1

Stripes, Rings, and Dots!

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

2

Cartesian (X) Stripes

stripes.glib

```
##OpenGL GLIB

Perspective 90
LookAt  0 0 2   0 0 0   0 1 0

Vertex      stripes.vert
Fragment  stripes.frag
Program    Stripes                          \
                    uA <0 1. 10>             \
                    uP <0. .25 1.>           \
                    uTol <0. 0. .5>          \
                    uAmp <-5. 0. 5.>         \
                    uFreq <0. 10. 20.>
Color    1  0.5   0
Sphere 1 200 200
```

3

Cartesian (X) Stripes

stripes.vert

```
#version 330 compatibility

uniform float  uAmp;
uniform float  uFreq;

out vec3  vColor;
out float  vX, vY;
out float  vLightIntensity;

const vec3 LIGHTPOS = vec3( 0., 0., 10. );

void
main( )
{
        vec3 tnorm = normalize( gl_NormalMatrix * gl_Normal );
        vec3 ECposition = ( gl_ModelViewMatrix * gl_Vertex ).xyz;
        vLightIntensity  = abs( dot( normalize(LIGHTPOS - ECposition), tnorm ) );

        vColor = gl_Color.rgb;
        vec3 MCposition = gl_Vertex.xyz;
        vX = MCposition.x;
        vY = MCposition.y;

        vX = vX + uAmp * sin( uFreq * vY );

        gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
}
```


4

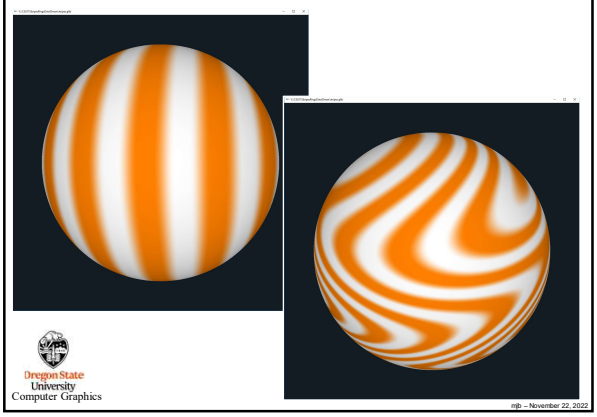
Cartesian (X) Stripes

stripes.frag

```
#version 330 compatibility

uniform float uA;
uniform float uP;
uniform float uTol;

in float  vX, vY;
in vec3  vColor;
in float  vLightIntensity;

const vec3 WHITE = vec3( 1., 1., 1. );

void
main( )
{
        float f = fract( uA*vX );

        float t = smoothstep( 0.5-uP-uTol, 0.5-uP+uTol, f )  -  smoothstep( 0.5+uP-uTol, 0.5+uP+uTol, f );
        vec3 rgb = vLightIntensity  * mix( WHITE, vColor, t );
        gl_FragColor = vec4( rgb, 1. )p;
}
```

5

Cartesian (X) Stripes



6

Rings

rings.glib

```
##OpenGL GLIB

Perspective 90
LookAt  0 0 2   0 0 0   0 1 0

Vertex      rings.vert
Fragment  rings.frag
Program   Rings                              \
                    uA <0 5. 10>             \
                    uP <0. .25 1.>           \
                    uTol <0. 0. .5>
Color    1 0.5 0
Sphere 1 200 200
```

Rings

7

rings.vert

```
#version 330 compatibility

uniform float  uAmp;
uniform float  uFreq;

out vec3  vColor;
out float vX, vY;
out float vLightIntensity;

const vec3 LIGHTPOS = vec3( 0., 0., 10. );

void
main( )
{
	vec3 tnorm = normalize( gl_NormalMatrix * gl_Normal );
	vec3 ECposition = ( gl_ModelViewMatrix * gl_Vertex ).xyz;
	vLightIntensity  = abs( dot( normalize(LIGHTPOS - ECposition), tnorm ) );

	vColor = gl_Color.rgb;
	vec3 MCposition = gl_Vertex.xyz;
	vX = MCposition.x;
	vY = MCposition.y;

	gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
}
```

