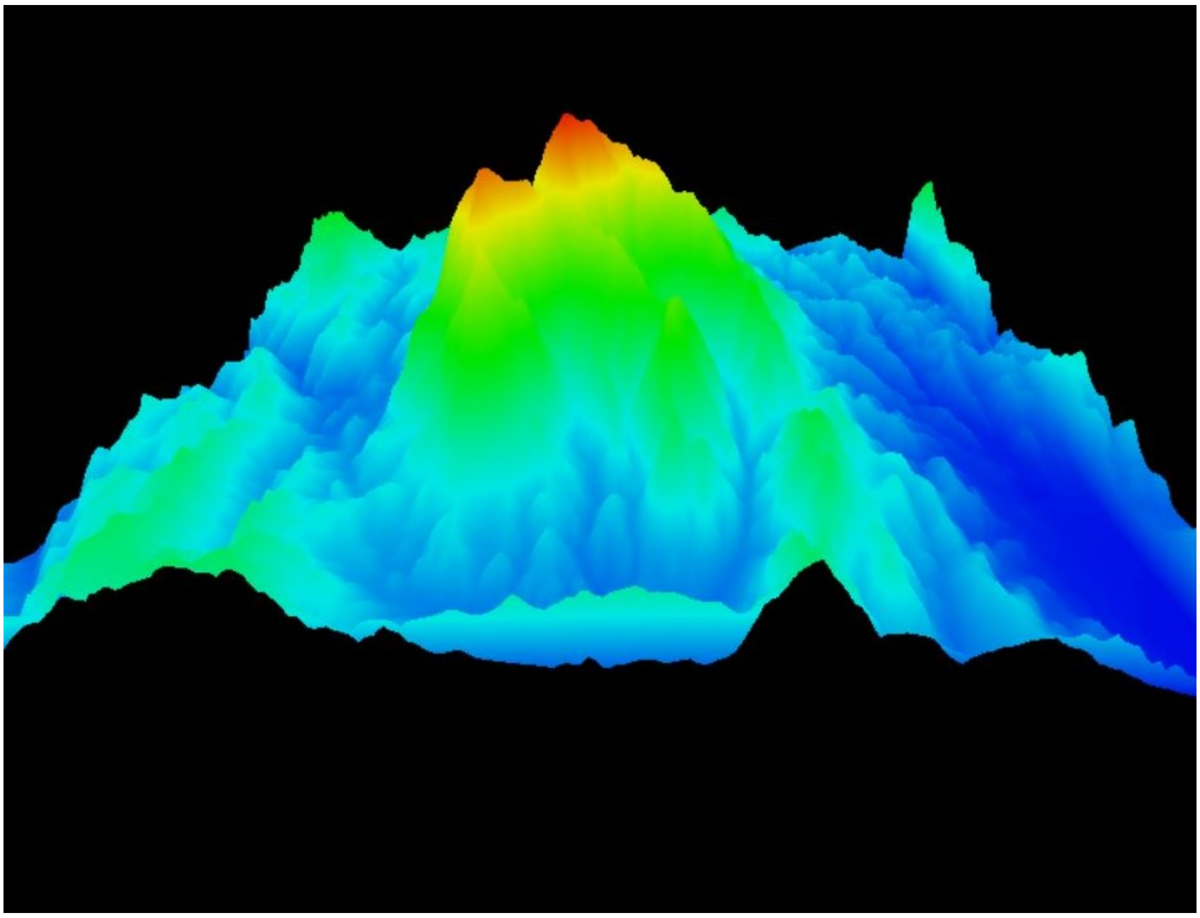
CS 557 Final Project

Terrain Mapping Using Height Maps and GLman

Margaux Masson - massonm@oregonstate.edu



Oregon State University

Computer Science

Winter 2019

Table of Contents

Chapter 1: **Introduction**

Introduction to Computer Graphics Final Project

This final project has for main purpose to apply all the techniques learned during this term. Therefore, I chose to implement a terrain mapping simulation using height maps and glman. This project involves implementing and manipulating 3D surfaces and normals, texturing, lighting, shaders and more.

