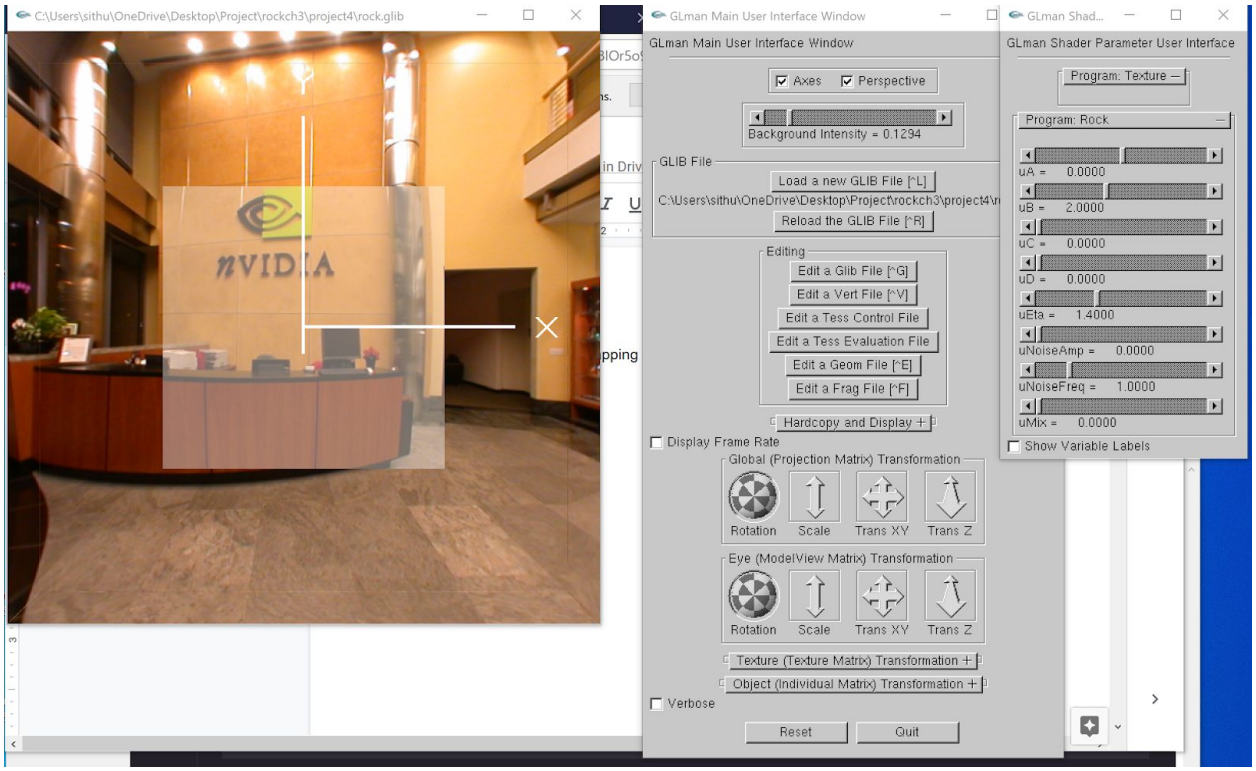
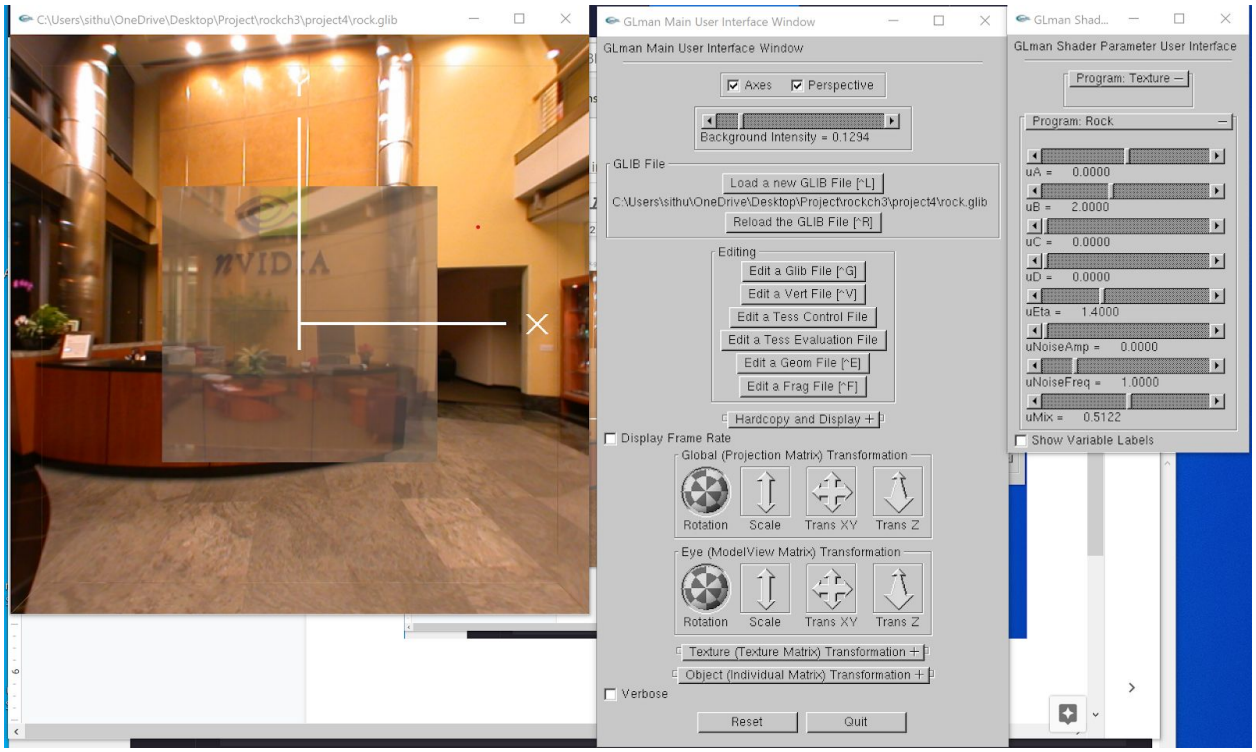
Title of the Project : Cube Mapping Reflective and Refractive Bump-mapped Surfaces
Video : https://media.oregonstate.edu/media/t/0_07nqcx0t
Name : Si Thu Lin
Email : linsi@oregonstate.edu

