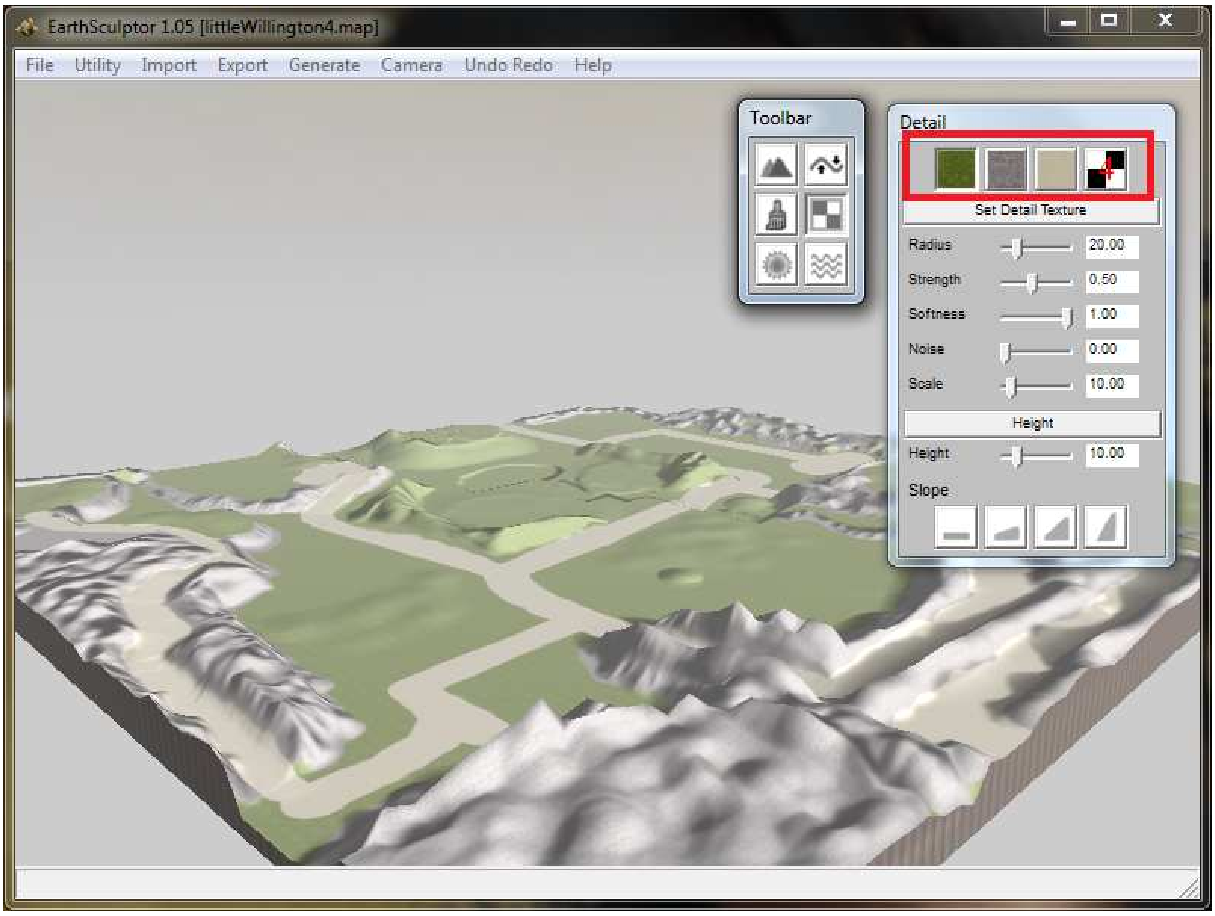
Save this image as “mask.png”. The result should look like illustration 3.

4. Now we export the shadow map via “Export → Terrain Textures”. We select:

- Hardware Renderer → Check
- Size → 512x512
- Details → Not checked
- Color → Not checked
- Light → Checked

Save the image as “shadow.png”.

5. Now we need the color map to satisfy the shader. Therefore, replace the red, green and blue texture with the real textures you want to see in your game. This is pretty simple, you do not need to paint all your map again. Just replace the detail textures by clicking “Set Detail Texture”.



