Project #4: Cube Mapping Reflective and Refractive Bump-mapped Surfaces

Jenny Zhong

zhongje@oregonstate.edu

Description:

Variables uA, uB, uC, and uD control Amplitude, Period, Phase shift and decay rate respectively. uD is set at 0.5, uEta is set at 1.2, uWhiteMix is set at 0.3 uA, uB, uNoiseAmp, uNoiseFreq, and uMix are the varying parameters. Variables uD, uEta and uWhiteMix do not vary.

```
uA <-1.0 0.0 1.0>          \
uB <0.0 2.0 5.0>           \
uC <0.0 0.0 12.56>                       \
uD 0.5                 \
uNoiseAmp <0. 0. 5.>       \
uNoiseFreq <0.0 0.1 0.5>  \
uEta 1.2           \
uMix <0. 0. 1.>            \
uWhiteMix 0.3              \
```

The surface of the quad spreads radially across X and Y, rising and falling Z and decaying in R. X and Y decay in radius around a center point to appear like ripples in the vertex shader. To set r and z, I used the following equations:

```
float r = sqrt((gl_Vertex.x)*(gl_Vertex.x)+(gl_Vertex.y)*(gl_Vertex.y));

float z = uA * ( cos(2.*PI*uB*r +uC) * exp(-uD*r) );
```

Use below lines to calculate dzdx and dzdy to use in equation for calculating normal vector.

```
float dzdx = uA * ( -sin(2.*PI*uB*r+uC) * 2.*PI*uB * exp(-uD*r) + cos(2.*PI*uB*r+uC) * -uD * exp(-uD*r) ) *(gl_Vertex.x/r);
```

float dzdy = uA * (-sin(2.*PI*uB*r+uC) * 2.*PI*uB * exp(-uD*r) + cos(2.*PI*uB*r+uC) * -uD * exp(-uD*r))*(gl_Vertex.y/r);

Normal vector for lighting is formed by taking the cross product of tangent vectors. Normal is calculated with below equation and passed over to fragment shader as an out vec3 vN.

vN = normalize(gl_NormalMatrix * cross(vec3(1., 0., dzdx), vec3(0., 1., dzdy)));

Set the quad to be 300 x 300 in the GLIB file so that there are enough vertices to create a smooth displacement function.

QuadXY -0.2 2. 300 300

Use noise texture capability to get two noise values. Two values will be used as an angle to rotate the normal about x and an angle to rotate the normal about y. Vary on two uniform variables uNoiseAmp and uNoiseFreq. vMC are the vec3 model coordinates passed from the vertex shader. Get noise value by indexing into a noise texture, and use all 4 octaves RGBA.

```
vec4 nvx = texture( Noise3, uNoiseFreq*vMCposition );

float angx = nvx.r + nvx.g + nvx.b + nvx.a  -  2.;      // -1. to +1.

angx *= uNoiseAmp;


vec4 nvy = texture( Noise3, uNoiseFreq*vec3(vMCposition.xy,vMCposition.z+0.5) );

float angy = nvy.r + nvy.g + nvy.b + nvy.a  -  2.;      // -1. to +1.

angy *= uNoiseAmp;
```

Rotate the normal with RotateNormal function, then multiply by the gl_NormalMatrix and normalize. Apply per-fragment lighting to uColor in the fragment shader.

```
vec3 n = RotateNormal( angx, angy, vN );
```

```
n = normalize(  gl_NormalMatrix * n  );
```