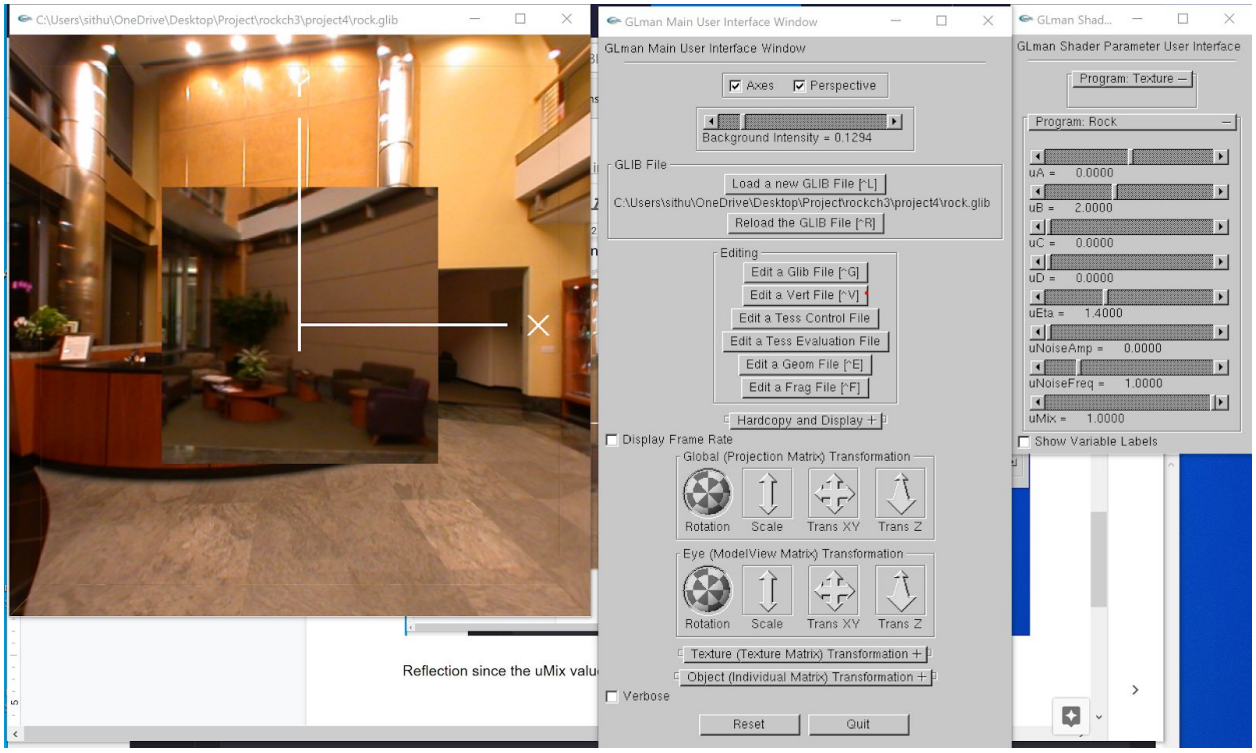
Refraction since the uMix value is 0.