Radius 20.00
Strength 0.50
Softness 1.00
Noise 0.00
Scale 10.00
Height 10.00

Illustration 6: Changing the detail texture to create the color map

Click “Export → Terrain Textures”. Set all check boxes and pick a size of 2048x2048. Save it as “texturedmap.png”.

6. The last export we need is our height map. Therefore select “Export → Heightmap Image” and save it as “heightmap.png”.

Now you should have a folder containing seven files (The screenshot shows some other file types, but that should not bother you):

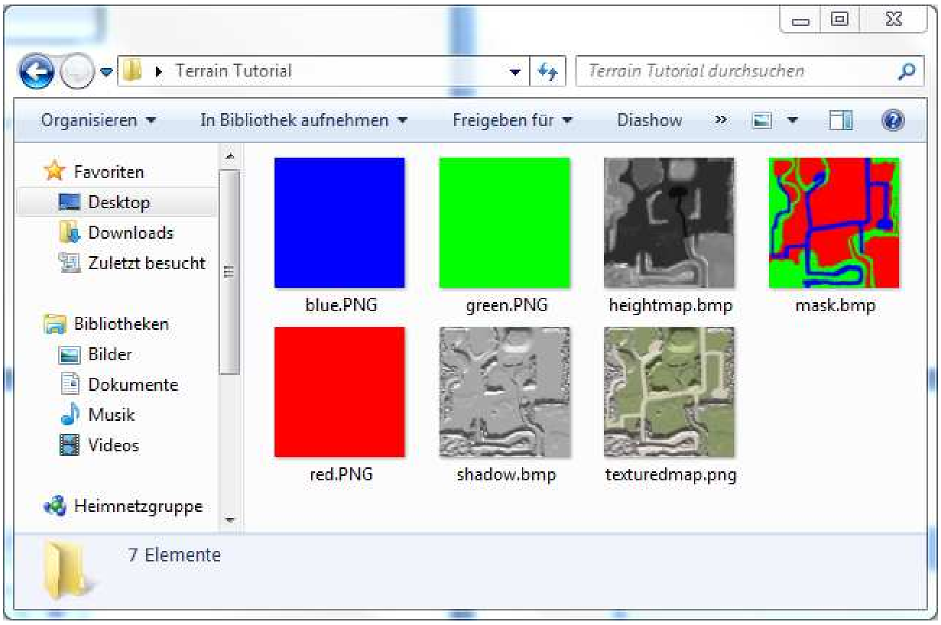


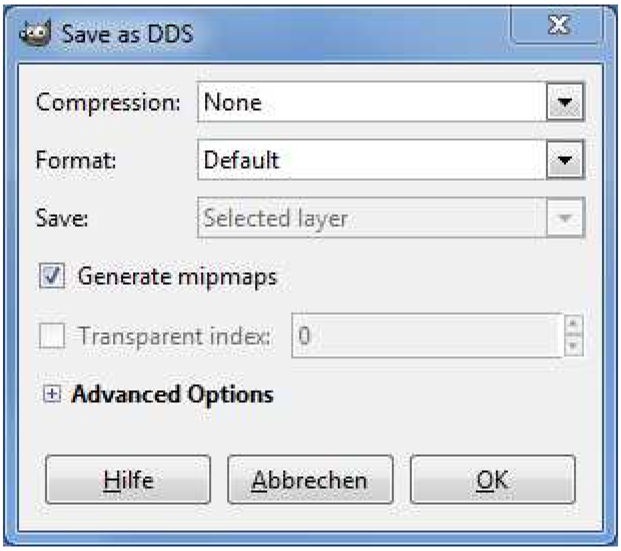
Illustration 7: Exported files

Close Earth Sculptor. We are done here.

4 Creating the textures

First of all, pick three textures that will replace your red, green and blue color from your "mask.png". Each one should have a size of 256x256 pixels. Put them together in a rectangle to form one image as show in illustration 4. Save your texture file as "textures.png".

Now open "GIMP" and load the textures file. Save it as "textures.dds" and check "Generate Mip Maps":


Save as DDS

Compression: None

Format: Default

Save: Selected layer

☑ Generate mipmaps

☐ Transparent index: 0

Illustration 8: Save textures file with mip maps

Do not close GIMP, we will use it again in step 6.

5 Creating normal and specular maps

A new technique to let textures appear more realistic regarding lightning, shadows and surfaces is allowed by the use of normal maps and specular maps.

