Oregon State University

CS_557_X001_W2022 COMPUTER GRAPHICS SHADERS

Final Project

Professor: Mike Bailey
Student: Chengxu Xu
(xucheng@oregonstate.edu)

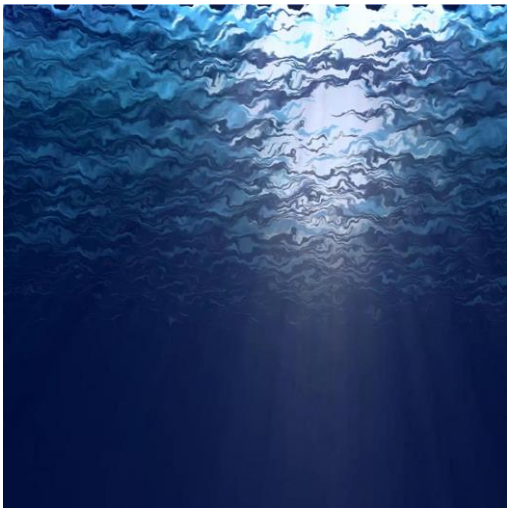
In this final project, I tried to implement a dynamic view of the ocean floor using the noise effect. My original idea was to implement an effect like Modeling Wave Motion, but most of the code for this project is already mentioned in slides, so I implemented another effect where I tried to control the shader by noise to change the shading effect on some areas and use it to simulate the dynamic effect of the sea.