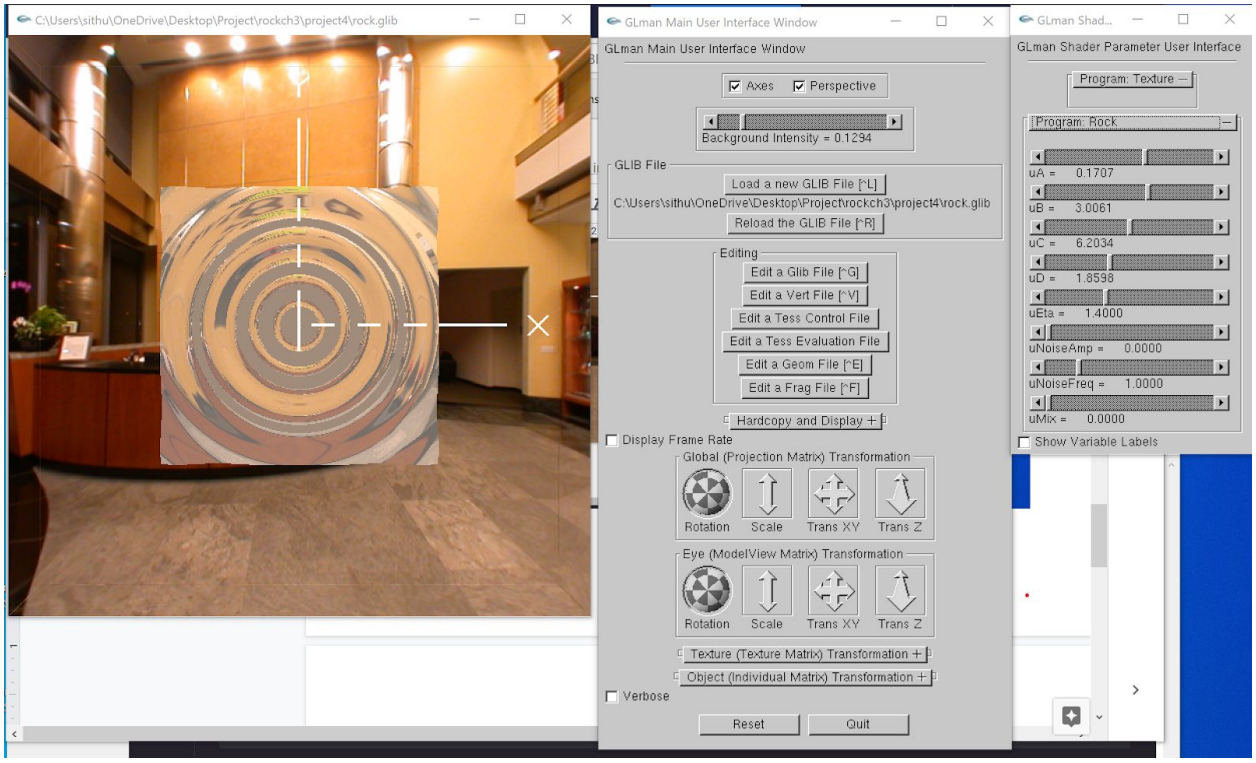
Blending of both the refraction and reflection since the uMix is around 0.5.



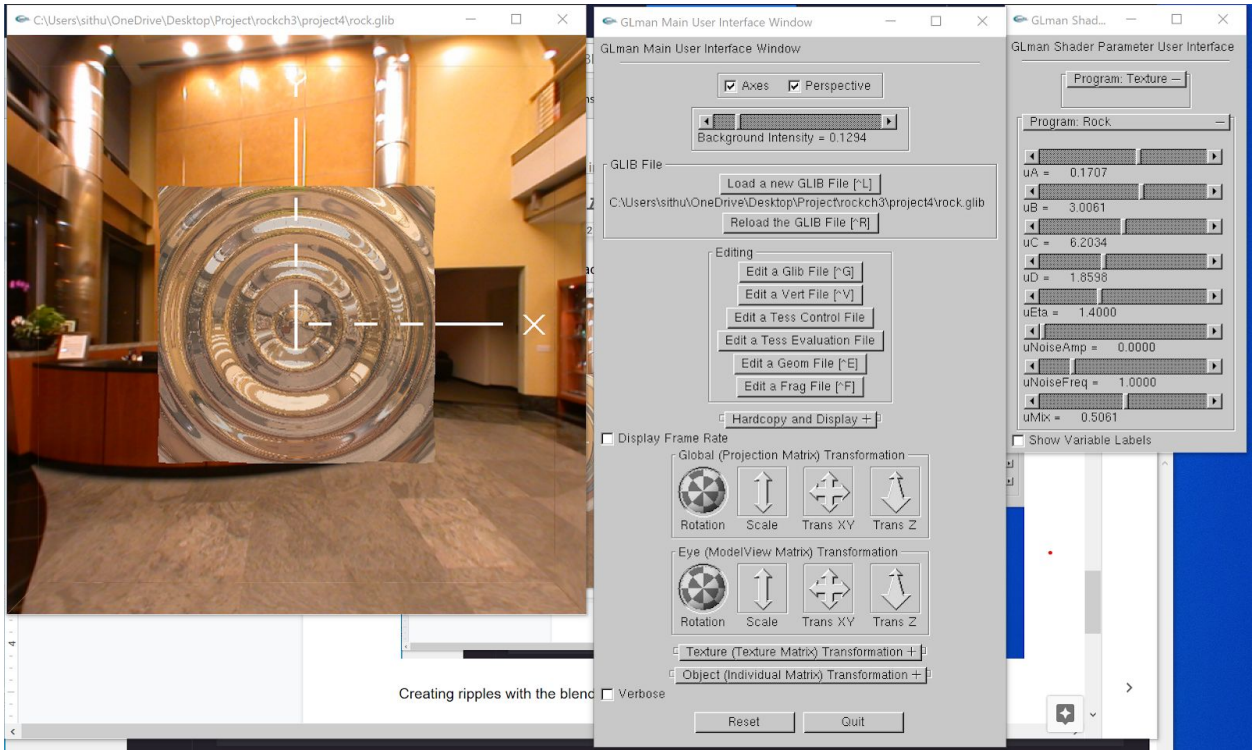
Reflection since the uMix value is 1.