Normal maps:

Normal mapping is a technique in 3D design to increase the level of detail of a 3D model without increasing the number of polygons. What normal maps basically do is to take the orientation of the surface normals from a detailed model and transfers it to an image so that it can be adapted to the surface of a model with lower details.

Hint: If you use quad textures it might come to these dark lines between the 4 fields. You should avoid that by generating a normal map for each texture and put it together to a quad texture manually.

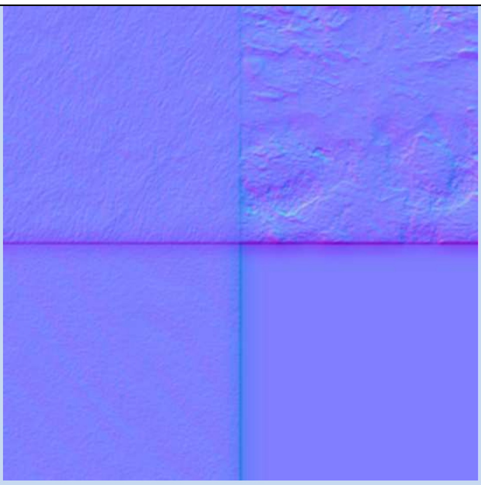


Illustration 9: Normal map

Specular maps:

A specular map is responsible for the reflectivity of our textures. Simple, isn't it?

Hint: Here holds the same as for the normal maps. If you use quad textures it might come to these dark lines between the 4 fields. You should avoid that by generating a specular map for each texture and put it together to a quad texture manually.

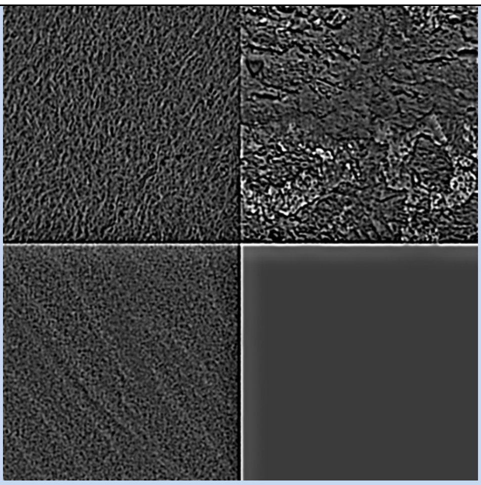


Illustration 10: Specular map

So how do we create those maps? I have chosen a great tool named ShaderMap Pro which is not free but only costs 20$. This tool has not only the advantage that it generates normal, specular and many more maps, it also displays the result in 3D so that you can see the effect your maps have on your texture.

1. Start Shader Map Pro, click "Load Diffuse" and load "textures.png".
2. Make sure that "Enable" and "Render" are only checked for "Normal Map" and "Specular Map" on the left side.
3. Click "Generate All Maps".

Repeat step 1-3 for “texturedmap.png” because we need specular and normal maps for the color map as well.

Copy the generated files (which are in the same directory as “textures.png” and/or “texturedmap.png”) to your working directory and rename “textures_NORM.tga” to “normal.tga”, “textures_SPEC.bmp” to “specular.bmp”, “texturedmaps_NORM.tga” to “texturednormap.tga” and “texturedmaps_SPEC.bmp” to “texturedspecular.bmp”.

Your directory should look like that:

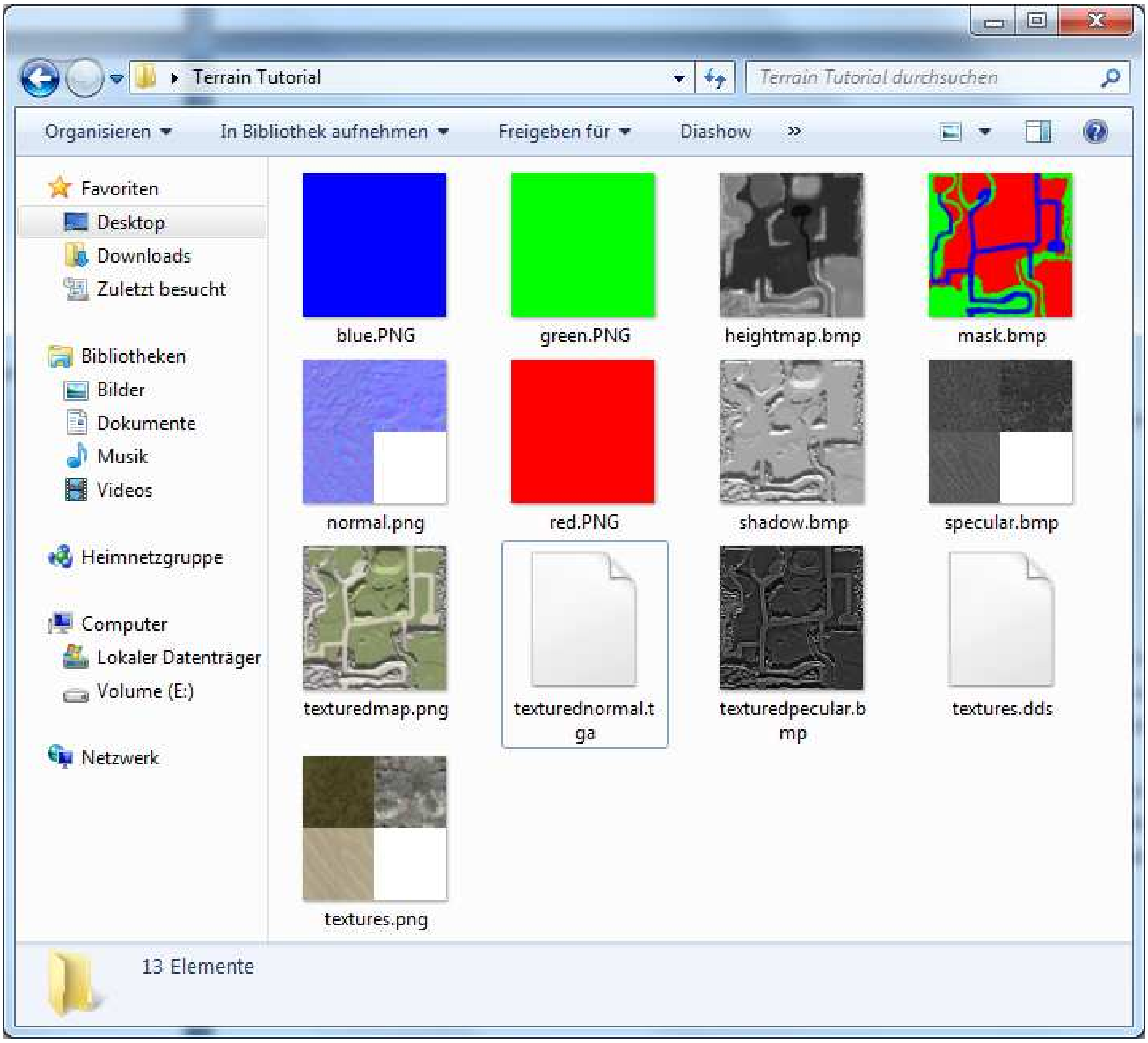


Illustration 11: Our files so far

Once again, other file formats should not bother you.

6 Put our textures together

Now we have everything we need, time to put the files in the format that Shade-C needs. Therefore, we open GIMP again and load “normal.tga”.

1. Click “Layer → Transparency → Add Alpha Channel…”
2. Click “Layer → Mask → Add Layer Mask…” (Initialize with the layer’s alpha channel)
3. Click “Layer → Mask → Show Layer Mask”

4. Open “specular.bmp” in a new window and copy past it into the alpha channel.
5. Click “Layer → Anchor layer”
6. Now export the image as “normalspec.tga” without RLE compression.

Load “mask.png” and do exactly the same but this time add “shadow.png” to the alpha channel. Export the new image as “maskshadow.tga”, again without RLE compression.

Load “texturednormal.png” and do exactly the same but this time add “texturedpecular.bmp” to the alpha channel. Export the new image as “texturednormalspec.tga”, again without RLE compression.

Now we need to convert all PNG files to TGA files because Gamestudio’s Model Editor is not able to read PNG files.

If you have done that close GIMP, we are done here.

