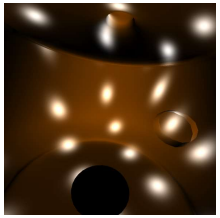
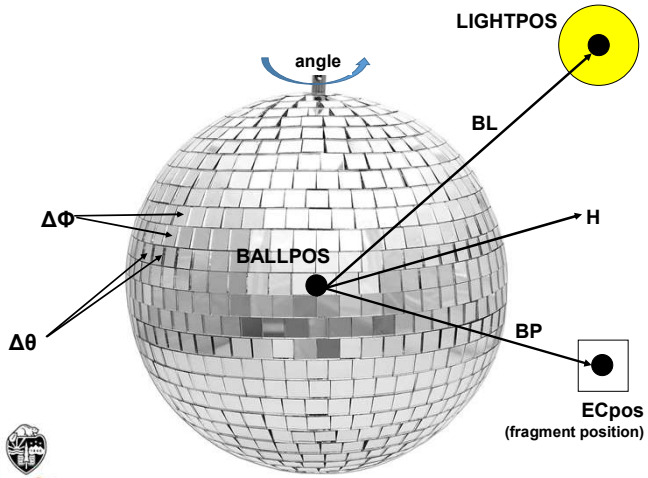
Disco Ball Lighting!

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





Oregon State University
Computer Graphics

disco.glib

```
##OpenGL GLIB

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

Vertex      disco.vert
Fragment disco.frag
Program  Disco Program  Disco                          \
                uNumFacets <5 15 50>                    \
                uPower <1000. 5000. 50000.>

Color 1.   0.5   0.
Teapot
```

Oregon State University
Computer Graphics

disco.vert

```
#version 330 compatibility

out vec3  vECpos;
out vec4  vColor;
out float   vLightIntensity;

const vec3 LIGHTPOS = vec3( 2., 0., 0. );

void
main( )
{
        vECpos = ( gl_ModelViewMatrix * gl_Vertex ).xyz;

        vec3 tnorm = normalize( vec3( gl_NormalMatrix * gl_Normal ) );
        vLightIntensity  = dot( normalize(LIGHTPOS - vECpos), tnorm );
        vLightIntensity = abs( vLightIntensity );

        vColor = gl_Color;

        gl_Position = gl_ModelViewProjectionMatrix * gl_Vertex;
}
```

Oregon State University
Computer Graphics

5

disco.frag, I

```
#version 330 compatibility

in vec3  vECpos;
in vec4  vColor;
in float  vLightIntensity;

uniform int    uNumFacets;
uniform float uPower;
uniform float Timer;          // built-in to glman

const float PI = 3.14159265;
const vec3 BALLPOS      = vec3( 0., 2., 0. );
const vec3 LIGHTPOS     = vec3( 2., 0., 0. );
const vec3 LIGHTCOLOR = vec3( 1., 1., 1. );

void
main( void )
{
      int numTheta = uNumFacets;              // # in longitude direction
      int numPhi    = uNumFacets;             // # in latitude direction
      float dtheta = 2. * PI / float(numTheta);
      float dphi   =       PI / float(numPhi);
                  // spherical coord angles between the facets
      vec3 BP = normalize( vECpos - BALLPOS );          // vector from ball center to the
                                                        // point we care about
      float angle = radians(Timer*360.);                // ball rotation angle
      float c = cos( angle );
      float s = sin( angle );
      vec3 bp;
      bp.x =  c*BP.x + s*BP.z;
      bp.y =  BP.y;
      bp.z = -s*BP.x + c*BP.z;                          // but, rotate the vector, not the ball
```


5

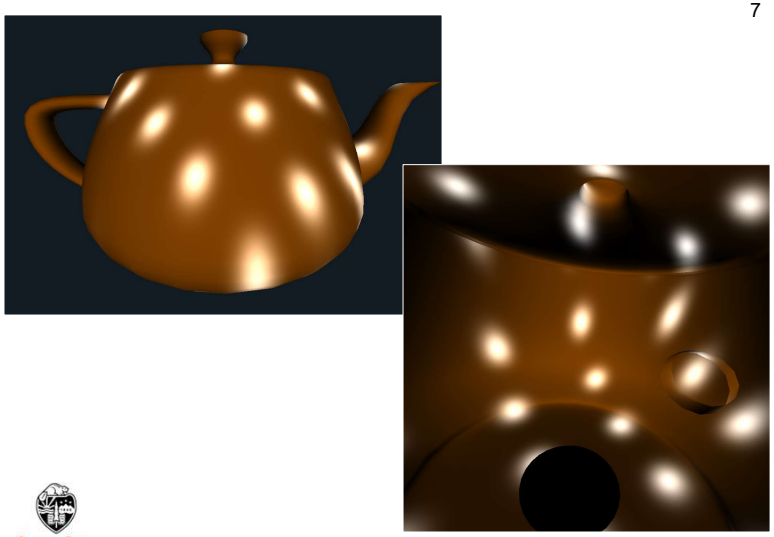
6

disco.frag, II

```
      vec3 BL = normalize( LIGHTPOS - BALLPOS );          // vector from the ball center
                                                          // to the light
      vec3  H = normalize( BL + bp  );
                  // vector halfway between BL and bp – if a facet aligns with this angle,
                  // the point we care about will get a lot of light
      float xz    = length( H.xz );
      float phi   = atan( H.y, xz );
      float theta = atan( H.z, H.x );       // turn the H vector into spherical coordinates

      int itheta = int( floor(  ( theta + dtheta/2. ) / dtheta  ) );
      int iphi   = int( floor(  ( phi   + dphi/2.  )   / dphi    ) );

      float theta0 = dtheta * float(itheta);
      float phi0   = dphi    * float(iphi);          // figure out what the closest facet to H is

      vec3 N0;
      N0.y = sin(phi0);
      xz    = cos(phi0);
      N0.x = xz*cos(theta0);
      N0.z = xz*sin(theta0);                // N0 is the discrete facet normal vector

      float d = max( dot( N0, H ), 0. );    // like the cone angle on a spotlight
      const float DMIN = 0.990;             // acos(0.990) is about 8 degrees
      if( d < DMIN )
            d = 0.;
      d = pow( d, uPower );                 // specular brightness

      gl_FragColor = vec4( vColor.rgb*vLightIntensity  +  d * LIGHTCOLOR, 1. ) ;
                              // diffuse + specular
}
```


6

7



7