In this report, I will expose the different steps which led me to implement this simulation, the issues I encountered and the solutions that I chose. Plus, some code explanation and some pictures to illustrate this report.

Chapter 2: **Project**

2.1 Reminder of the initial proposal

Data visualization - Height map For the final project, I would like to implement a height-map with colors corresponding to the "altitude" of each points with as input, data of the heights.

I want to use a gridded surface which will be the base of the height map. Then I would like to apply the elevation z of the terrain/heights to the different points (x,y) of the surface according to some data (csv file for example). So I would be able to use it as a terrain visualization or any data that can be represent with a height map.

2.2 Differences between proposal and actual project

After reading several books and articles about it, I realized that there was a much more efficient way to implement this solution. Indeed, using the depth/height maps and using in the proper way the shaders and glman, I could get a very interesting result requiring much less computation. So I decided to implement this solution instead of creating a gridded surface and apply all the heights to each points.

2.3 Project Timeline

2.3.1 Learning more about the concept of depth/height maps

The first step of this project was to understand better the concept of depth/height maps and how I could apply and use it in glman. I wanted to be able to modify the elevation in real time using the sliders so I needed to use glman and thought that I could use the height map as a texture which will then "shape" the flat quad into the terrain corresponding to the height map. I also wanted to display the terrain with a color level according to the elevation. For instance, the dark blue color corresponds to terrain at the lowest level, while the dark red color/pink corresponds to the highest height.

In order to implement all parts of this idea, I started by shaping the surface in the vertex shader, and then I applied the coloring in the fragment shader.

2.3.2 Vertex shader

The main goal here was to be able to pass the coordinates from the 2D texture, which is the height map, into the vertex and then shape the quad according to these heights which is done thanks to these lines:

```
vST = gl_MultiTexCoord0.st;
vec3 vert = gl_Vertex.xyz;
vert.z = texture2D(depth, vST).r * uMultiplier;
Z = vert.z;
```

Listing 2.1: creation of the flat surface

The variable uMultiplier is an uniform variable that can be modified by the user through the slider and which allows the user to change the elevation level.

However, it is also important to implement the right lighting. Here is the entire code for the vertex shader:

```
vST = gl_MultiTexCoord0.st;
vec3 vert = gl_Vertex.xyz;
vert.z = texture2D(depth, vST).r * uMultiplier;
vec4 ECposition = gl_ModelViewMatrix * gl_Vertex;
vec3 aLocalNormal = gl_Normal;
vN = normalize( gl_NormalMatrix * aLocalNormal ); // normal vector
vN.z = vN.z + vert.z;
vL = LIGHTPOSITION - ECposition.xyz; // vector from the point
// to the light position
vE = vec3( 0., 0., 0. ) - ECposition.xyz; // vector from the point
// to the eye position
gl_Position = gl_ModelViewProjectionMatrix * vec4(vert,1);
```

Listing 2.2: creation of the flat surface

For instance, for this height map:

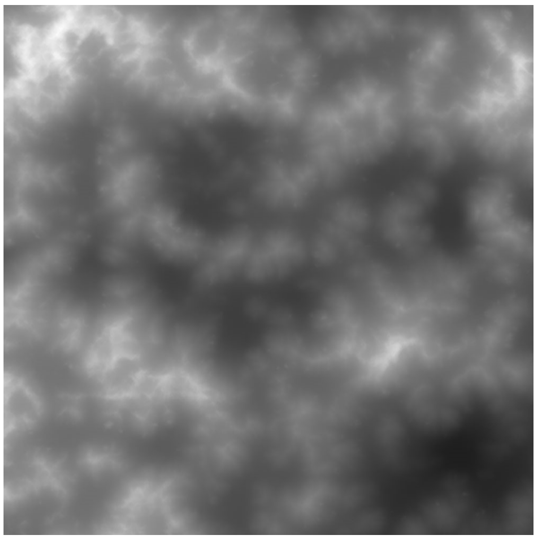


Figure 2.1: Height map of a terrain

The results with two different values of uMultiplier are:

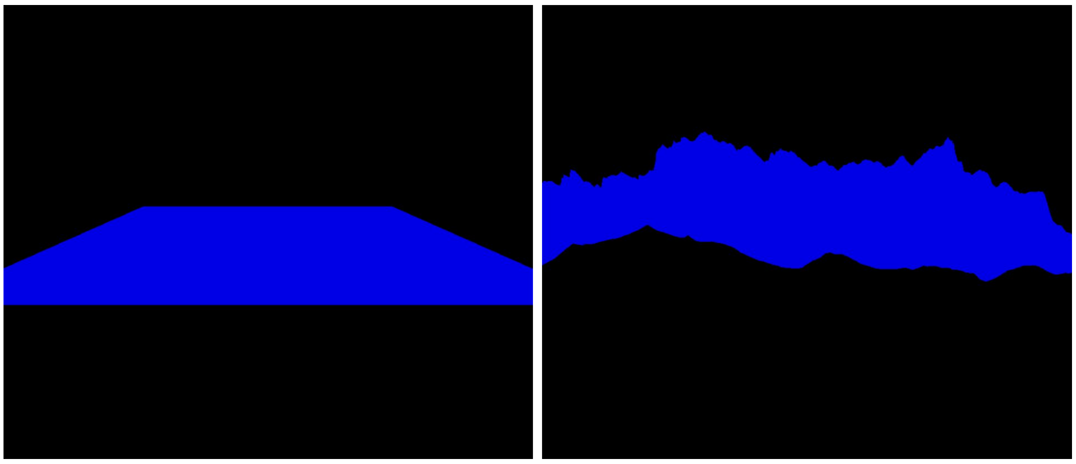


Figure 2.2: Terrain mapping using the previous height map with two different values of uMultiplier

2.3.3 Fragment shader

After implementing the deformation of the surface according to the height map passed through the texture, the next goal was to apply the right coloring according to the elevation of each points.

To do so, I used the function vec3 ChromaDepth(float t) from a previous project:

```
vec3
ChromaDepth( float t )
{
  t = clamp( t, 0., 1. );

  float b = 1.;
  float g = 0.0;
  float r = 1. - 6. * ( t - (5./6.) );

  if( t <= (5./6.) )
  {
    b = 6. * ( t - (4./6.) );
    g = 0.;
    r = 1.;
  }

  if( t <= (4./6.) )
  {
    b = 0.;
    g = 1. - 6. * ( t - (3./6.) );
    r = 1.;
  }

  if( t <= (3./6.) )
  {
    b = 0.;
    g = 1.;
    r = 6. * ( t - (2./6.) );
  }

  if( t <= (2./6.) )
  {
    b = 1. - 6. * ( t - (1./6.) );
    g = 1.;
    r = 0.;
  }

  if( t <= (1./6.) )
  {
    b = 1.;
    g = 6. * t;
  }

  return vec3( r, g, b );
}
```
Listing 2.3: Function used to create the grass field

And in order to get the elevation live even when the user is modifying the variable uMultiplier, I needed to get the Z from the vertex shader. So I added this line in the vertex shader:

```
Z = vert.z;
```
Listing 2.4: Function used to create the grass field