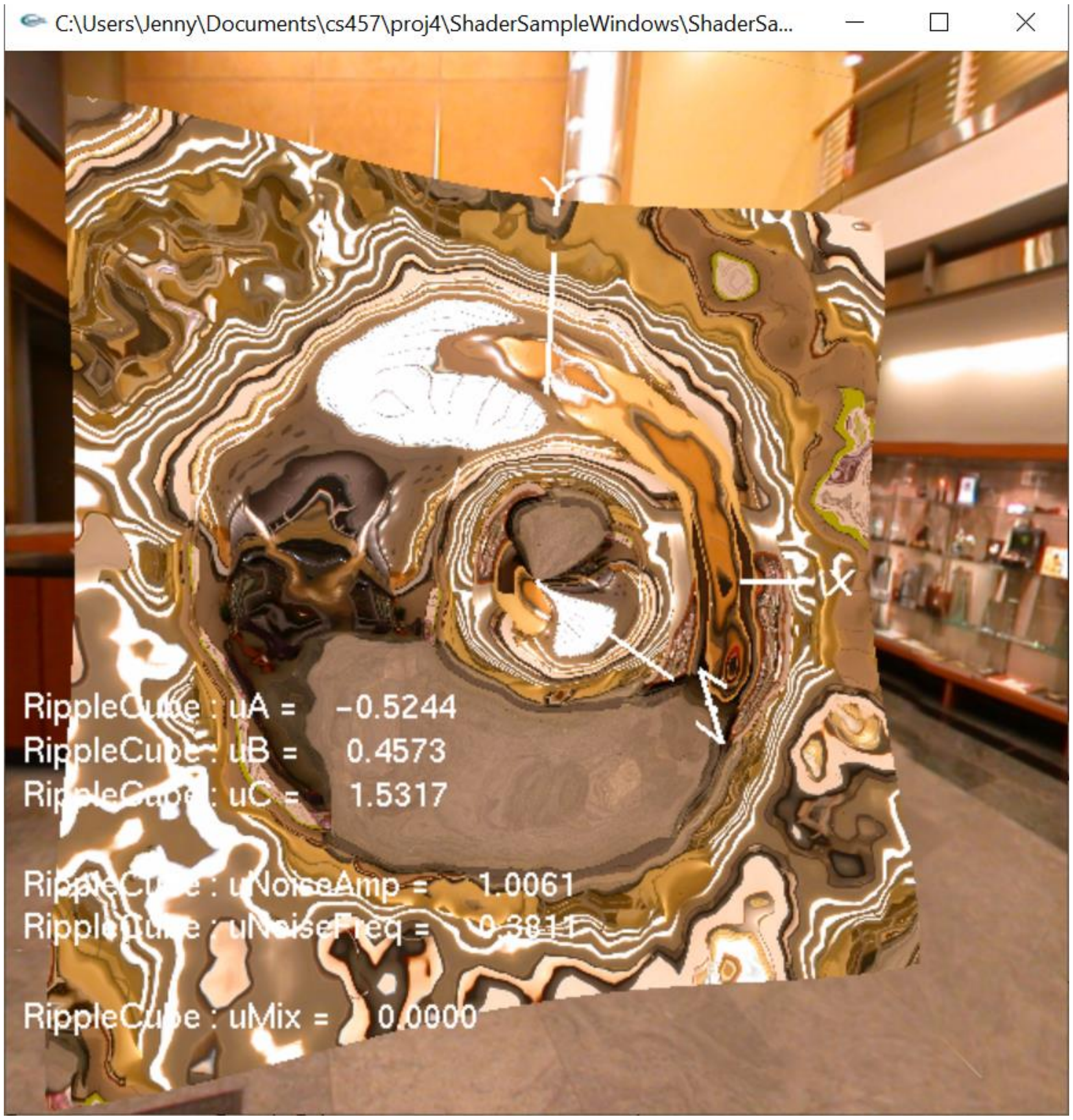
In the fragment shader, if all elements of refractVector are 0.0 then treat as total internal reflection, no refraction. Else, mix reflective and refractive outputs. Mix with white to show object being refracted.

```
if( all( equal( refractVector, vec3(0.,0.,0.) ) ) ) // like saying "if all elements
of the refractVector are == 0.0 ..."
{
refractColor = reflectColor; // . . . then treat this as a total internal
reflection
}
else
{
refractColor = texture( uRefractUnit, refractVector ).rgb; // on Macs, use
textureCube( )
refractColor = mix( refractColor, WHITE, uWhiteMix );
}
```

Six walls for the cubemapping are created in the GLIB file by importing the six BMP files.

Video Link: https://media.oregonstate.edu/media/t/1_uwgaf0fv

Screenshots:



