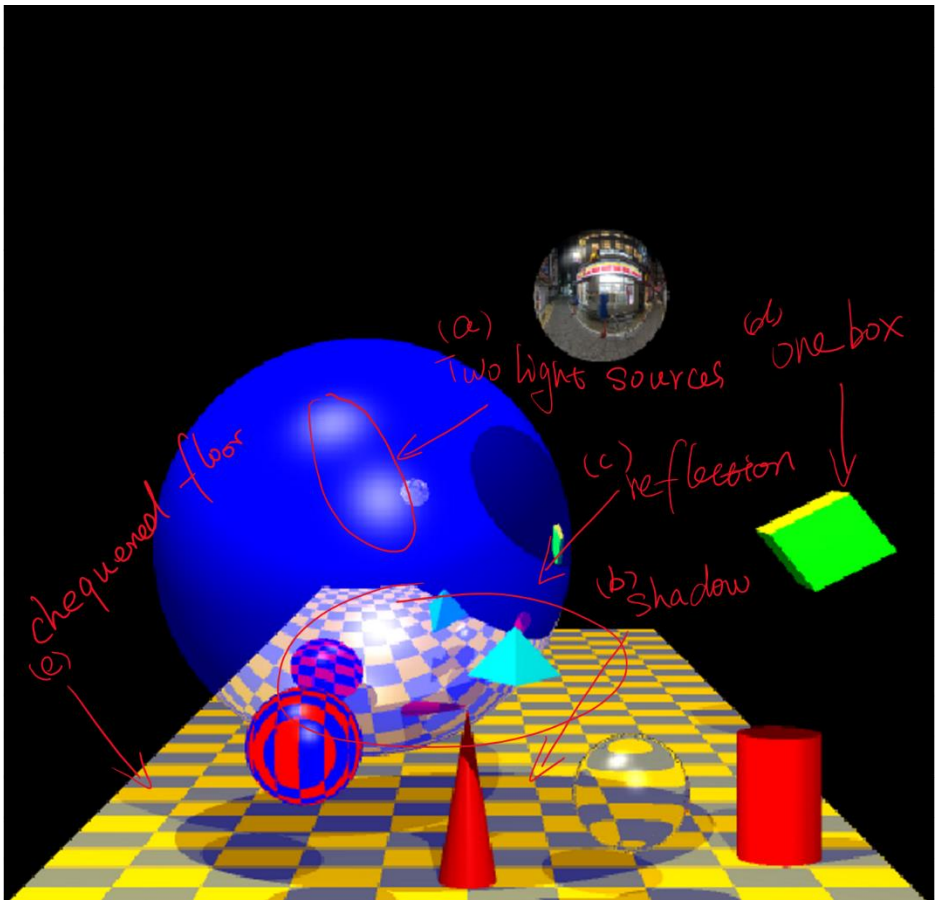
Cosc363 assignment 2

Enyang Zhang(72004622)

In assignment 2, several feature of ray tracer has been implemented.
Using qt creator to compile
Config qt by projects->build&run->working directory->change working directory to "code" folder to connect image

The Basic Ray Tracer:

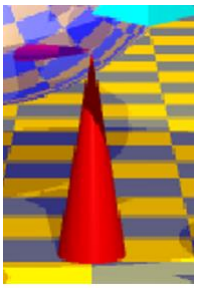
Basic requirements are shown on the picture below.



(a) The light source has been implemented. The diffuse specular reflection is generated by light source. (From lab7).
(b) Shadow has been implemented. If l.n or either shadow.xindex>-1 or shadow.xdist < lightDist satisfied, we add up ambient diffuse and specular components to get shadow. (from lab8)
(c) The biggest blue sphere has been implemented the reflection by using the code provided from lab8 and the ray.xindex is 0.
(d) The box shown in the picture is created by using 5 planes.
(e) The chequered floor has been implemented as shown in the picture. To do this, ray.xpt.x and ray.xpt.z is been mod 2, so that if they both zero or not zero, then they will be assigned blue colour and other case they will be assigned yellow.

Extensions:

(a)1. A cone has been created using equation provided in lecture notes.



$$(x - x_c)^2 + (z - z_c)^2 = \left(\frac{R}{h}\right)^2 (h - y - y_c)^2$$

$$\tan\theta = \frac{R}{h}$$

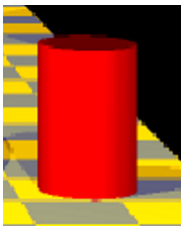
- Surface normal vector (normalized):

$$\boldsymbol{n} = (\sin\alpha\cos\theta, \sin\theta, \cos\alpha\cos\theta)$$

Class Cone calculated the intersection and normal for trace ray.