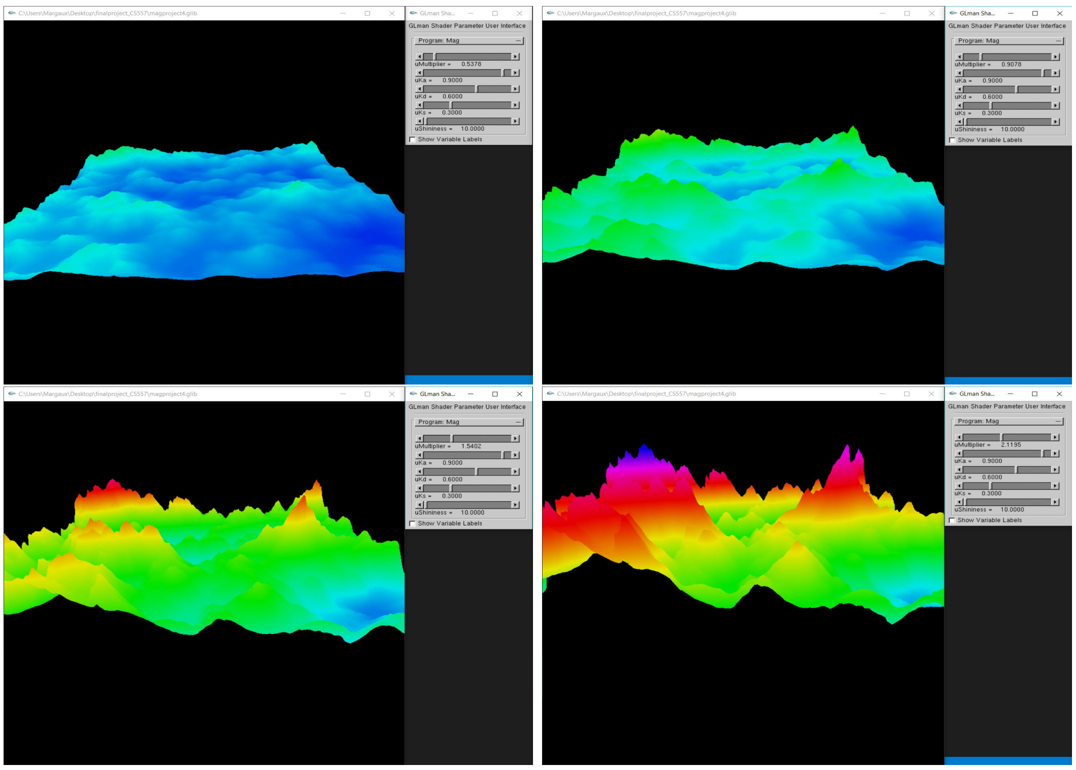
With Z an out float variable. Then in the fragment, I get back this value (which is an in float Z;) and use as following:

```
    vec3 myColor = ChromaDepth(Z/2);

  vec3 Normal = normalize(vN);
  vec3 Light = normalize(vL);
  vec3 Eye = normalize(vE);
  vec3 ambient = uKa * myColor;
  float d = max( dot(Normal,Light), 0. ); // only do diffuse if the light can
    see the point
  vec3 diffuse = uKd * d * myColor;
  float s = 0.;
  if( dot(Normal,Light) > 0. ) // only do specular if the light can see the
    point
  {
    vec3 ref = normalize( reflect( -Light, Normal ) );
    s = pow( max( dot(Eye,ref),0. ), uShininess );
  }
  vec3 specular = uKs * s * vec3(1,1,1);

  gl_FragColor = vec4( ambient + diffuse + specular, 1. );
```
Listing 2.5: Function used to create the grass field

Now the surface is shaped according to the height map given in input and colored according to the elevation level:



uMultiplier = 0.5378
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

uMultiplier = 0.9078
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

uMultiplier = 1.5402
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

uMultiplier = 2.1195
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

Figure 2.3: Terrain mapping with coloring using the previous height map with four different values of uMultiplier

Chapter 3: **Conclusion**

3.1 What I learned

Through this project, I learned how to combined together some projects done during the term, for instance I re used the texture implementation and changed in order to do what I wanted inside the vertex shader. I also re used the function ChromaDepth in order to color-map the elevation of the terrain.

Moreover, I learn more about the height/depth maps and how to use them. This project can also be used to transpose other objects than terrains into 3D.