Oregon State University Computer Graphics

mjb – November 22, 2022

Rings

8

rings.frag

```
#version 330 compatibility

uniform float uA;
uniform float uP;
uniform float uTol;

in float  vX, vY;
in vec3  vColor;
in float  vLightIntensity;

const vec3 WHITE = vec3( 1., 1., 1. );

void
main( )
{
	float r = sqrt( vX*vX + vY*vY );
	float rfrac = fract( uA*r );

	float t =    smoothstep( 0.5-uP-uTol, 0.5-uP+uTol, rfrac ) -
		     smoothstep( 0.5+uP-uTol, 0.5+uP+uTol, rfrac );		// "smoothpulse"

	vec3 rgb = vLightIntensity *  mix( WHITE, vColor, t );
	gl_FragColor = vec4( rgb, 1. );
}
```

Oregon State University Computer Graphics

mjb – November 22, 2022

Rings (= Polar Stripes)

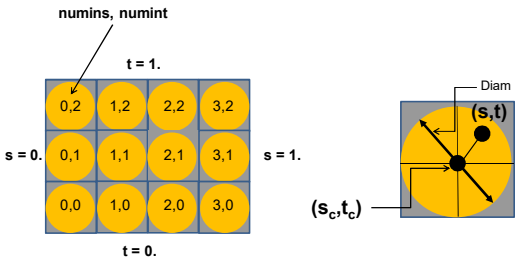
9

Oregon State University Computer Graphics

mjb – November 22, 2022

Circular Dots are a "Local Pattern"

10



$$(s-s_c)^2+(t-t_c)^2 \le \left(\frac{Diam}{2.}\right)^2$$

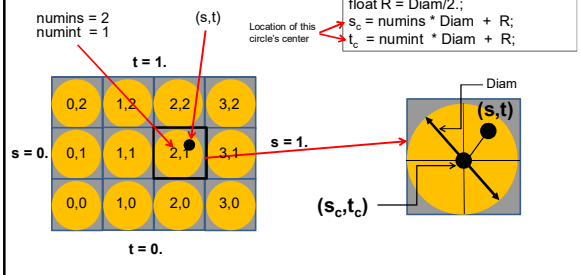
Oregon State University Computer Graphics

mjb – November 22, 2022

Circular Dots

11

```
int numins = int( vST.s / Diam );
int numint  = int( vST.t  / Diam );
float R = Diam/2.;
s_c = numins * Diam  +  R;
t_c = numint  * Diam  +  R;
```



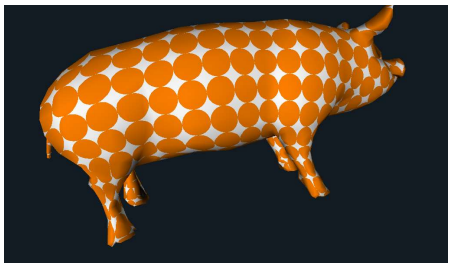
$$(s-s_c)^2+(t-t_c)^2 \le (R)^2$$

Oregon State University Computer Graphics

mjb – November 22, 2022

Circular Dots

12



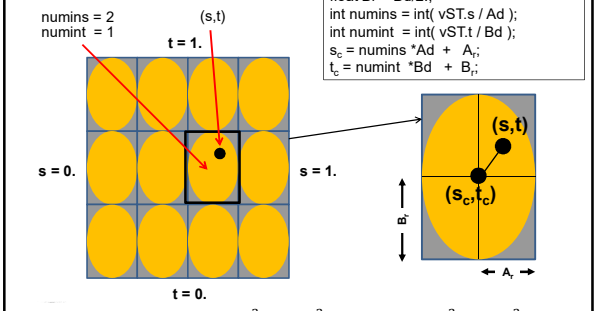
Oregon State University Computer Graphics

mjb – November 22, 2022

Elliptical Dots

13

```
float Ar = Ad/2.;
float Br = Bd/2.;
int numins = int( vST.s / Ad );
int numint  = int( vST.t / Bd );
sc = numins *Ad  +  Ar;
tc = numint  *Bd  +  Br;
```