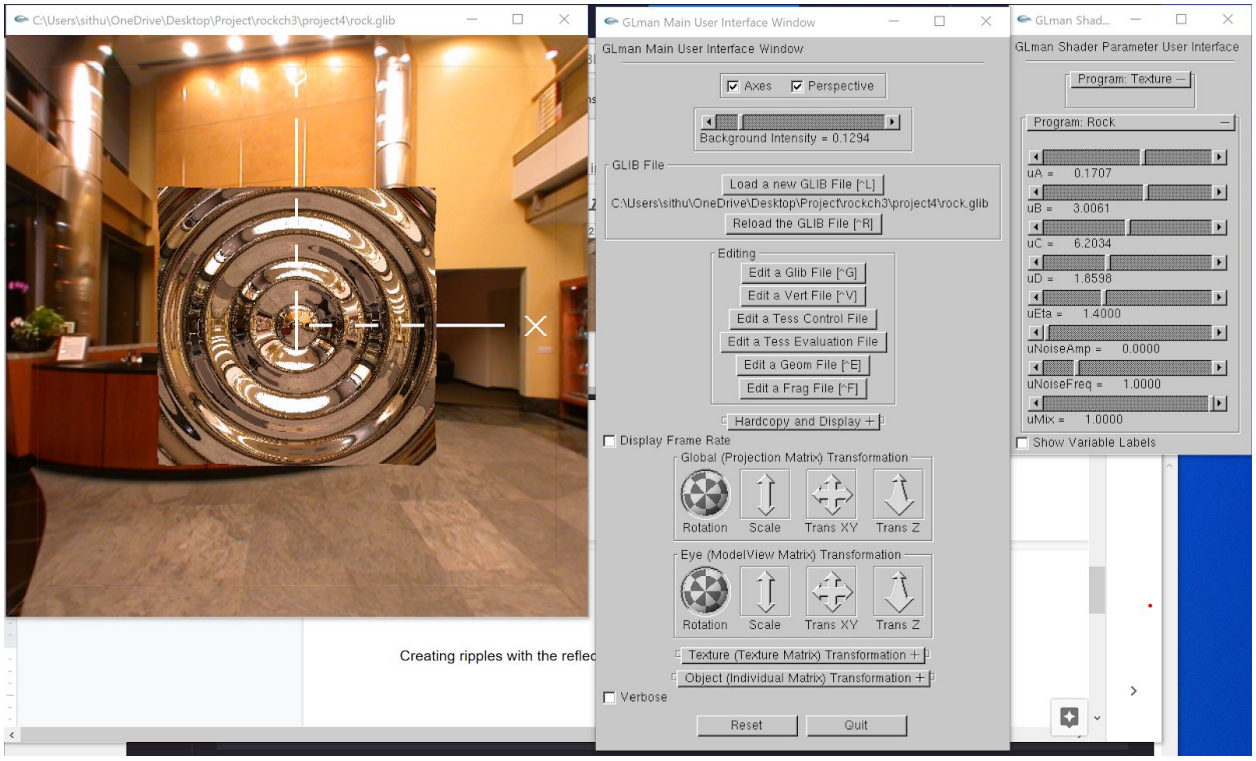
Creating ripples with the refraction.



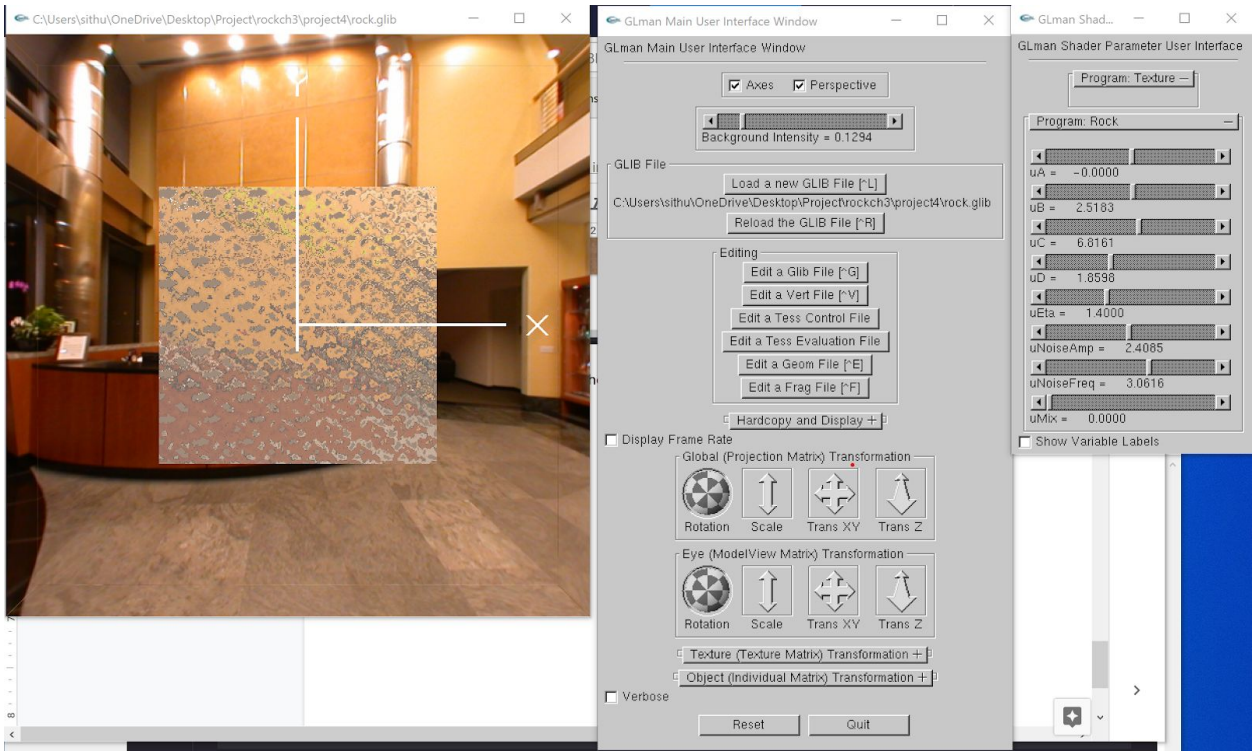
Creating ripples with the blending of refraction and reflection.