3.2 Useful Links

3.2.1 Github repository

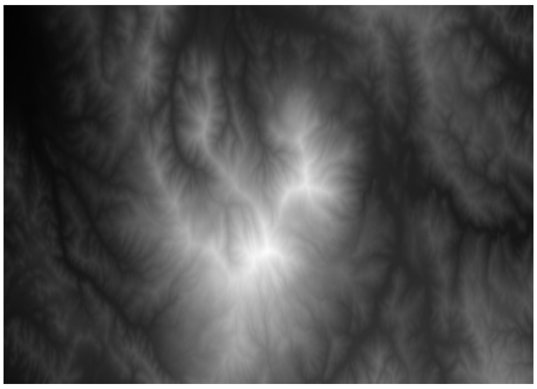
You can find the code for this project: `https://github.com/MargauxMasson/Terrain_Mapping_Using_Height_Maps_GLman`

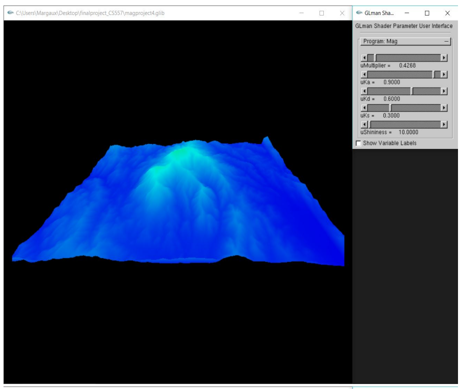
Clone the repository and use glman to execute the glib file.

3.2.2 Video

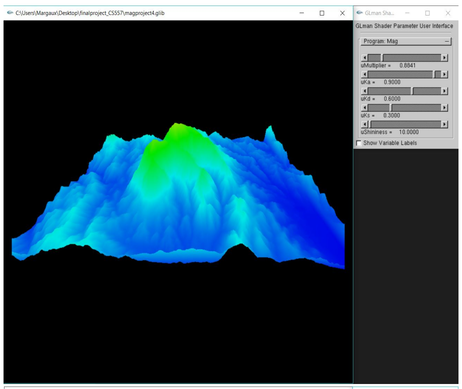
Link to the video `https://media.oregonstate.edu/media/t/0_v1cwn7ka`

3.3 Images: height map and results corresponding

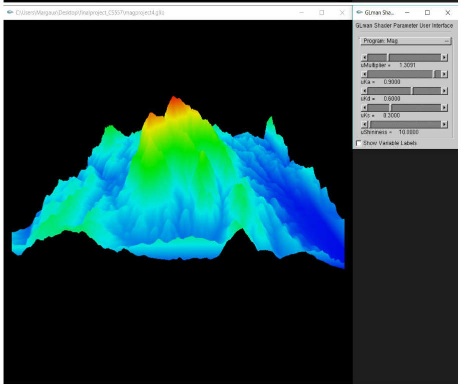




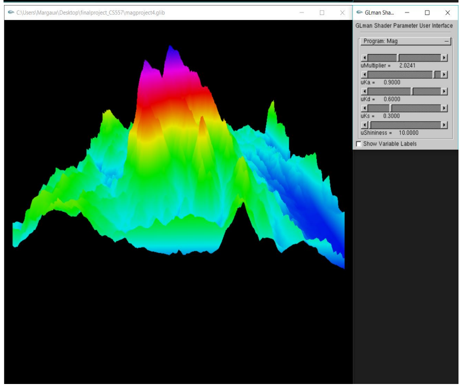
```
C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Shader Parameter User Interface
Program: Mag
uMultiplier =    0.4268
uKa =    0.9000
uKd =    0.6000
uKs =    0.3000
uShininess =    10.0000
Show Variable Labels
```