7 Create the terrain in the Gamestudio Model Editor

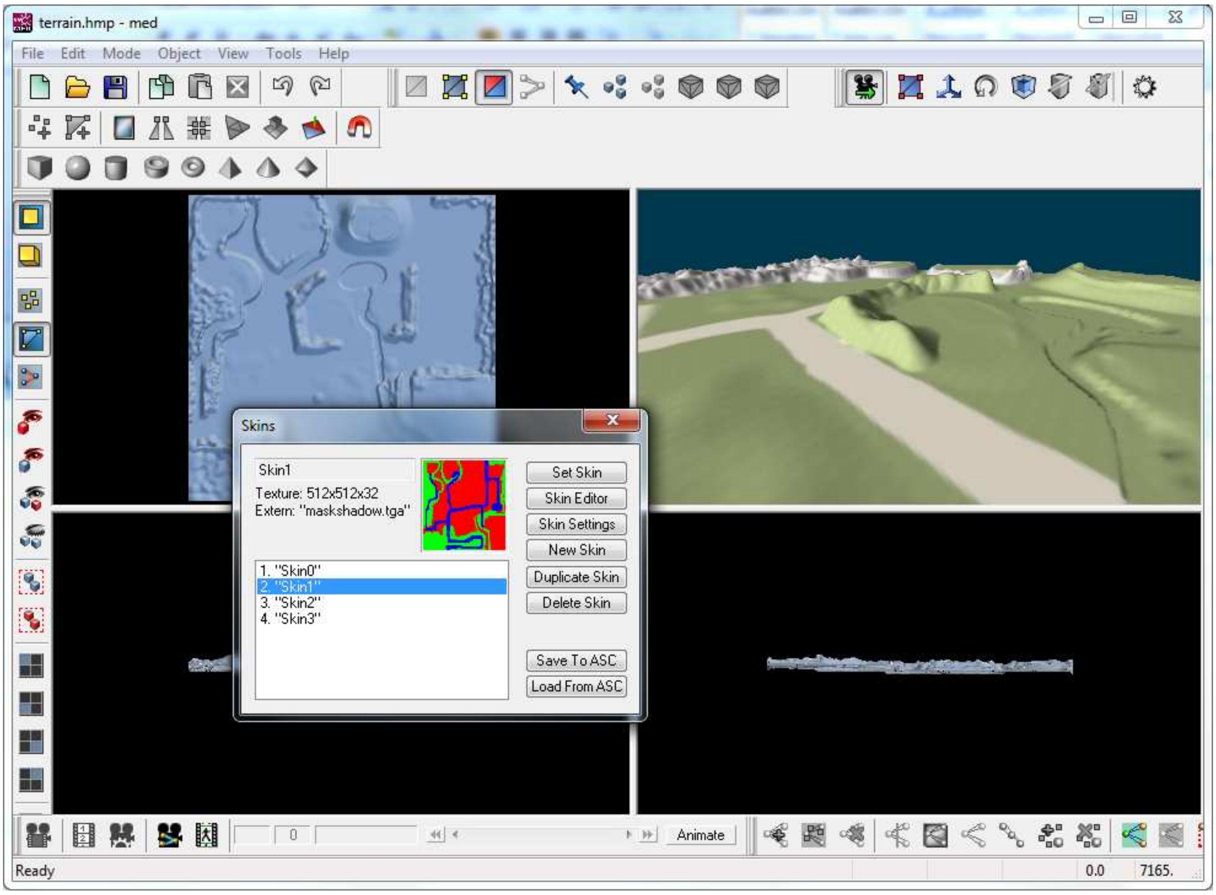
At first, we import our height map to create our basic model:

1. Open MED and click “Import → Terrain from Image…” and select “heightmap.bmp”.
2. Select for “Vertices horizontal” and “Vertices vertical” a number of 256.
3. The z position should be calculated by this formula: R*20 + G*20 + B*20
4. Create your terrain.

Now we add the skins we created:

5. Click “Object → Manage Skins”.
6. At first we change the first skin which has been imported from our height map and select an external file. This is because the shader does not use the first skin and so we keep our terrain file slim. Therefore, click “Skin Settings”, check “Texture File → Extern” and select “texturedmap.bmp”. Close the “Skin Settings” dialog.
7. Click “New Skin”, check “Texture” and “Extern” and select the file “maskshadow.tga”.
8. Click “New Skin”, check “Texture” and “Extern” and select the file “texturednormalspec.tga”.
9. Click “New Skin”, check “Texture” and “Extern” and select the file “normalspec.tga”

Your window should look like the following illustration:



Skin1
Texture: 512x512x32
Extern: "maskshadow.tga"

1. "Skin0"
2. "Skin1"
3. "Skin2"
4. "Skin3"

Illustration 12: The terrain in MED

Save the terrain as “terrain.hmp” and close MED. Time for some programming.