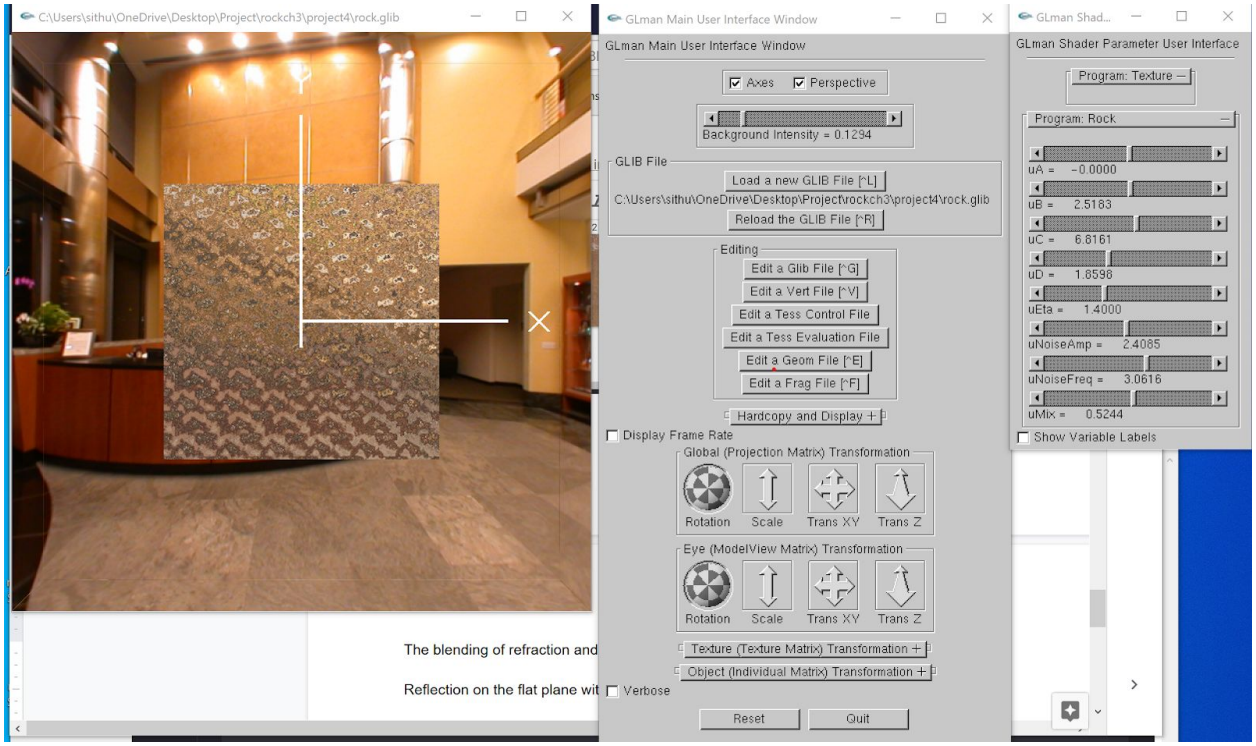
Creating ripples with the reflection.



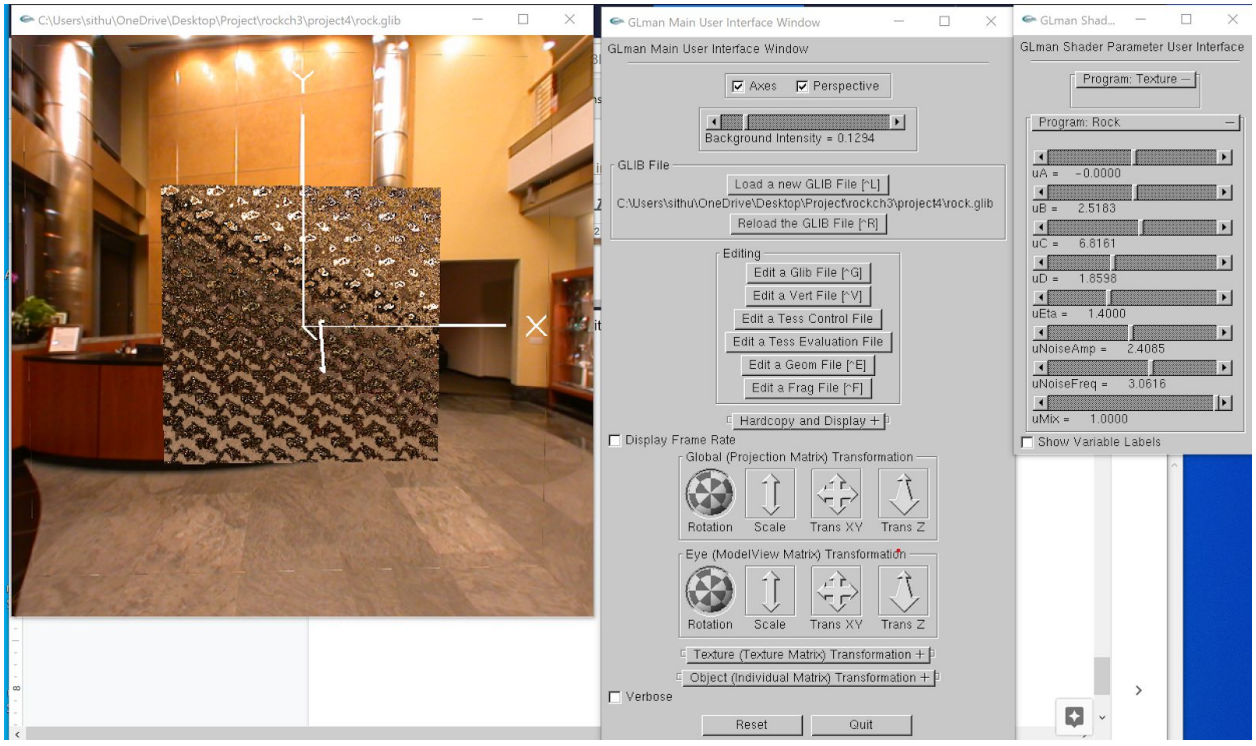
Refraction with noise on the flat plane.