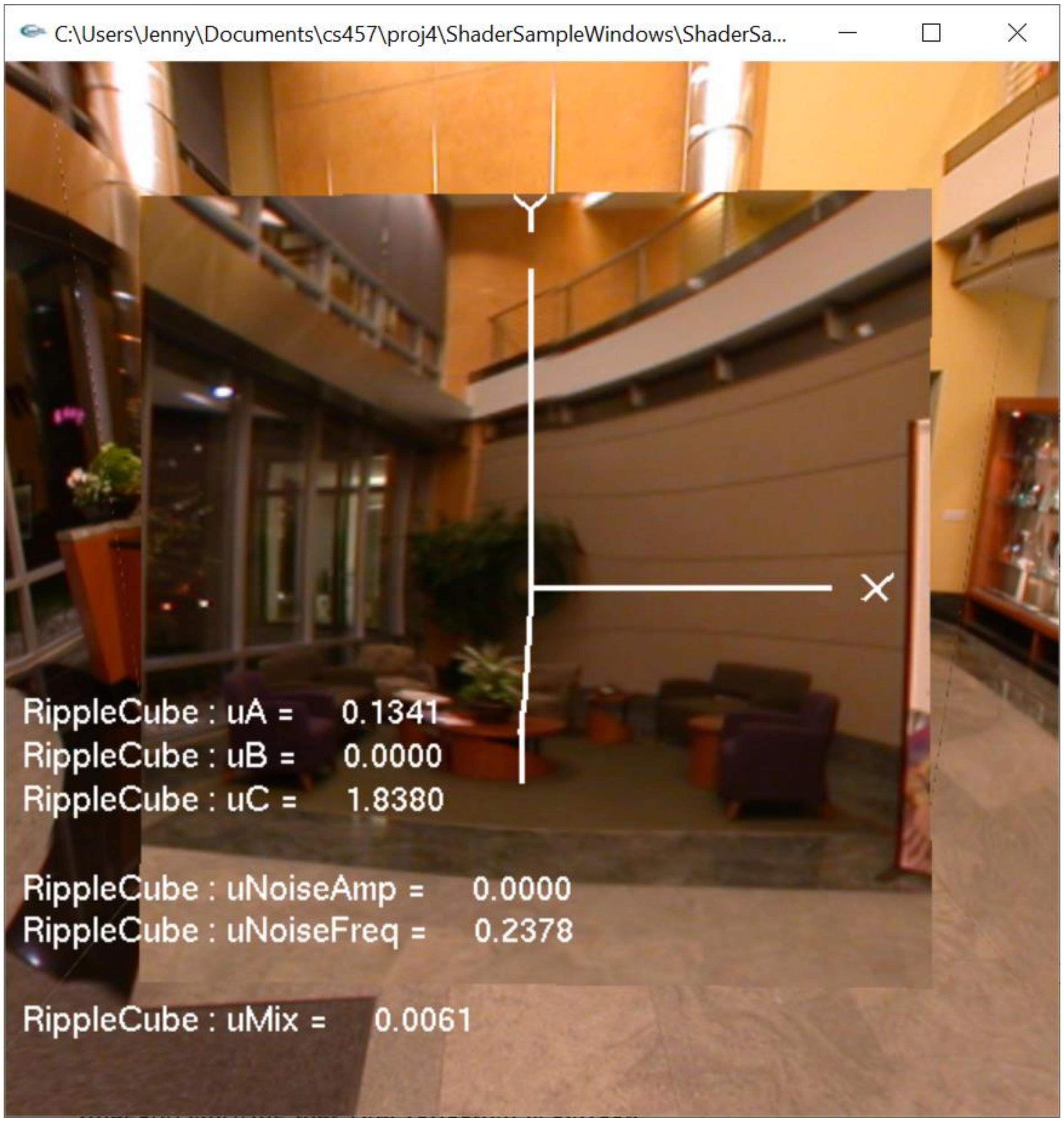
C:\Users\Jenny\Documents\cs457\proj4\ShaderSampleWindows\ShaderSa...

RippleCube : uA = -0.5244
RippleCube : uB = 0.4573
RippleCube : uC = 1.5317

RippleCube : uNoiseAmp = 1.0061
RippleCube : uNoiseFreq = 0.3811

RippleCube : uMix = 0.0000

Figure 1 uNoiseAmp at 1.0061, uNoiseFreq at 0.3811, uMix at 0., uA at -0.5244, uB at 0.4573, uC at 1.5317, uMix at zero so no refraction – vary noise frequency and amplitude to show bumpmapping manipulated surface normal



