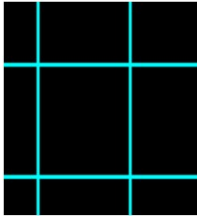


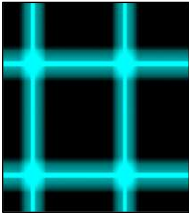
1

Bloom





Oregon State University
Mike Bailey
mjb@cs.oregonstate.edu





Oregon State University
Computer Graphics

mjb - December 26, 2024

2

Remember our Texture Image Basics in Shaders

Index the image using the usual texture indexing

$(0 \leq s, t \leq 1.)$

When you get back an RGB from the texture, remember that, if the texture's numbers are **colors**:

$(0. \leq r, g, b \leq 1.)$

If the texture contains **data**, then the numbers can be anything.

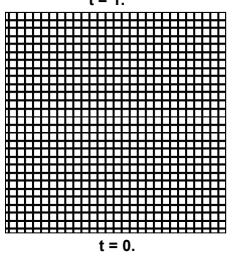
Also, if you need to know the texel resolution of this texture, do this:

```
ivec2 ires = textureSize( ulmageUnit, 0 );
float ResS = float( ires.s );
float ResT = float( ires.t );
```

ResT

s = 0.

t = 1.



t = 0.

ResS

Thus, to get from the current texel's (s,t) to a neighboring texel's (s,t), add

$\pm (1./ResS, 1./ResT)$



Oregon State University
Computer Graphics

mjb - December 26, 2024

3

And Remember How We Achieved a Blur Effect

Blur Convolution:

3x3

$$B = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

5x5

$$B = \frac{1}{100} \begin{bmatrix} 1 & 2 & 4 & 2 & 1 \\ 2 & 4 & 8 & 4 & 2 \\ 4 & 8 & 16 & 8 & 4 \\ 2 & 4 & 8 & 4 & 2 \\ 1 & 2 & 4 & 2 & 1 \end{bmatrix}$$



Oregon State University
Computer Graphics

mjb - December 26, 2024

4

blur.frag

```
vec2 stp0 = vec2(1./ResS, 0. );
vec2 st0p = vec2(0. , 1./ResT);
vec2 stpp = vec2(1./ResS, 1./ResT);
vec2 stpm = vec2(1./ResS, -1./ResT);

vec3 i00 = texture( ulmageUnit, vST ).rgb;
vec3 im1m1 = texture( ulmageUnit, vST-stpp ).rgb;
vec3 ip1p1 = texture( ulmageUnit, vST+stpp ).rgb;
vec3 im1p1 = texture( ulmageUnit, vST-stpm ).rgb;
vec3 ip1m1 = texture( ulmageUnit, vST+stpm ).rgb;
vec3 im10 = texture( ulmageUnit, vST-stp0 ).rgb;
vec3 ip10 = texture( ulmageUnit, vST+stp0 ).rgb;
vec3 i0m1 = texture( ulmageUnit, vST-st0p ).rgb;
vec3 i0p1 = texture( ulmageUnit, vST+st0p ).rgb;

vec3 blur = vec3(0.,0.,0.);
blur += 1.*(im1m1+ip1m1+ip1p1+im1p1);
blur += 2.*(im10+ip10+i0m1+i0p1);
blur += 4.*(i00);
blur /= 16.;

gl_FragColor = vec4( blur, 1. );
```



Oregon State University
Computer Graphics

mjb - December 26, 2024

Welcome to *superblur!*

5

superblur.glib

```
##OpenGL GLIB
Ortho -1. -1. 1. 1.
LookAt 0 0.5 0 0 0 0 1 0

Texture2D 5 grid.bmp

Background 0 0 0 0
Clear
Vertex superblur.vert
Fragment superblur.frag
Program SuperBlur

uHalfSize <0 0 11>
ImageUnit 5

Scale 0.5
QuadXY .2 3.
```