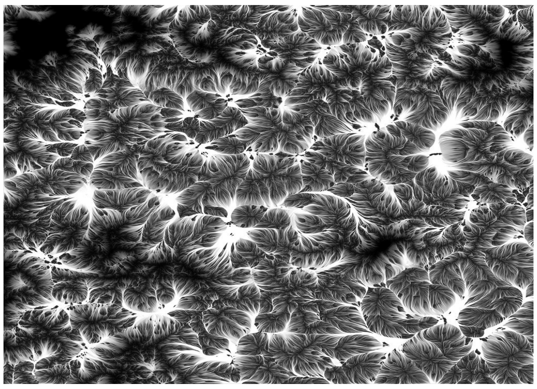
```
C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Shader Parameter User Interface
Program: Mag
uMultiplier =    0.8841
uKa =    0.9000
uKd =    0.6000
uKs =    0.3000
uShininess =    10.0000
Show Variable Labels
```

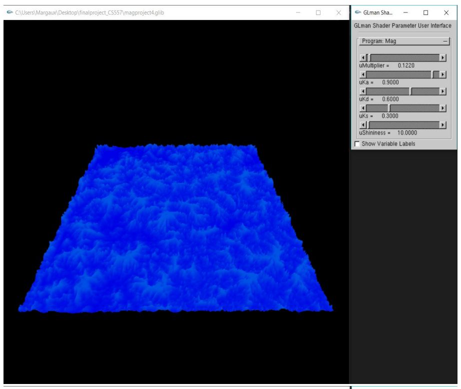


```
C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Shader Parameter User Interface
Program: Mag
uMultiplier =    1.3091
uKa =    0.9000
uKd =    0.6000
uKs =    0.3000
uShininess =    10.0000
Show Variable Labels
```



```
C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Shader Parameter User Interface
Program: Mag
uMultiplier =    2.0241
uKa =    0.9000
uKd =    0.6000
uKs =    0.3000
uShininess =    10.0000
Show Variable Labels
```