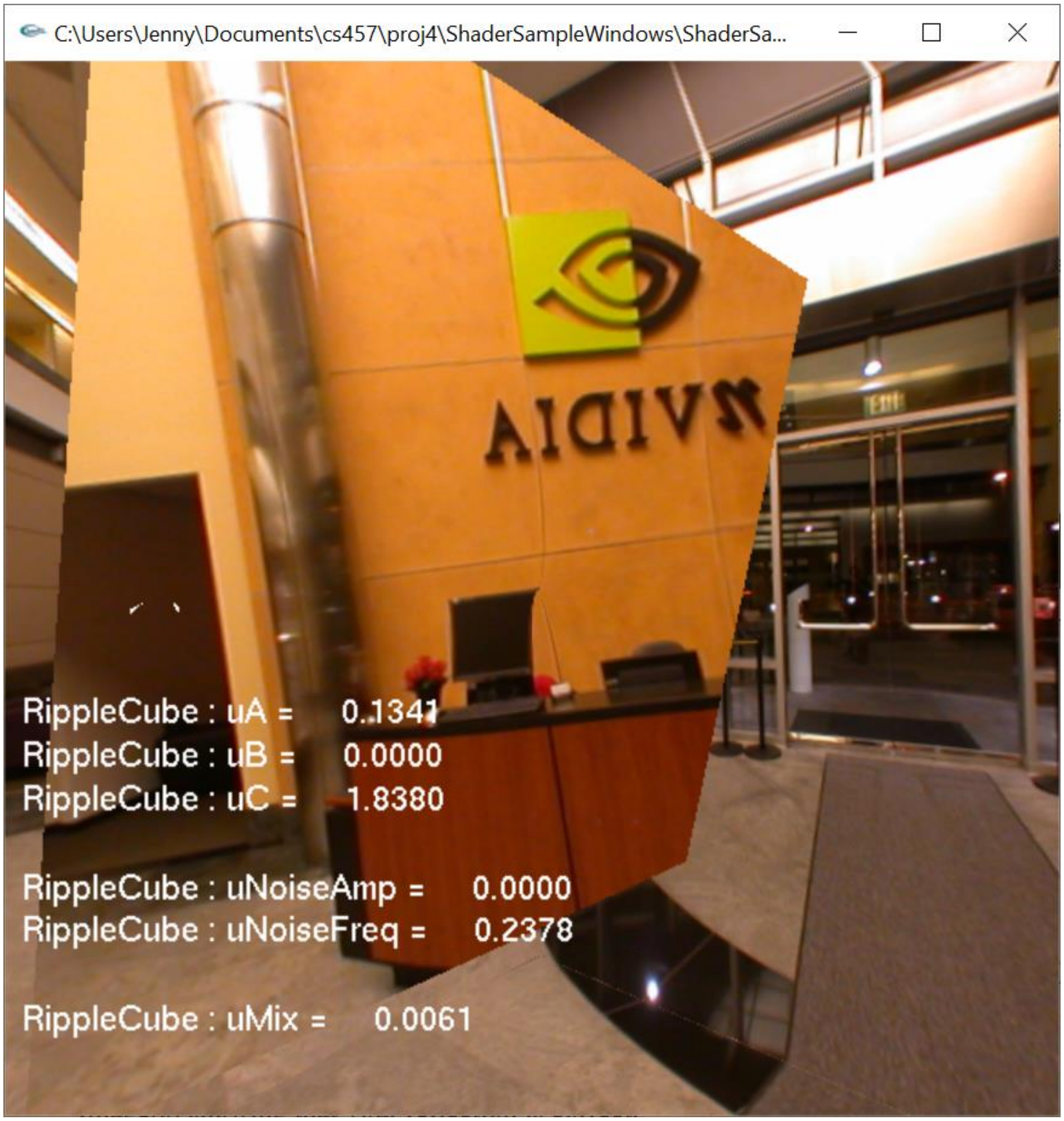
C:\Users\Jenny\Documents\cs457\proj4\ShaderSampleWindows\ShaderSa...

```
RippleCube : uA =     0.1341
RippleCube : uB =     0.0000
RippleCube : uC =     1.8380

RippleCube : uNoiseAmp =     0.0000
RippleCube : uNoiseFreq =     0.2378

RippleCube : uMix =     0.0061
```