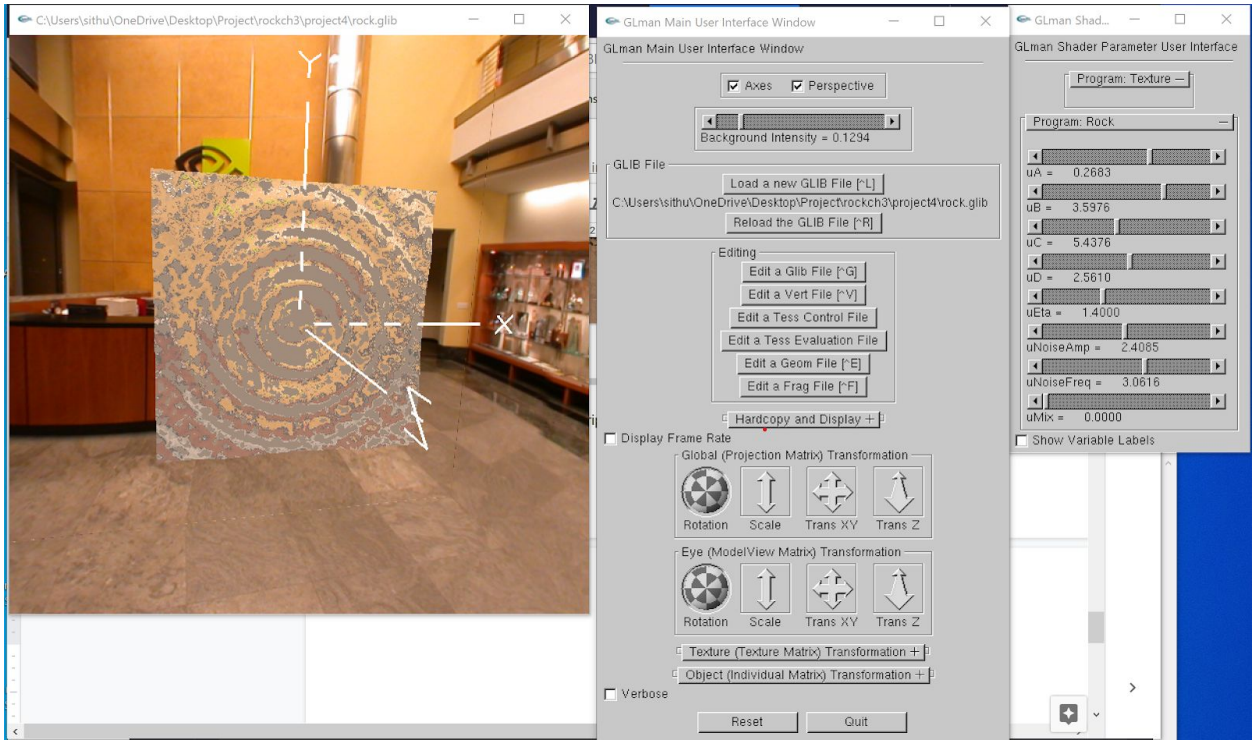
The blending of refraction and reflection with noise on the flat plane.



