CS 557 -- Winter Quarter 2016

Project #7 Report

A Tessellated Parametric Triangular Bézier Patch

Name: Kai Shi

ID: 932-504-512

In project7, I Draw a Bézier triangle surface patch and tessellate the triangle with a tessellation shader. Each edge of triangle surface can change the z coordinate and under checkbox I can choose whether outer and inner densities change according to the z.

(1) Code

glib:

```
##OpenGL GLIB
Perspective 70
LookAt  2 -1 1  0 2 -1  0 0 1

Vertex        mtriangle.vert
Fragment      mtriangle.frag
TessControl   mtriangle.tcs
TessEvaluation mtriangle.tes
Geometry      mtriangle.geom
Program MidTriangle                     \
                uOuter01 <3 5 50> \
                uOuter12 <3 5 50> \
                uOuter20 <3 5 50> \
                uInner <1 10 50> \
                uZ01 <-2 0 2> \
                uZ12 <-2 0 2> \
                uZ20 <-2 0 2> \
                uAdaptToZs <false> \
                uShrink <0. 0.9 1.> \
                uKa <0. 0.1 1.0> \
                uKd <0. 0.7 1.0> \
                uKs <0. 0.2 1.0> \
                uShininess <3. 10. 1000.> \
                uLightX <-10. 0. 10.> uLightY <-10. 8. 10.> uLightZ <-10. 8. 10.>

Color 1. .5 0.
NumPatchVertices 3
glBegin gl_patches
  glVertex 0. 0. 0.
  glVertex 2. 0. 0.
  glVertex 0. 2. 0.
glEnd
```

vert:

```
void
main( )
{
	gl_Position = gl_Vertex;
}
```

tcs:

```
#version 400 compatibility
#extension GL_ARB_tessellation_shader : enable

uniform float uZ01, uZ12, uZ20;
uniform int uOuter01, uOuter12, uOuter20, uInner;
uniform bool uAdaptToZs;

layout( vertices = 3 ) out;

void
main( )
{
	gl_out[ gl_InvocationID ].gl_Position = gl_in[ gl_InvocationID ].gl_Position;
	if( uAdaptToZs )
	{
		gl_TessLevelOuter[0] = float(uOuter12)+uZ12;
		gl_TessLevelOuter[1] = float(uOuter20)+uZ20;
		gl_TessLevelOuter[2] = float(uOuter01)+uZ01;
		gl_TessLevelInner[0] = gl_TessLevelInner[1] = float(uInner)
+uZ12+uZ20+uZ01;
	}
	else
	{
		gl_TessLevelOuter[0] = float(uOuter12);
		gl_TessLevelOuter[1] = float(uOuter20);
		gl_TessLevelOuter[2] = float(uOuter01);
		gl_TessLevelInner[0] = gl_TessLevelInner[1] = float(uInner);
	}
}
```

tes:

```
#version 400 compatibility
#extension GL_ARB_tessellation_shader : enable
layout( triangles, equal_spacing, ccw) in;

uniform float uZ01, uZ12, uZ20;

out vec3 teNormal;
out vec3 teECposition;

void
main( )
{
	vec4 p0 = gl_in[0].gl_Position;
	vec4 p1 = gl_in[1].gl_Position;
	vec4 p2 = gl_in[2].gl_Position;
	vec4 p3 = gl_in[3].gl_Position;

	vec4 p01 = vec4 ((p0.x+p1.x)/2,(p0.y+p1.y)/2,uZ01,1);
	vec4 p12 = vec4 ((p1.x+p2.x)/2,(p2.y+p1.y)/2,uZ12,1);
	vec4 p20 = vec4 ((p2.x+p0.x)/2,(p0.y+p2.y)/2,uZ20,1);

	float u = gl_TessCoord.x;
	float v = gl_TessCoord.y;
	float w = gl_TessCoord.z;

	float b0 = u*u;
	float b1 = v*v;
	float b2 = w*w;
	float b01 =2*u*v ;
	float b12 = 2.*v*w;
	float b20 = 2.*w*u;

	float db0du = 2.*u;
	float db0dv = 0.;

	float db1du = 0.;
	float db1dv = 2.*v;
```

```
	float db2du = -2.*(1.-u-v);
	float db2dv = -2.*(1.-u-v);

	float db01du = 2.*v;
	float db01dv = 2.*u;

	float db12du = -2.*v;
	float db12dv = 2.*(1.-u-2.*v);

	float db20du = 2.*(1.-2.*u-v);
	float db20dv = -2.*u;

	teECposition = ( gl_ModelViewMatrix * ( b0*p0 + b01*p01 + b1*p1 + b12*p12
+ b2*p2 + b20*p20 ) ).xyz;
	gl_Position = vec4( teECposition, 1. );

	vec4 dpdu = db0du*p0 + db01du*p01 + db1du*p1 + db12du*p12 + db2du*p2 +
db20du*p20;
	vec4 dpdv = db0dv*p0 + db01dv*p01 + db1dv*p1 + db12dv*p12 + db2dv*p2 +
db20dv*p20;

	teNormal = gl_NormalMatrix * normalize( cross( dpdu.xyz, dpdv.xyz ) );
}
```