Welcome to *superblur!*

6

superblur.frag

```
#version 330 compatibility
uniform sampler2D ImageUnit;
uniform int uHalfSize;
in vec2 vST;

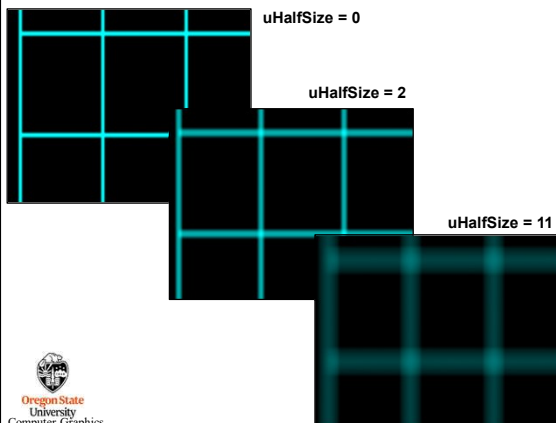
void main()
{
    ivec2 ires = textureSize( ImageUnit, 0 );
    float ds = 1. / float( ires.s );
    float dt = 1. / float( ires.t );

    int halfSize = uHalfSize / 01; // be sure it's an odd number
    vec3 rgbSum = vec3( 0., 0., 0. );
    float maxDistance = length( vec2( float(halfSize), float(halfSize) ) );
    maxDistance += 1.; // don't want a zero weight
    float weightSum = 0.;

    for( int i = -halfSize; i <= halfSize; i++ )
    {
        float di = float(i);
        for( int j = -halfSize; j <= halfSize; j++ )
        {
            float dj = float(j);
            vec2 st = vST + vec2( di*ds, dj*dt );
            float weight = maxDistance - length( vec2(di,dj) );
            rgbSum += weight*texture(ImageUnit,st).rgb;
            weightSum += weight;
        }
    }
    rgbSum /= weightSum;
    gl_FragColor = vec4( rgbSum, 1. );
}
```

Welcome to *superblur!*

7

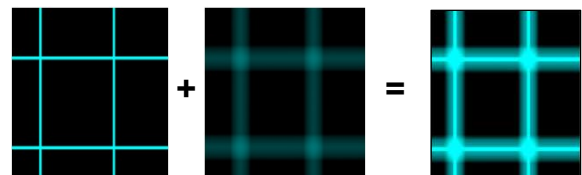


Bloom

8

Bloom is a two-pass algorithm designed to create a "glowing" effect.

The first pass does a blur on the image.
The second pass adds the blur image to the original image.



Bloom
9

bloom.glsl

```

##OpenGL GLSL
Ortho -1. -1. 1. 1.
LookAt 0 0.5 0 0 0 0 1 0

Texture2D 5 grid.bmp
Texture2D 6 512 512
RenderToTexture 6
Background 0 0.5 0
Clear
Vertex superblur.vert
Fragment superblur.frag
Program SuperBlur
    uHalfSize <0 0 11>
    ImageUnit 5

QuadXY .2 3.

RenderToTexture
Background 0. 0.0 0
Clear
Ortho

Texture2D 7 grid.bmp
Vertex bloom.vert
Fragment bloom.frag
Program Bloom
    BlurUnit 6
    ImageUnit 7
    uAdd <0. 0. 4.00>

QuadXY .2 5.
        
```

Computer Graphics

Bloom
10

bloom.frag

```

#version 330 compatibility

uniform sampler2D BlurUnit;
uniform sampler2D ImageUnit;
uniform float uAdd;

in vec2 vST;

void main()
{
    vec3 rgb = texture( ImageUnit, vST ).rgb;
    vec3 rgbblur = texture( BlurUnit, vST ).rgb;
    gl_FragColor = vec4( rgb + uAdd*rgbblur, 1. );
}
        
```

Computer Graphics

Bloom
(uHalfSize=11)
11

Computer Graphics