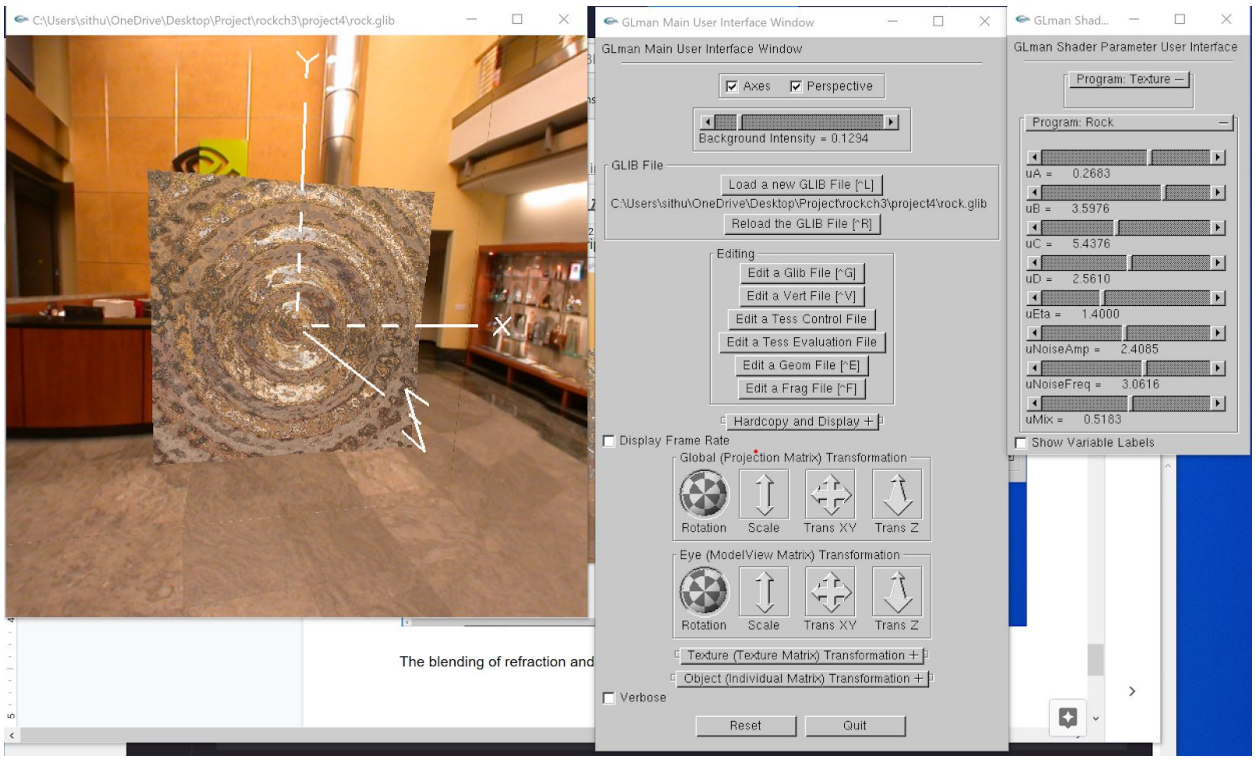
Reflection with noise on the flat plane.



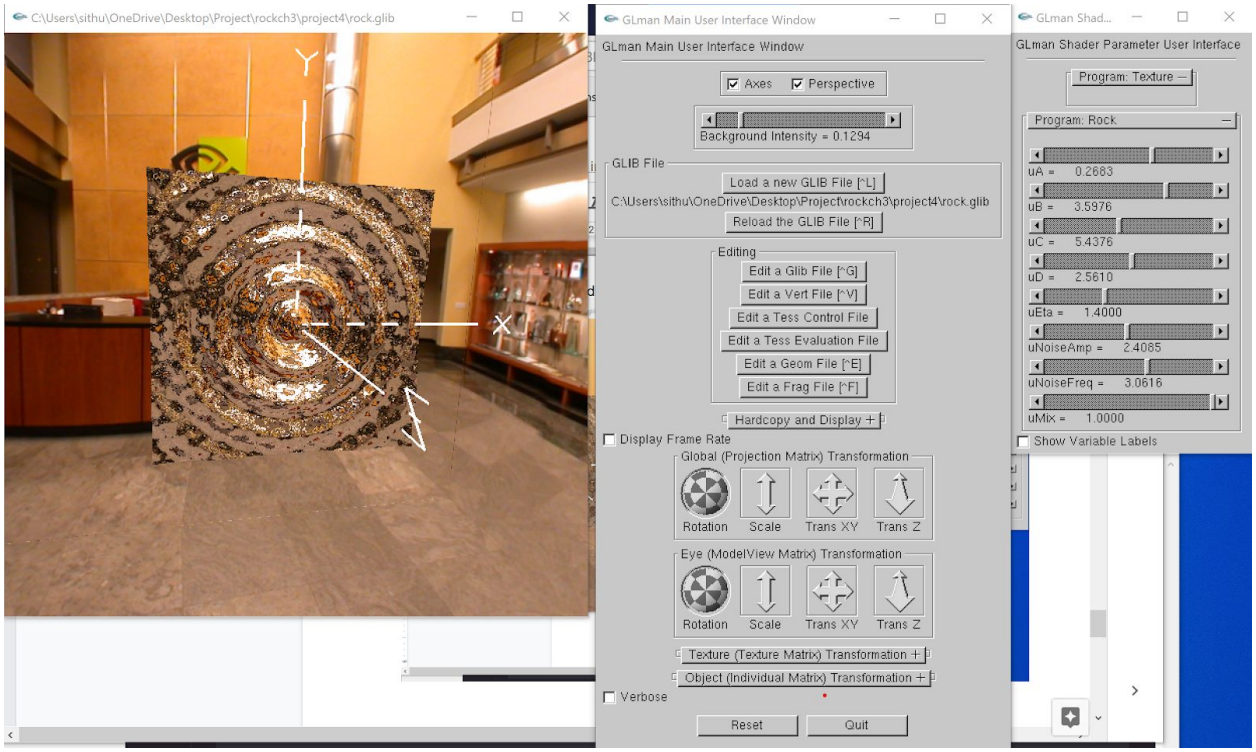
Refraction with noise on the ripples.