Figure 2 Reflection A

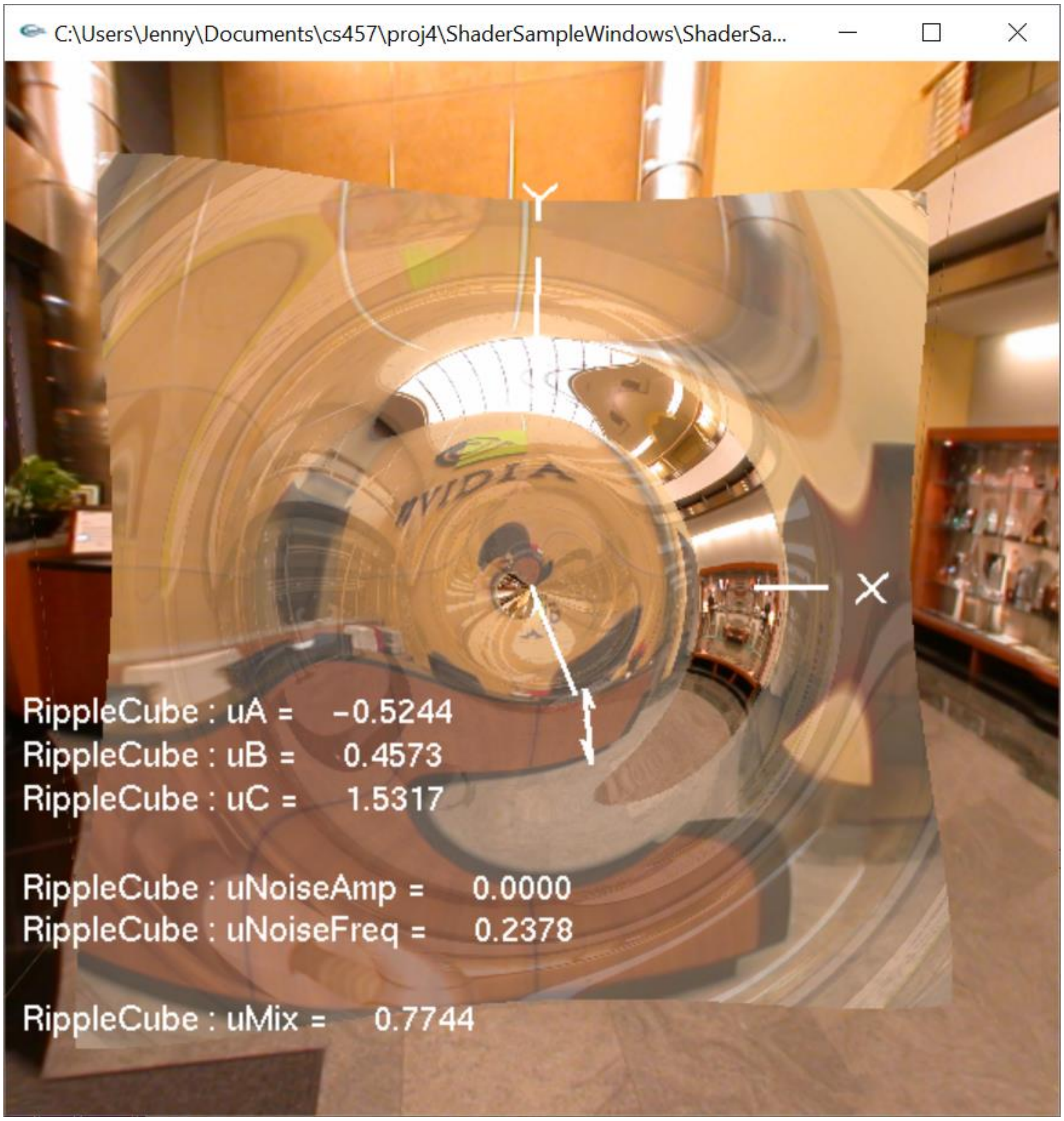

C:\Users\Jenny\Documents\cs457\proj4\ShaderSampleWindows\ShaderSa...

```
RippleCube : uA =     0.1341
RippleCube : uB =     0.0000
RippleCube : uC =     1.8380

RippleCube : uNoiseAmp =     0.0000
RippleCube : uNoiseFreq =     0.2378

RippleCube : uMix =     0.0061
```

Figure 3 Reflection B


C:\Users\Jenny\Documents\cs457\proj4\ShaderSampleWindows\ShaderSa...

```
RippleCube : uA =    -0.5244
RippleCube : uB =     0.4573
RippleCube : uC =     1.5317

RippleCube : uNoiseAmp =     0.0000
RippleCube : uNoiseFreq =    0.2378

RippleCube : uMix =     0.7744
```

Figure 4 Mix Reflective and Refractive outputs