By calculate two roots of it t1 and t2 is $-b \pm \frac{\sqrt{delta}}{2*a}$ where $delta = b^2 - 2ac$

In this way, we have intersection point.



2. A Cylinder has been created using equation provided in lecture notes

$$(x - x_c)^2 + (z - z_c)^2 = R^2$$

A cylinder at (x_c, y_c, z_c), with axis parallel to t
y-axis, radius R and height h is given by

$(x - x_c)^2 + (z - z_c)^2 = R^2$

$0 \le (y - y_c) \le h$

Normal vector at (x, y, z) p_0

(un-normalized) $\boldsymbol{n} = (x - x_c,\ 0,\ z - z_c)$

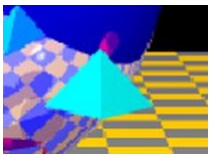
(normalized) $\boldsymbol{n} = ((x - x_c)/R,\ 0,\ (z - z_c)/R)$

ay equation:

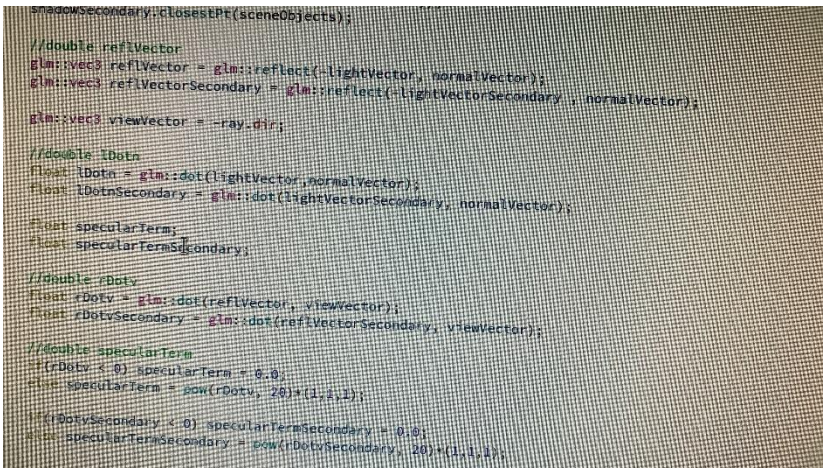
$x = x_0 + d_x t;$ $y = y_0 + d_y t;$ $z = z_0 + d_z t;$

Use the same method to calculate the intersection point as cone.

3. A tetrahedron has been created by setting one plane to have two same vertex. And calculate the coordination of four vertex. But the top vertex is on the middle of side of original vertex.



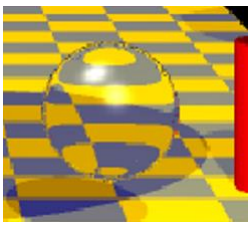
(b)Multiple light sources are implemented, using the same way as first light source. Therefore, all procedure of light source has been doubled like what it is in picture.



```
shadowSecondary.closestPt(sceneObjects);

//double reflVector
glm::vec3 reflVector = glm::reflect(-lightVector, normalVector);
glm::vec3 reflVectorSecondary = glm::reflect(-lightVectorSecondary, normalVector);

glm::vec3 viewVector = -ray.dir;

//double lDotn
float lDotn = glm::dot(lightVector,normalVector);
float lDotnSecondary = glm::dot(lightVectorSecondary, normalVector);

float specularTerm;
float specularTermSecondary;

//double rDotv
float rDotv = glm::dot(reflVector, viewVector);
float rDotvSecondary = glm::dot(reflVectorSecondary, viewVector);

//double specularTerm
if(rDotv < 0) specularTerm = 0.0;
else specularTerm = pow(rDotv, 20)*(1,1,1);

if(rDotvSecondary < 0) specularTermSecondary = 0.0;
else specularTermSecondary = pow(rDotvSecondary, 20)*(1,1,1);
```

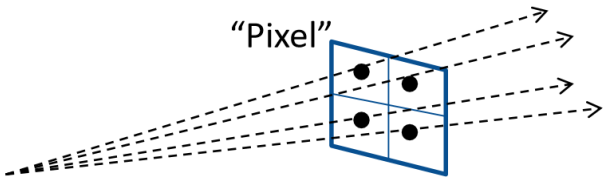
(c)(d)transparent and refraction has been implemented for sphere 3. According to lecture notes.

```
n = sceneObjects[ray.xindex]->normal(ray.xpt);
g = glm::refract(d, n, eta);
Ray refrRay(ray.xpt, g)
refrRay.closestPt(sceneObjects);
m = sceneObjects[refrRay.xindex]->normal(refrRay.xpt);
h = glm::refract(g, –m, 1.0f/eta);
```