geom:

```
#version 400 compatibility

#extension GL_EXT_gpu_shader4: enable

#extension GL_EXT_geometry_shader4: enable

layout( triangles ) in;

layout( triangle_strip, max_vertices=32 ) out;

uniform float uShrink;

uniform float uLightX, uLightY, uLightZ;
```

```
in vec3 teECposition[3];
in vec3 teNormal[3];
out vec3 gNs;
out vec3 gLs;
out vec3 gEs;

vec3 LightPos = vec3( uLightX, uLightY, uLightZ );

vec3 V[3];

vec3 CG;

void

ProduceVertex( int v )

{

        gNs = teNormal[v];

        gLs = LightPos - teECposition[v];

        gEs = vec3(0.,0.,0.) - teECposition[v];

        gl_Position = gl_ProjectionMatrix * vec4( CG + uShrink * ( V[v] - CG ), 1. );

        EmitVertex( );

}

void

main( )

{

        V[0] = gl_PositionIn[0].xyz;
```

```
        V[1] = gl_PositionIn[1].xyz;

        V[2] = gl_PositionIn[2].xyz;

        CG = ( V[0] + V[1] + V[2] ) / 3.;

        ProduceVertex( 0 );
        ProduceVertex( 1 );
        ProduceVertex( 2 );

}
```

frag:

```
#version 400 compatibility
in vec3 gNs;
in vec3 gLs;
in vec3 gEs;

uniform float uKa, uKd, uKs;
uniform float uShininess;

void
main( )

{
        vec3 Normal;
        vec3 Light;
        vec3 Eye;

        vec4 uColor=vec4 (1.,0.5,0.,1.);

        Normal = normalize(gNs);
        Light = normalize(gLs);
        Eye = normalize(gEs);

        vec4 ambient = uKa * uColor;
        float d = max( dot(Normal,Light), 0. );
        vec4 diffuse = uKd * d * uColor;
```

```
	float s = 0.;
	if( dot(Normal,Light) > 0. )
	{
		vec3 ref = normalize( 2. * Normal * dot(Normal,Light) - Light );
		s = pow( max( dot(Eye,ref),0. ), uShininess);

	}
	vec4 specular = uKs * s * vec4 (1,1,1,1);

	gl_FragColor = vec4( ambient.rgb + diffuse.rgb + specular.rgb, 1. );

}
```

(2) What I did and Reasons

First step in the glib file use

NumPatchVertices 3

glBegin gl_patches

glVertex 0. 0. 0.

glVertex 2. 0. 0.

glVertex 0. 2. 0.

glEnd

to create a triangle surface with three points (0, 0, 0), (2, 0, 0,), (0, 2, 0).

Vertex shader is simple. Then, Tessellation Control Shader, I need to define when uAdaptToZs checkbox is true, gl_TessLevelOuter and gl_TessLevelInner are what and if checkbox is false, what are they. When checkbox is false, gl_TessLevelOuter[] are simply equal to uOuter12, 20, 01. gl_TessLevelInner[] are uInner. However, when

checkbox is true, which means the Outerlevel should relates to uZ value. So, I add uZ value at the end of uOuter, and then pass it to the gl_TessLevelOuter[], such as: gl_TessLevelOuter[0] = float(uOuter12)+uZ12;

For the Innerlevel, each edge's uZ change will affect the inner densities, so, three uZ value need to be added: gl_TessLevelInner[0] = gl_TessLevelInner[1] = float(uInner)+uZ12+uZ20+uZ01;