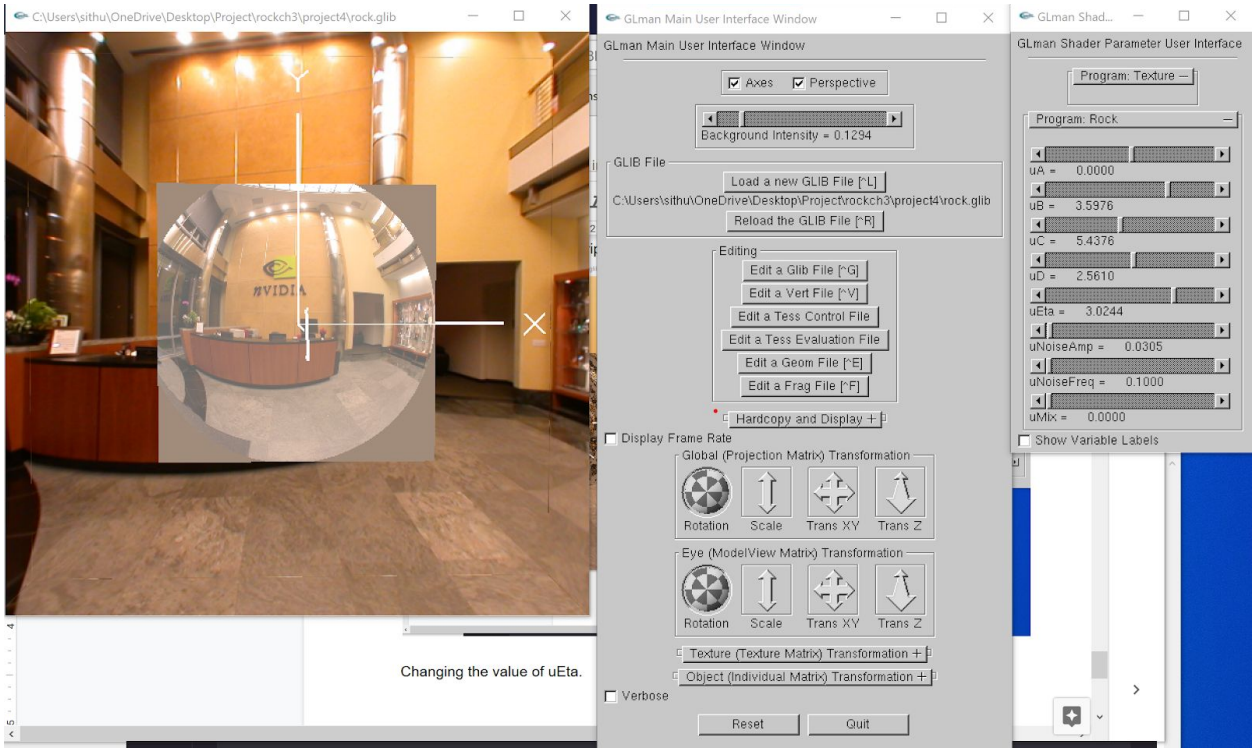
The blending of refraction and reflection with noise on the ripples.



Reflection with noise on the ripples.



Changing the value of uEta.



Codes for finding normal for the current in the Vertex Shader.

```
{
      vST = gl_MultiTexCoord0.st;
        float x = gl_Vertex.x;
        float y = gl_Vertex.y;
        float r=sqrt(pow(x,2.0)+pow(y,2));
        float drdx=x/r;
        float drdy=y/r;
        float dzdr = uA * ( -sin(2.*PI*uB*r+uC) * 2.*PI*uB *
exp(-uD*r) + cos(2.*PI*uB*r+uC) * -uD * exp(-uD*r) ) ;
float dzdx = dzdr * drdx;
float dzdy = dzdr * drdy;









      //vec4 p = gl_Vertex;
vec4 p=vec4(x,y,uA*cos((2*PI*uB*r)+uC)*exp(-
uD*r),gl_Vertex.w);

      vMC = p.xyz;




      vec3 Tx = vec3(1., 0., dzdx);
      vec3 Ty = vec3(0., 1., dzdy);


      vec3 normal = normalize(cross(Tx, Ty));
```

Codes for calculating the eye direction for the current vertex in the Vertex Shader.

```
      vec3 ECposition = vec3(gl_ModelViewMatrix * p);

      vNf = gl_NormalMatrix * normal;
      //vLf = eyeLightPosition - ECposition.xyz;
      vEf = normalize(ECposition-vec3(0., 0., 0.));
```

Codes for calculating the refract vector and reflect vector to calculate the refract color and reflect color for the current fragment in the fragment shader.

```
        Normal = RotateNormal(angx, angy, vNf);
        //Light = normalize(vLf);
        Eye = normalize(vEf);
```

```
vec3 eyeDir=normalize(vEf);
vec3 fNormal=normalize(Normal);
vec3 vRefractVector=refract(eyeDir,fNormal,uEta);
vec3 vReflectVector=reflect(eyeDir,fNormal);
```

```
vec4 refractcolor = textureCube( uRefractUnit,
vRefractVector );
vec4 reflectcolor = textureCube( uReflectUnit,
vReflectVector );
refractcolor = mix( refractcolor, WHITE, .40 );
```

Codes for mixing the refract color and reflect color and assigning the result to the color of the current fragment.

```
      gl_FragColor = vec4( mix( refractcolor, reflectcolor,
uMix ).rgb, 1. );
```