8 Write the code

The last part is simple:

1. Create a new folder in your Gamestudio project’s folder.
2. Create a new file and name it “main.c”. Now add the following lines of code:

```
#define PRAGMA_PATH "shade-c"

#include <acknex.h>
#include <default.c>
#include "sc_core.c";

BMAP* myTerrainColor = "textures.dds";

MATERIAL* mtl_myTerrain =
{
   skin1 = myTerrainColor;
   effect = "sc_obj_terrain.fx";
}

void main() {
   level_load("terrain.hmp");
   wait(4);
   video_switch(8,32,0);
```

```
    sc_setup();
    level_ent.material = mtl_myTerrain;
}
```

Illustration 13: Code to use the shader

The code is very easy.

- We create a bitmap where we load our textures for the map.
- We create a material with the sc_obj_terrain shader from Shade-C and add our texture bitmap to the material via the skin1 variable.
- We load our height map as level.
- We initialize Shade-C via “sc_setup();”
- We assign our material to the level_ent (Which is a global pointer to the current level entity → manual)

3. Add the folder “shade-c” to your project directory.
4. Run it! It should work and look good ☺

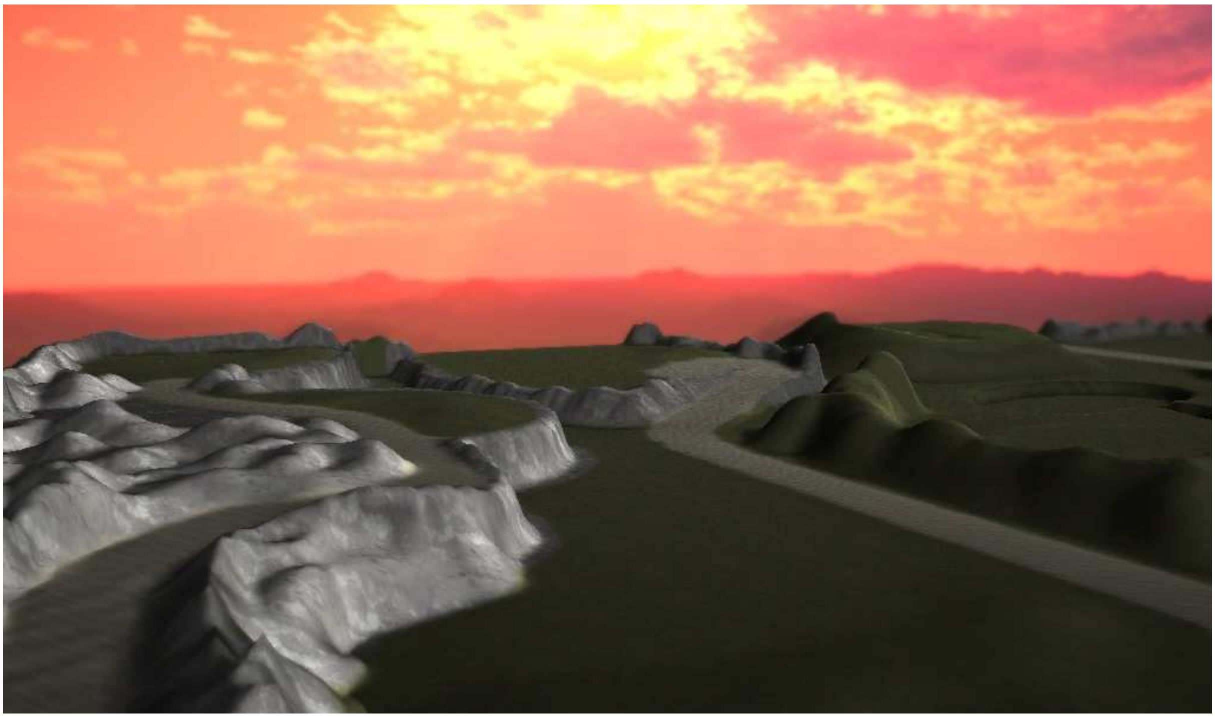


Illustration 14: The terrain in action (with a nice sky cube)

Thanks to BoH_Havok for his great shader collection!