Next stage is Tessellation Evaluation Shader. The things I need to decide is the position of p01, p12, p20. For p01, when uZ value s not involved, the coordinate of p01 should be (p0+p1)/2. But uZ needs to be considered, so, p01's x,y coordinates still be (p0+p1)/2, but p01's z coordinate should equal to uZ01. The same operation on p12 and p20. Such as:

vec4 p01 = vec4 ((p0.x+p1.x)/2,(p0.y+p1.y)/2, uZ01, 1);

vec4 p12 = vec4 ((p1.x+p2.x)/2,(p2.y+p1.y)/2, uZ12, 1);

vec4 p20 = vec4 ((p2.x+p0.x)/2,(p0.y+p2.y)/2, uZ20, 1);

Then in Geometry Shader, using CG + uShrink * (V[v] - CG) to make three points of every triangle come closer to Centroid ("CG"). (Actually, professor has given this part.)

Last part is fragment shader. I set the lighting in this stage. Do ambient, diffuse, and specular lighting and only do specular if the light can see the point. Last passing vec4(ambient.rgb + diffuse.rgb + specular.rgb, 1.) to the gl_FragColor.

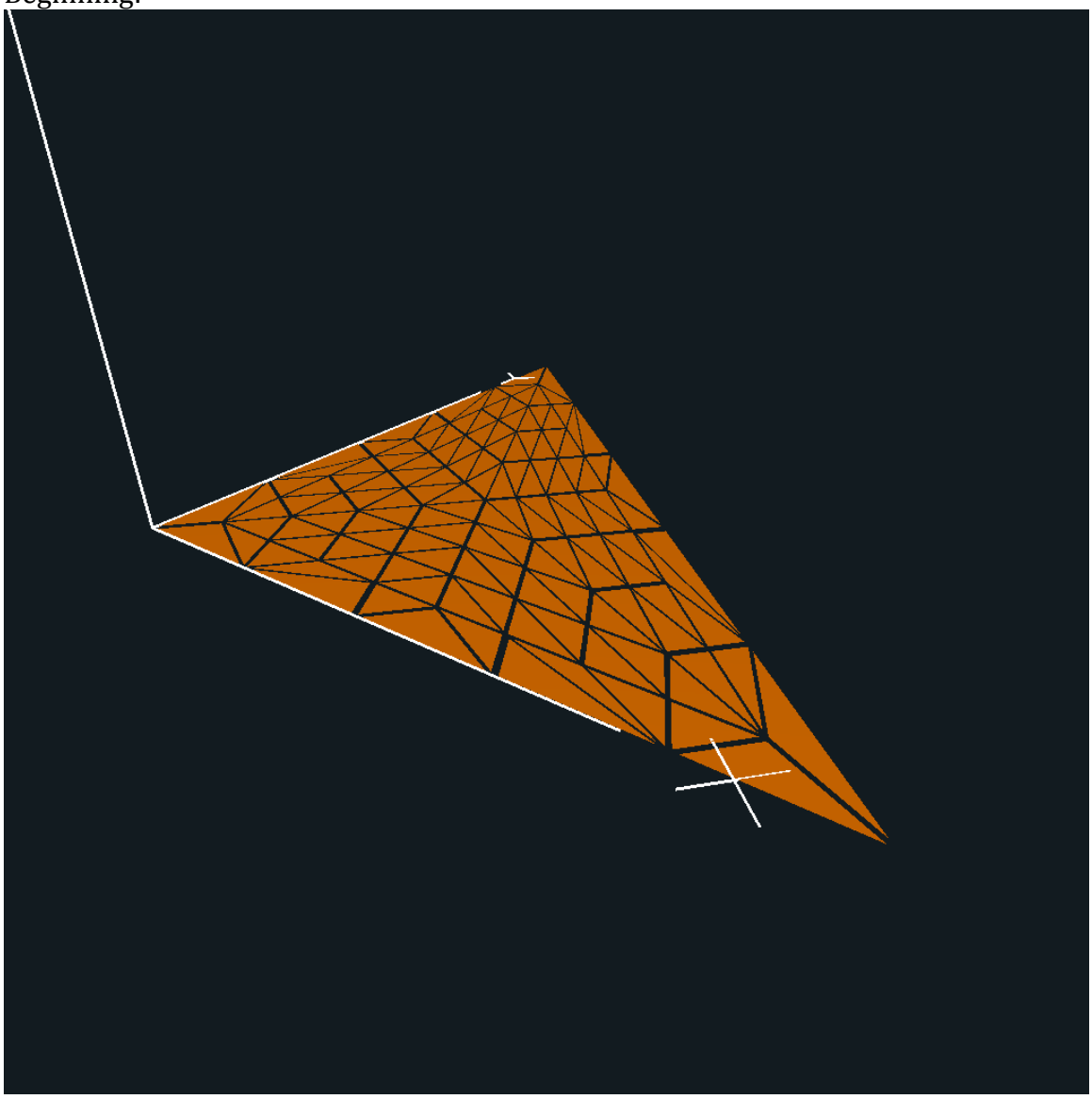
Last important part is deciding the range of uOuter01, 12, 20, uInner and uZ01, 12, 20. When checkbox is true, gl_TessLevelOuter[0] = float(uOuter12)+uZ12 this knid of codes would be executed. So, as we can see, we need to guarantee uOuter12+uZ12, uOuter01+uZ01, uOuter20+uZ20 are >= 1, especially when uZ01, 12, 20 are negative. So, I set like that:

uOuter01 <3 5 50> \
uOuter12 <3 5 50> \
uOuter20 <3 5 50> \
uInner <1 10 50> \

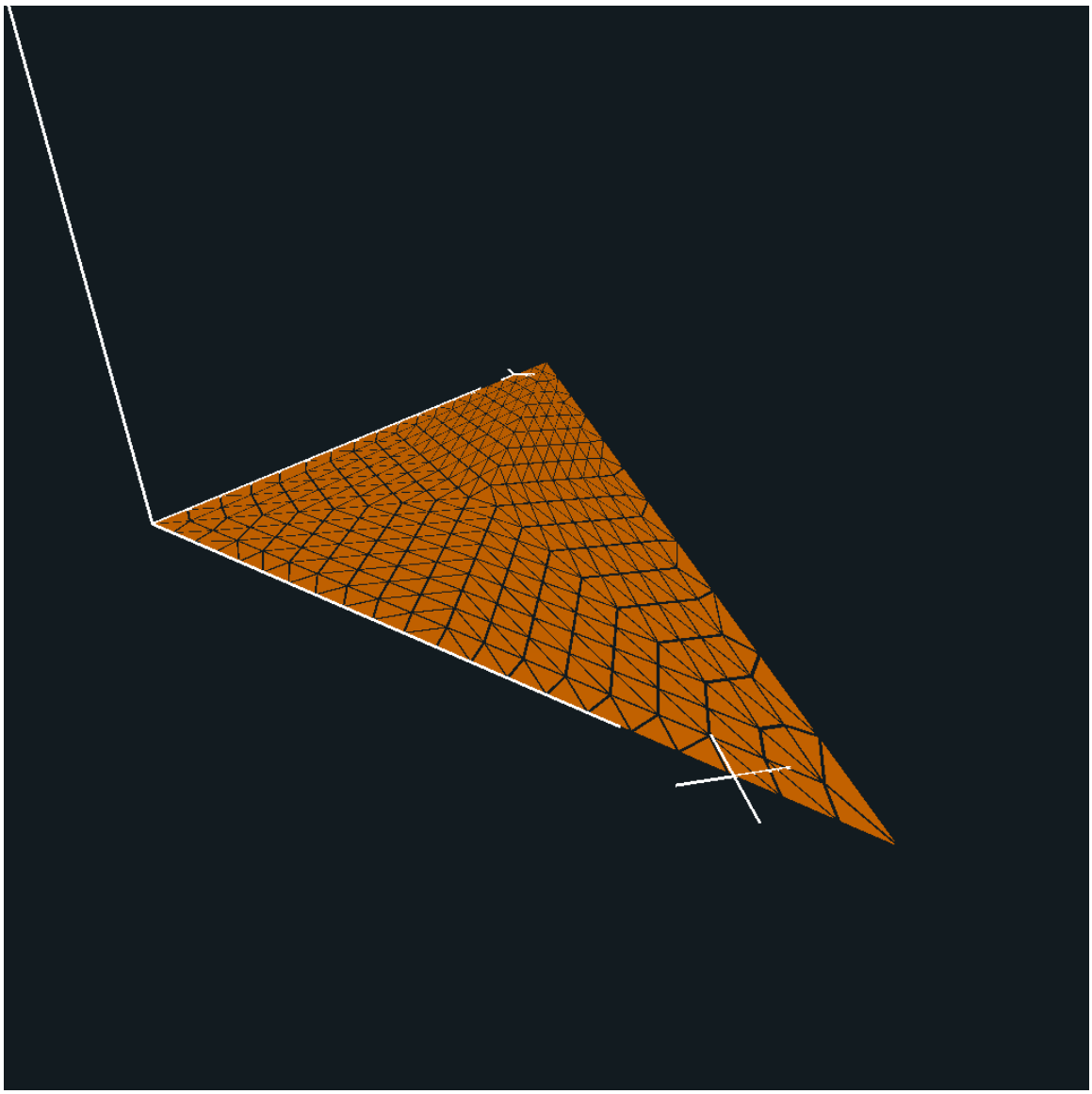
uZ01 <-2 0 2> \
uZ12 <-2 0 2> \
uZ20 <-2 0 2> \
to guarantee uOuter and uInner at least 3, and it add the minium uZ(-2), 3+(-2) still >=1. That would guarantee at least one triangle is presented in the screen.