Kaltura link: https://media.oregonstate.edu/media/t/1_wrr8vkxi

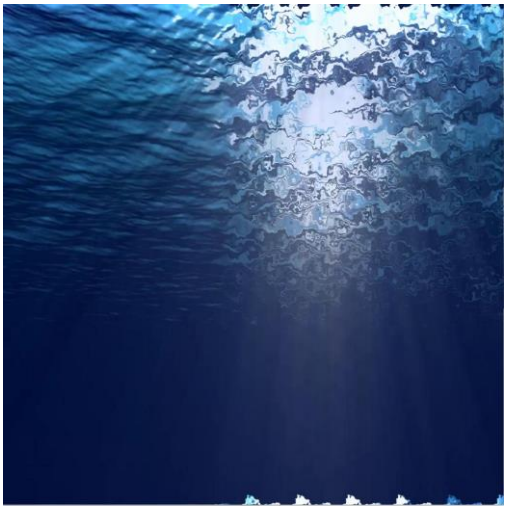
Screen Shots：

Original

adjust uNoiseFreq & u NoiseAmp

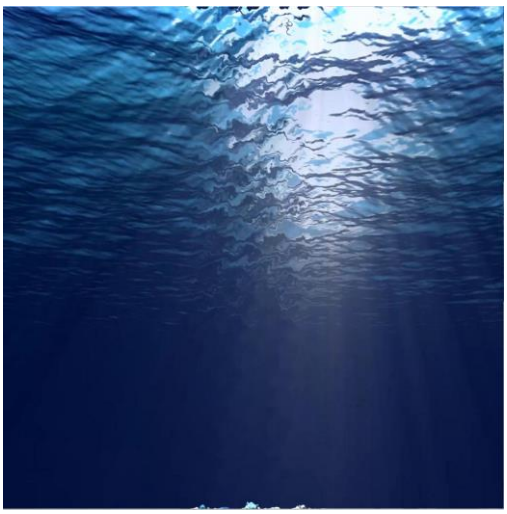


adjust uLeftRight

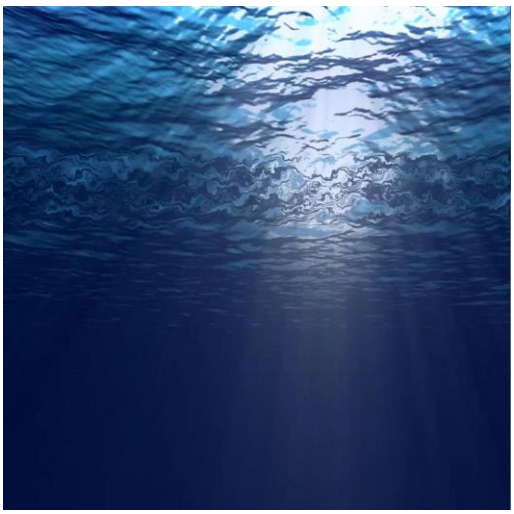


adjust uUpDown

adjust uWidth

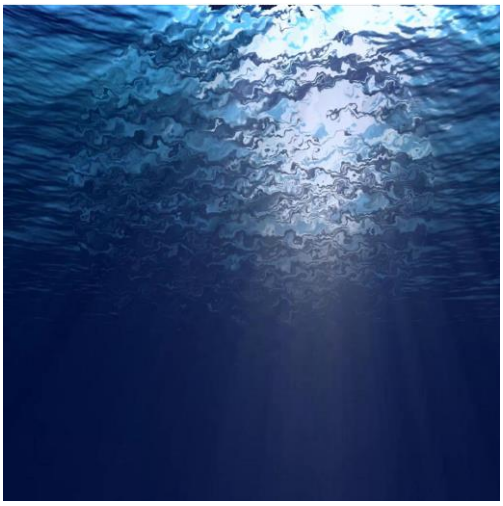


adjust uHeight



Click uCircle

adjust uRadius



Key snippets:

Special parameter predefined:

```
Vertex   ocean.vert
Fragment ocean.frag
Program  Ocean                          \
    uLeftRight   <0. .5 1.>             \
    uUpDown  <0. .66 1.>                \
    uWidth    <0.01 .5 .5>              \
    uHeight      <0.01 .5 .5>           \
    uCircle <0. 0. 1.>                  \
    uTaper <0. 1. 1.>                   \
    uRadius     <0. 0.1 .5>             \
    uNoiseFreq <0. 1.1 10.>             \
    uNoiseAmp <0. .01 .1>               \
    uDs <0. 4. 100.>                    \
    uImageUnit  5
```

Dynamic area selection:

```
bool inCircle(){
    if(sqrt(pow((vST.s-uLeftRight),2) + pow((vST.t-uUpDown),2))  <= uRadius){
        return true;
    }
    return false;
}
bool inRectangle(){
    float s = vST.s;
    float t = vST.t;
    if( s <= uLeftRight + uWidth && s >= uLeftRight - uWidth && t <= uUpDown + uHeight && t >= uUpDown - uHeight){
        return true;
    }
    return false;
}
```

Get circle area taper with uRadius and smoothstep() function:

```
float getCircleTaper(){
    float ConicalBusBar = sqrt(pow((vST.s-uLeftRight),2) + pow((vST.t-uUpDown),2));
    float mTaper = smoothstep(uRadius*2,uRadius,ConicalBusBar);
    return mTaper;
}
```

Get rectangle area taper:

```
float getRectTaper(){
    float minS = uLeftRight - uWidth;
    float maxS = uLeftRight + uWidth;
    float minT = uUpDown - uHeight;
    float maxT = uUpDown + uHeight;

    float ds = max ((max(minS - vST.s,0.)),(vST.s - maxS));
    float dt = max ((max(minT - vST.t,0.)),(vST.t - maxT));
    float d = sqrt(ds*ds + dt*dt);
    float mTaper = smoothstep(.1,0,d);
    return mTaper;
}
```

Judging the type of area selected, If not in the selected area then the taper is 1:

```
if(uCircle){
    if(inCircle()){
        len = true;
        mTaper = 1.;
    }else{
        mTaper = getCircleTaper();
    }
}else{
    if(inRectangle()){
        len = true;
        mTaper = 1.;
    }else{
        mTaper = getRectTaper();
    }
}
```

For coloring in dynamic regions: calculate the vST value with noise, uDS and taper, and calculate the corresponding color parameter for the corresponding position in the image.

```
float n = nv.r + nv.g + nv.b + nv.a; // range from 1. -> 3.
n -= 2;
n *= uNoiseAmp;

float newS, newT;
newS = vST.s + mTaper * n * cos(2 * PI * Timer * uDs);
newT = vST.t + mTaper * n * sin(2 * PI * Timer * uDs);
vec2 newVST = vec2( newS, newT);

newColor = texture( uImageUnit, newVST ).rgb;
gl_FragColor = vec4( newColor, 1. );
```

Coloring the remaining parts according to the image parameter:

```
}else{
    newColor = texture( uImageUnit, vST ).rgb;
    gl_FragColor = vec4( newColor, 1. );
}
```