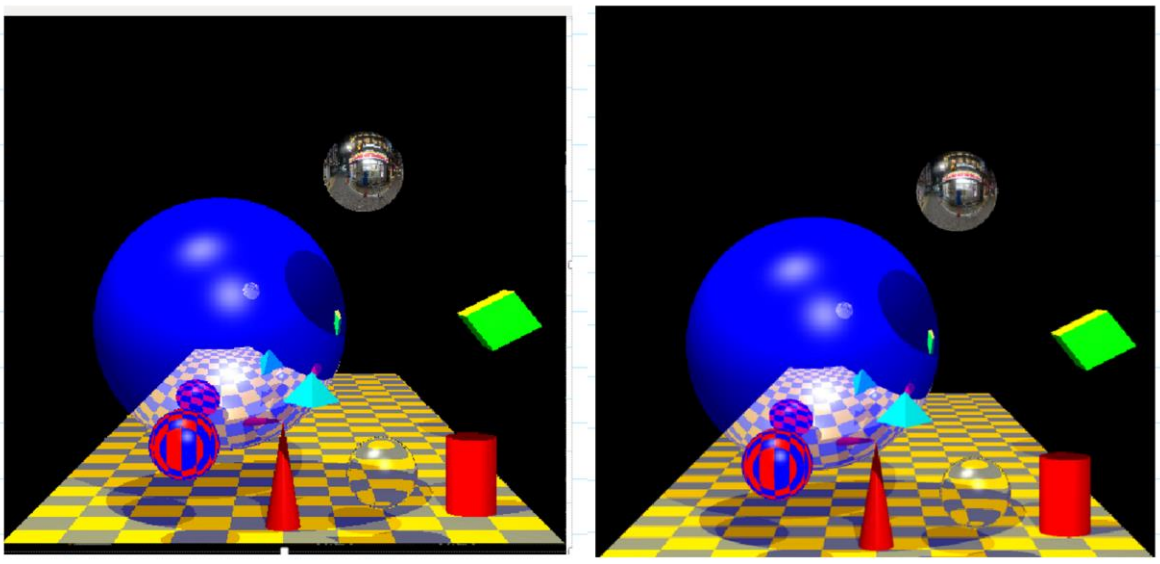


The ETA = 1/1.01

(e)Anti-aliasing has been implemented, according to lecture notes,



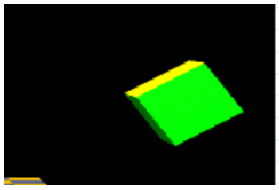
Instead we calculate the colour for just once, we make 4 ray go through four parts of the pixel and calculate the average colour index.



Due to jaggedness along edges of polygons and shadows, we using super sampling to make the edge more smooth.

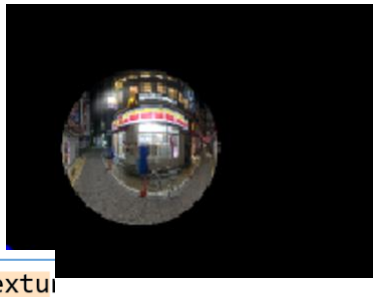
(f) The box has been implemented rotation using rotational matrix

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



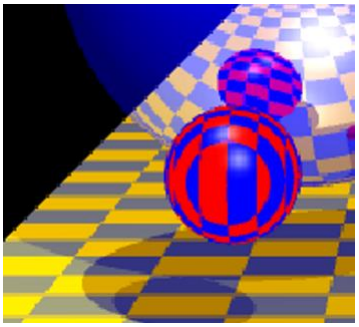
So that the box rotated along z-axis 30 degree in anti-clockwise direction.

(g) A non-planar object textured using an image has been implemented for sphere 2. (Note this is texture not reflection). To do this, use the way provided in lecture notes.



```
if(ray.xindex == 3)   //Assuming the plane to be textu
{
   texcoords = (ray.xpt.x - a1)/(a2-a1);
   texcoordt = (ray.xpt.y - b1)/(b2-b1);
   col = texture.getColorAt(texcoords, texcoordt);
}
```

(h) A non-planar object textured using a procedural pattern has been implemented. To do this, I used the same way as chequered floor. And texture the sphere 2.



Reference:

www.textrues.comtexture

https://en.wikipedia.org/wiki/Rotation_matrix

lecture 9 notes

lab7 – 10