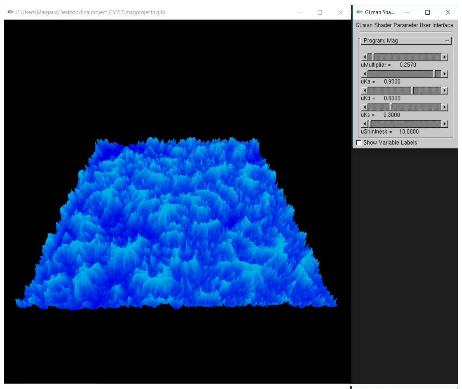


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib

GLman Shader Parameter User Interface

Program: Mag

uMultiplier = 0.1220
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

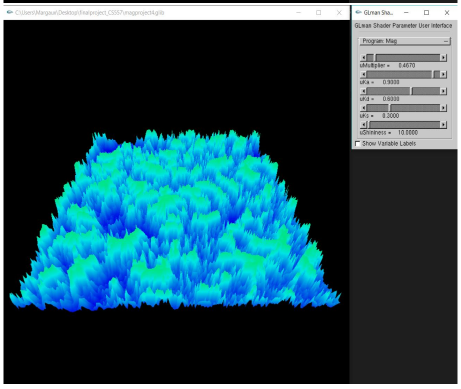


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib

GLman Shader Parameter User Interface

Program: Mag

uMultiplier = 0.2570
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

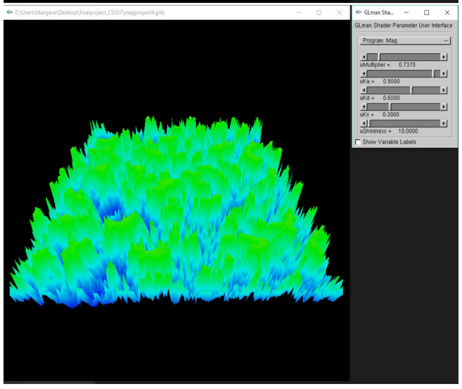


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib

GLman Shader Parameter User Interface

Program: Mag

uMultiplier = 0.4670
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

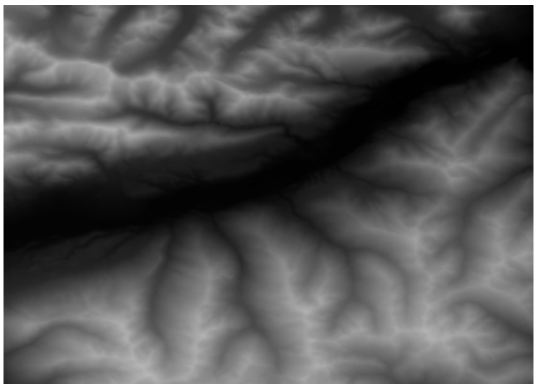


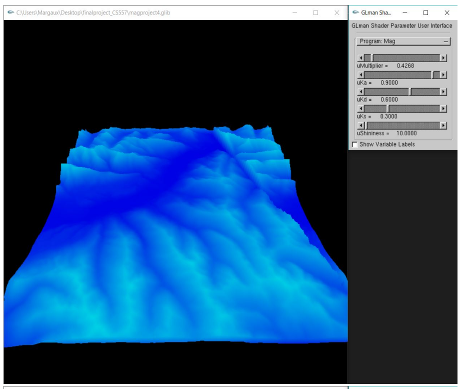
C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib

GLman Shader Parameter User Interface

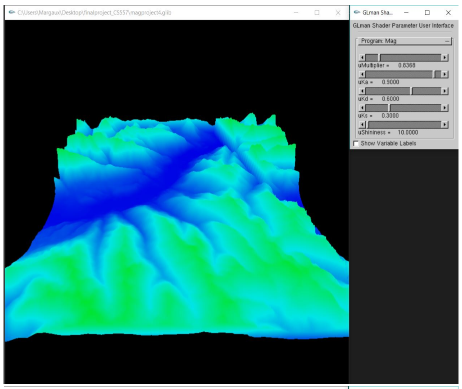
Program: Mag

uMultiplier = 0.7370
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000

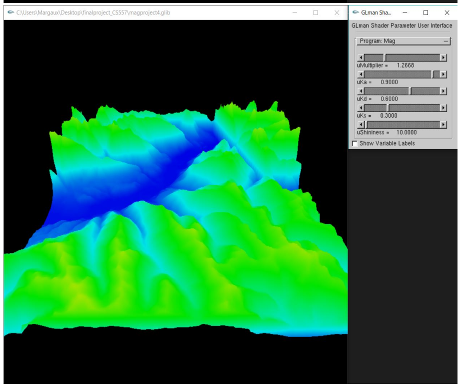



C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Sha...
GLman Shader Parameter User Interface
Program: Mag
Show Variable Labels

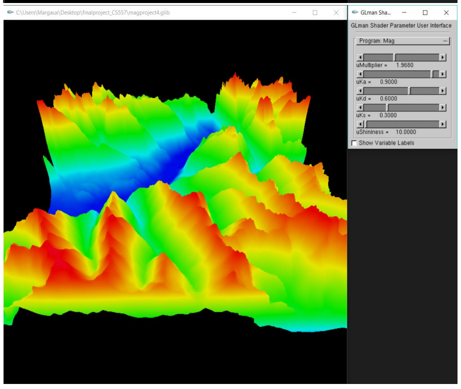
uMultiplier = 0.4268
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Sha...
GLman Shader Parameter User Interface
Program: Mag
Show Variable Labels

uMultiplier = 0.8368
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Sha...
GLman Shader Parameter User Interface
Program: Mag
Show Variable Labels

uMultiplier = 1.2668
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000


C:\Users\Margaux\Desktop\finalproject_CS557\magproject4.glib
GLman Sha...
GLman Shader Parameter User Interface
Program: Mag
Show Variable Labels

uMultiplier = 1.9680
uKa = 0.9000
uKd = 0.6000
uKs = 0.3000
uShininess = 10.0000