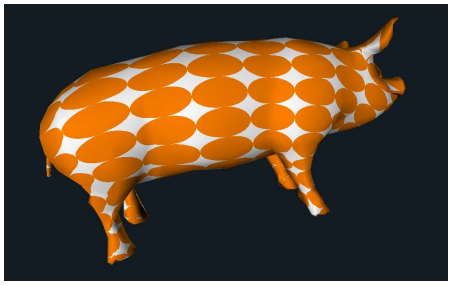
$(s-s_c)^2+(t-t_c)^2 \le R^2 \Rightarrow \left(\frac{s-s_c}{R}\right)^2+\left(\frac{t-t_c}{R}\right)^2 \le 1 \Rightarrow \left(\frac{s-s_c}{A_r}\right)^2+\left(\frac{t-t_c}{B_r}\right)^2 \le 1$

Oregon State University Computer Graphics — Circle — Circle — Ellipse

mjb – November 22, 2022

Elliptical Dots

14

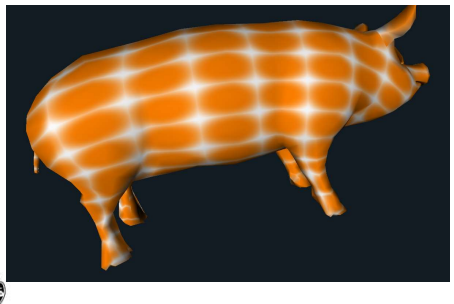


Oregon State University Computer Graphics

mjb – November 22, 2022

Elliptical Dots with Tolerance

15

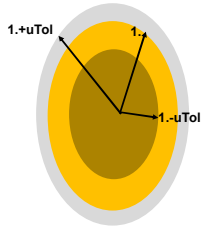


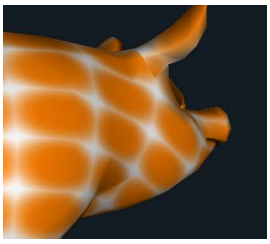
Oregon State University Computer Graphics

mjb – November 22, 2022

Elliptical Dots with Tolerance

16





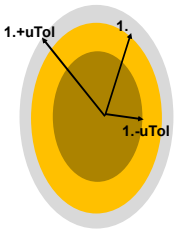
$$1-uTol \le \left(\frac{s-s_c}{A_r}\right)^2+\left(\frac{t-t_c}{B_r}\right)^2 \le 1+uTol$$

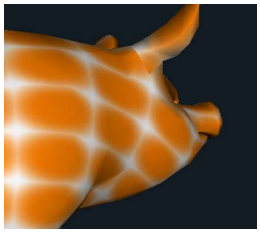
Oregon State University Computer Graphics

mjb – November 22, 2022

Elliptical Dots with Tolerance

17





$1-uTol \le \left(\frac{s-s_c}{A_r}\right)^2+\left(\frac{t-t_c}{B_r}\right)^2 \le 1+uTol$

$float\ d = \left(\frac{s-s_c}{A_r}\right)^2+\left(\frac{t-t_c}{B_r}\right)^2$

Inside the ellipse, d < 1.
At the boundary of the ellipse, d = 1.
Outside the ellipse, d > 1.

```
float t = smoothstep( 1.-uTol,  1.+uTol, d );
vec3 color = mix( ORANGE, WHITE, t );
```

Oregon State University Computer Graphics

mjb – November 22, 2022

Soon we will see how to create elliptical dots with Noise!

18

Oregon State University Computer Graphics

mjb – November 22, 2022