(3) Results

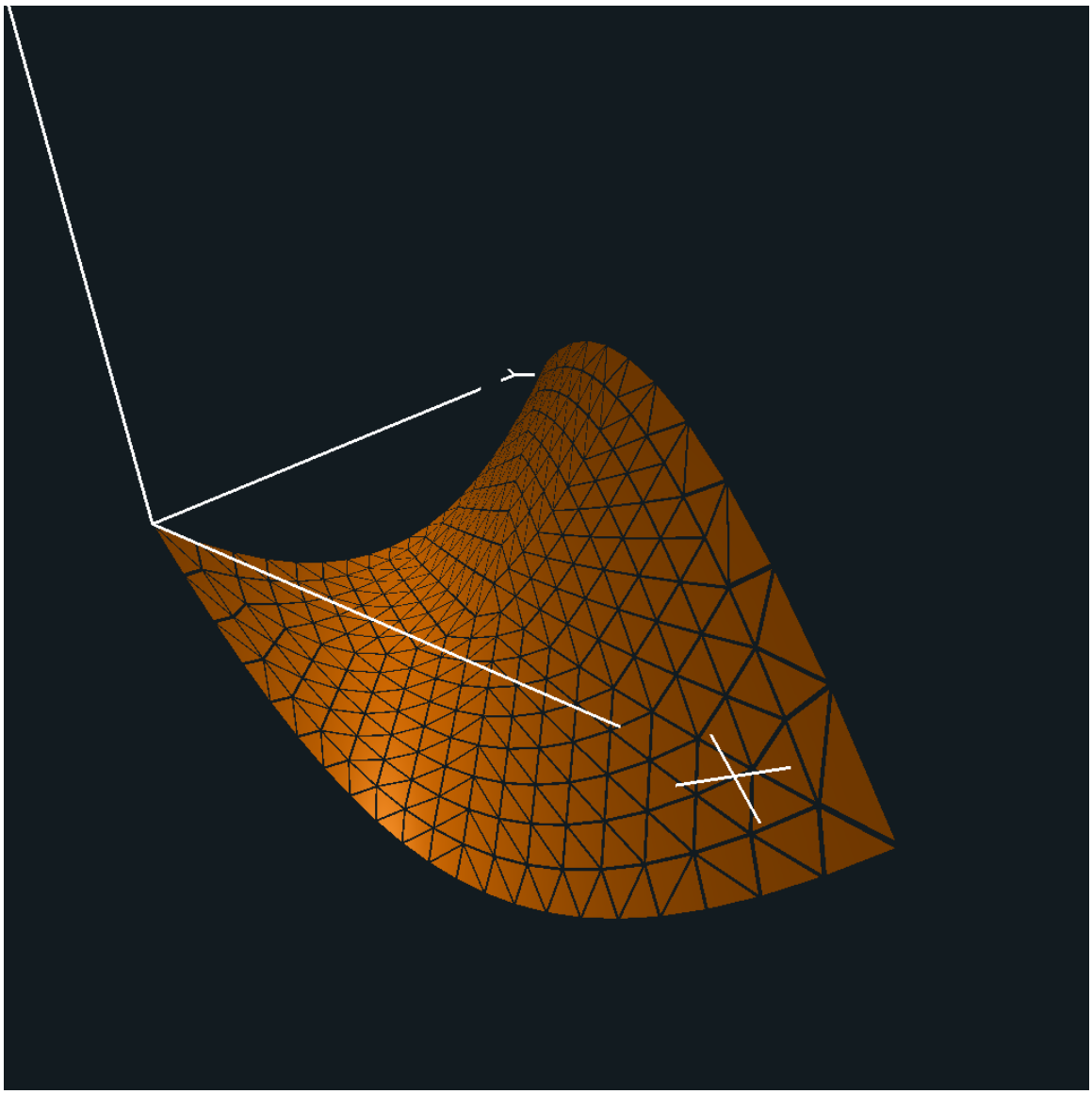
Beginning:



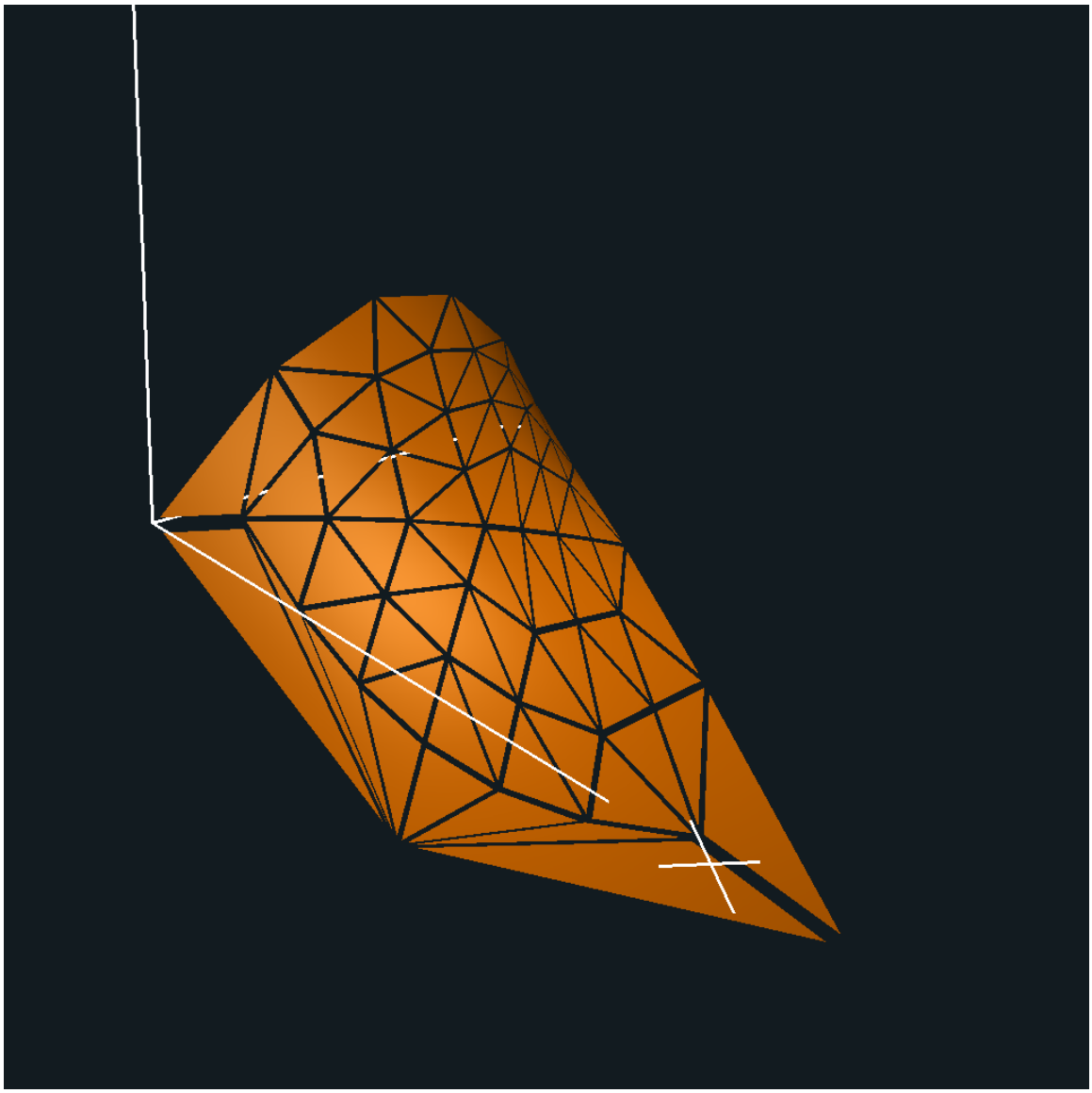
Changing Outer and Inner:

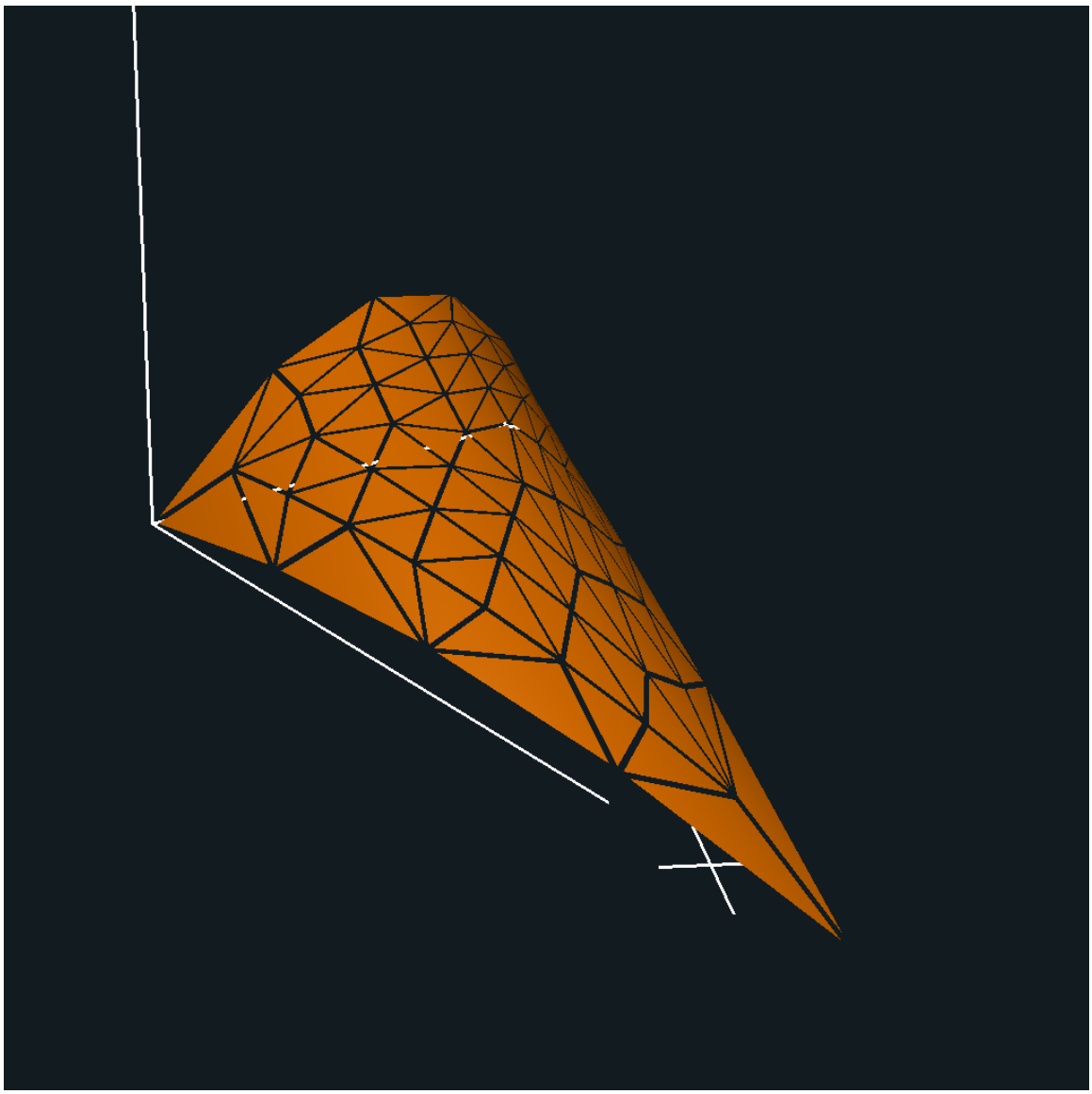


Checkbox is False, Outer and Inner will not change by uZ:



When checkbox is True, Outer and Inner will change by uZ (following two pictures show this):





Shrinking = 1, only can see a triangle: