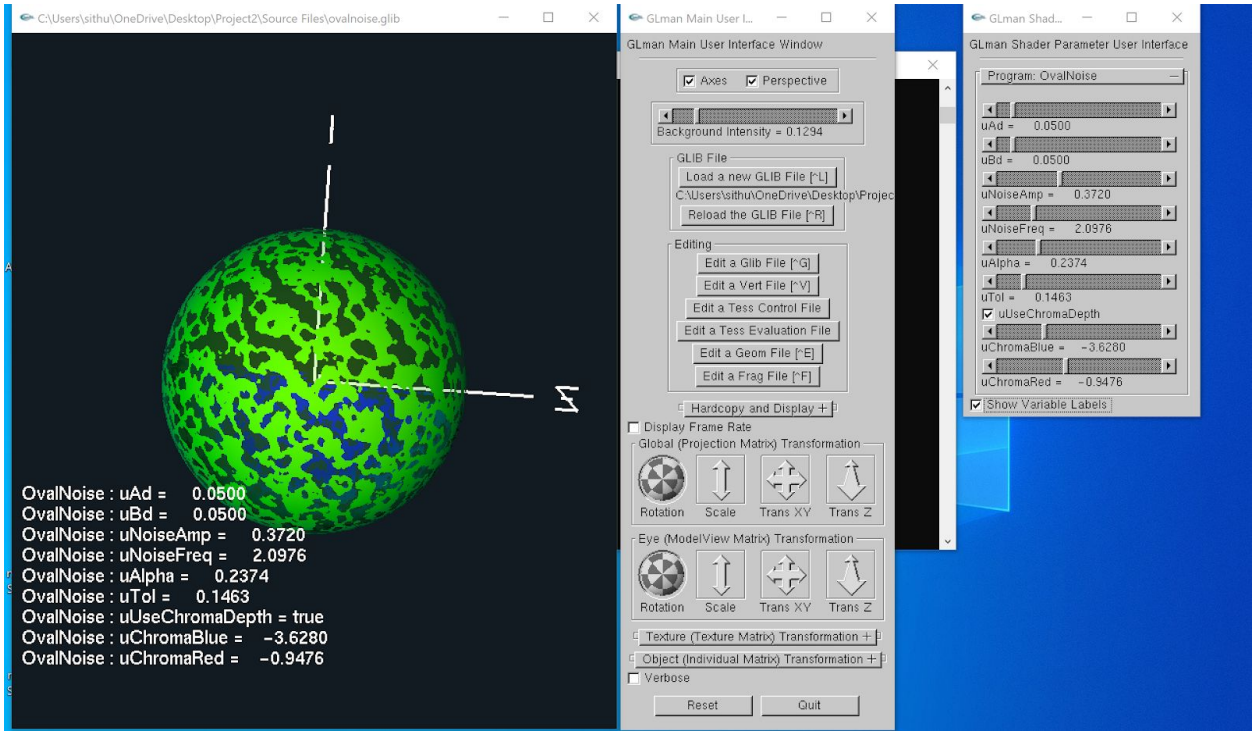
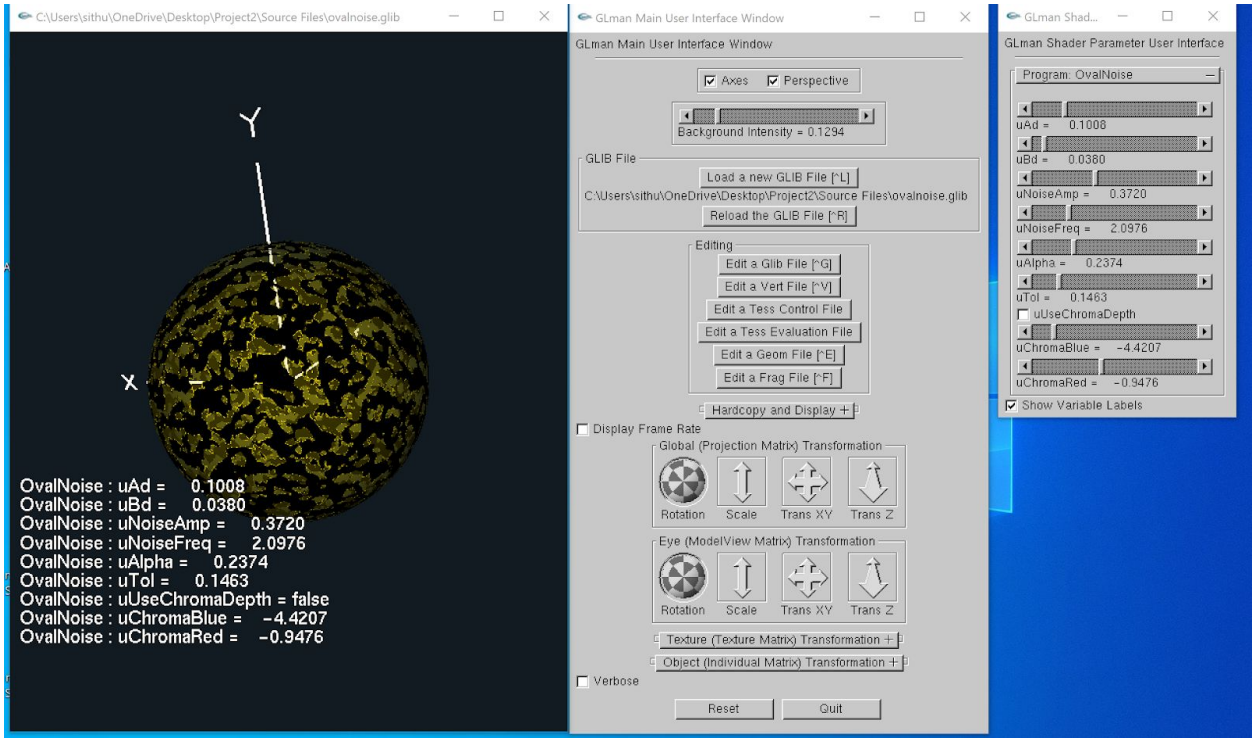
Title : Noisy Elliptical Dots
Name : Si Thu Lin
Email : linsi@oregonstate.edu
Video : https://media.oregonstate.edu/media/t/0_7ks46vz9




C:\Users\sithu\OneDrive\Desktop\Project2\Source Files\ovalnoise.glib
Y
X
OvalNoise : uAd = 0.1008
OvalNoise : uBd = 0.0380
OvalNoise : uNoiseAmp = 0.3720
OvalNoise : uNoiseFreq = 2.0976
OvalNoise : uAlpha = 0.2374
OvalNoise : uTol = 0.1463
OvalNoise : uUseChromaDepth = false
OvalNoise : uChromaBlue = -4.4207
OvalNoise : uChromaRed = -0.9476
GLman Main User Interface Window
GLman Main User Interface Window
Axes
Perspective
Background Intensity = 0.1294
GLIB File
Load a new GLIB File [^L]
C:\Users\sithu\OneDrive\Desktop\Project2\Source Files\ovalnoise.glib
Reload the GLIB File [^R]
Editing
Edit a Glib File [^G]
Edit a Vert File [^V]
Edit a Tess Control File
Edit a Tess Evaluation File
Edit a Geom File [^E]
Edit a Frag File [^F]
Hardcopy and Display +
Display Frame Rate
Global (Projection Matrix) Transformation
Rotation
Scale
Trans XY
Trans Z
Eye (ModelView Matrix) Transformation
Rotation
Scale
Trans XY
Trans Z
Texture (Texture Matrix) Transformation +
Object (Individual Matrix) Transformation +
Verbose
Reset
Quit
GLman Shad...
GLman Shader Parameter User Interface
Program: OvalNoise
uAd = 0.1008
uBd = 0.0380
uNoiseAmp = 0.3720
uNoiseFreq = 2.0976
uAlpha = 0.2374
uTol = 0.1463
uUseChromaDepth
uChromaBlue = -4.4207
uChromaRed = -0.9476
Show Variable Labels

```
#version 330 compatibility
out vec4  vColor;
out vec3  vMCposition;
out float vLightIntensity;
out vec2  vST;
out float Z;
const vec3 LIGHTPOS = vec3( -2., 0., 10. );

void
main( )
{
    vST = gl_MultiTexCoord0.st;
     vec3 tnorm = normalize( gl_NormalMatrix * gl_Normal );
     vec3 ECposition = vec3( gl_ModelViewMatrix *
gl_Vertex );
     vLightIntensity  = abs( dot( normalize(LIGHTPOS -
ECposition), tnorm )  );
     vColor = gl_Color;
     vMCposition = gl_Vertex.xyz;
     gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
     Z = ECposition.z;
}
```

```
##OpenGL GLIB
LookAt  0 0 3  0 0 0  0 1 0
Perspective 70

Vertex   ovalnoise.vert
Fragment ovalnoise.frag
Program  OvalNoise
\
        uAd <.01 .05 .5>  uBd <.01 .05 .5>
\
        uNoiseAmp <0. 0. 1.>  uNoiseFreq <0. 1. 10.>       \
        uAlpha <0. 1. 1.>
\
        uTol <0. 0. 1.>                                  \
        uUseChromaDepth <false>                              \
        uChromaBlue <-5.  -3.8  0.>
\
        uChromaRed  <-3.  -1.1  2.>

Color 1. .9 0
Sphere
```

```
main( )
{
 vec4 nv = texture3D( Noise3, uNoiseFreq * vMCposition );
      float n = nv.r + nv.g + nv.b + nv.a;//range is 1. -> 3.
      n = n - 2.;     //range is -1. -> 1.

      float s = vST.s;
      float t = vST.t;
      float sp = 2. * s;
      float tp = t;

      float Ar = uAd/2.;
      float Br = uBd/2.;


      int numins = int( sp / uAd );
      int numint = int( tp / uBd );

      gl_FragColor = vColor;     // default color
      float alpha = 1.;

      float sc = float(numins)*uAd + Ar;
      float tc = float(numint)*uBd + Br;
      sp = sp - sc;
      tp = tp - tc;
  float oldDist = sqrt( sp*sp + tp*tp );
 float newDist = oldDist + uNoiseAmp * n;
 float scale = newDist/oldDist;
      sp *= scale;                              // scale by
noise factor
sp /= Ar;                                  // ellipse equation

tp *= scale;                               // scale by noise
factor
tp /= Br;                                  // ellipse equation
```

```
tp *= scale;                              // scale by noise
factor
tp /= Br;                                 // ellipse equation

float d = sp*sp + tp*tp;
     if( abs( d - 1. ) <= uTol )
     {
     float j = smoothstep( 1.-uTol, 1.+uTol,d);
     gl_FragColor = mix( uOvalColor, vColor, j );
     }
     else if( d <= 1.-uTol)
     {
     float a = smoothstep( 1.-uTol, 1.+uTol,d );
     gl_FragColor = mix( uOvalColor, vColor, a );
     }
     else if(d > 1.+uTol)
     {
     alpha = uAlpha;
     gl_FragColor = vColor;
     if (uAlpha==0.){
     discard;
     }
     }

     if (uUseChromaDepth)
     {
     float t = (2./3.) * ( Z - uChromaRed ) / ( uChromaBlue -
uChromaRed );
     t = clamp( t, 0., 2./3. );
     gl_FragColor.xyz = Rainbow( t );
     }
     gl_FragColor = vec4( vLightIntensity*gl_FragColor.xyz,
alpha);

}
```