20140559 조형준, HW4

1. Shading

   A. Phong Shading

      Lab 3에서 사용한 VertexShader와 FragmentShader의 Phong reflection model 부분 코드를 그대로 인용하였습니다.

      (VertexShader.glsl, FragmentShader.glsl)

   B. Flat Shading

      Phong Shading에서 fragmentNormal의 type에 flat을 추가하였습니다.

      (VertexShader2.glsl, FragmentShader2.glsl)

   C. Toon Shading

      intensity를 계산한 후에 intensity의 범위에 따라서 color를 정해주었습니다.

```
if (intensity.z > 0.95)
        color = vec3(0.35, 0.6, 1.0);
else if (intensity.z > 0.5)
        color = vec3(0.21, 0.36, 0.6);
else if (intensity.z > 0.25)
        color = vec3(0.14, 0.24, 0.4);
else
        color = vec3(0.07, 0.12, 0.2);
```

      (VertexShader2.glsl, FragmentShader2.glsl)

2. Lighting

   모든 Lighting은 fragment shader에서 구현하였습니다. 키보드에서 1을 입력하면 Directional Light의 on/off, 2를 입력하면 Point Light의 on/off, 3을 입력하면 Spot Light의 on/off를 조정할 수 있습니다.

   A. Directional Light

      기존에 있던 uLight를 지우고, 새로운 uniform variable들을 사용했습니다.

      (uniform vec3 Color, uniform float Intensity, uniform vec3 Direction, uniform int dL)

uniform vec3 Color는 Directional Light의 color를 담고 있고, uniform float Intensity는 Directional Light의 intensity를 담고 있으며, uniform vec3 Direction은 Directioal Light의 방향을 담고 있습니다. uniform int dL은 Directional Light가 켜져 있는지 나타냅니다.

```
if (dL == 1)
{
        vec3 tolightd = normalize(-Direction);
        vec3 hd = normalize(toV + tolightd);

        float speculard = pow(max(0.0, dot(hd, normal)), 64.0);//r dot v == n dot h
        float diffused = max(0.0, dot(normal, tolightd));//l dot n

        vec3 intensityd = fragmentColor *diffused + vec3(0.6, 0.6, 0.6)*speculard;
        intensityd = intensityd * Intensity;

        colord = pow(intensityd, vec3(1.0 / 2.2)) * Color;
}
```

toon shading에서는 마지막 줄인 colord ...가 없습니다.

B. Point Light

기존에 있던 uLight를 지우고, 새로운 uniform variable들을 사용했습니다.

(uniform vec3 pColor, uniform float pIntensity, uniform vec3 pLocation, uniform vec3 Fallout, uniform int pL)

uniform vec3 pColor는 Point Light의 color를 담고 있고, uniform float pIntensity는 Point Light의 intensity를 담고 있으며, uniform vec3 pLocation은 Point Light의 위치를 담고 있습니다. uniform vec3 Fallout은 fallout function의 0차항, 1차항, 2차항 정보를 갖고 있습니다. uniform int pL은 Point Light가 켜져 있는지 나타냅니다.

```
if (pL == 1)
```

```
        {
                vec3 tolightp = normalize(pLocation - fragmentPosition);
                vec3 hp = normalize(toV + tolightp);

                float specularp = pow(max(0.0, dot(hp, normal)), 64.0);//r dot v == n dot h
                float diffusep = max(0.0, dot(normal, tolightp));//l dot n

                vec3 intensityp = fragmentColor *diffusep + vec3(0.6, 0.6, 0.6)*specularp;
                intensityp = intensityp * pIntensity;
                float d = length(pLocation - fragmentPosition);
                float att = 1.0 / (Fallout.x + Fallout.y * d + Fallout.z * d * d);
                intensityp = intensityp * att;

                colorp = pow(intensityp, vec3(1.0 / 2.2)) * pColor;
        }
```

toon shading에서는 마지막 줄인 colorp ...가 없습니다.

C. Spot Light

기존에 있던 uLight를 지우고, 새로운 uniform variable들을 사용했습니다.

(uniform vec3 sColor, uniform float sIntensity, uniform vec3 Axis, uniform vec3 sLocation, uniform float Radius, uniform vec3 Fallout, uniform int sL)

uniform vec3 sColor는 Spot Light의 color를 담고 있고, uniform float sIntensity는 Spot Light의 intensity를 담고 있으며, uniform vec3 Axis는 Spot Light의 방향, uniform vec3 sLocation은 Spot Light의 위치를 담고 있습니다. uniform float Radius는 Spot Light의 각도를 나타냅니다. uniform vec3 Fallout은 fallout function의 0차항, 1차항, 2차항 정보를 갖고 있습니다. uniform int sL은 Spott Light가 켜져 있는지 나타냅니다.

```
        if (sL == 1)
```

```
        {
            vec3 tolights = normalize(sLocation - fragmentPosition);
            vec3 hs = normalize(toV + tolights);

            float speculars = pow(max(0.0, dot(hs, normal)), 64.0);//r dot v == n dot h
            float diffuses = max(0.0, dot(normal, tolights));//l dot n
            vec3 intensitys;
            if (dot(tolights, normalize(-Axis)) > cos(Radius))
                intensitys = fragmentColor *diffuses + vec3(0.6, 0.6, 0.6)*speculars;
            else
                intensitys = vec3(0.0,0.0,0.0);
            intensitys = intensitys * sIntensity;
            float d = length(sLocation - fragmentPosition);
            float att = 1.0 / (Fallout.x + Fallout.y * d + Fallout.z * d * d);
            //float spotatt;
            intensitys = intensitys * att;

            colors = pow(intensitys, vec3(1.0 / 2.2)) * sColor;
        }
```

toon shading에서는 마지막 줄인 colors ...가 없습니다.

D. Light Animation

각각 Light에 대해, 그 light가 켜져 있다면 정해진 축을 기준으로 draw를 할 때 마다 0.02 radian씩 회전하도록 하였습니다.

E. Arcball

sky frame에서 회전이 일어날 때, lighting이 fixed location(direction)을 유지하도록 하기 위해 각각 light의 direction, location을 sky camera의 회전과 대비되도록 회전시켜 주었습니다.

~~~ = glm::vec3(glm::inverse(skyRBT) * glm::inverse(aFrame * m * glm::inverse(aFrame)) * skyRBT * ~~~;

3. Creativity

A. object color change

draw가 일어날 때 마다 Phong shading과 Flat shading을 적용한 토끼의 색깔을 바꾸었습니다. (r 값을 0.001f씩 증가시켰습니다.)

B. object rotation

HW2에서 했던 것처럼 각 object의 회전을 구현하였습니다. 키보드 O key를 이용하여 object change가 가능합니다.
~~~