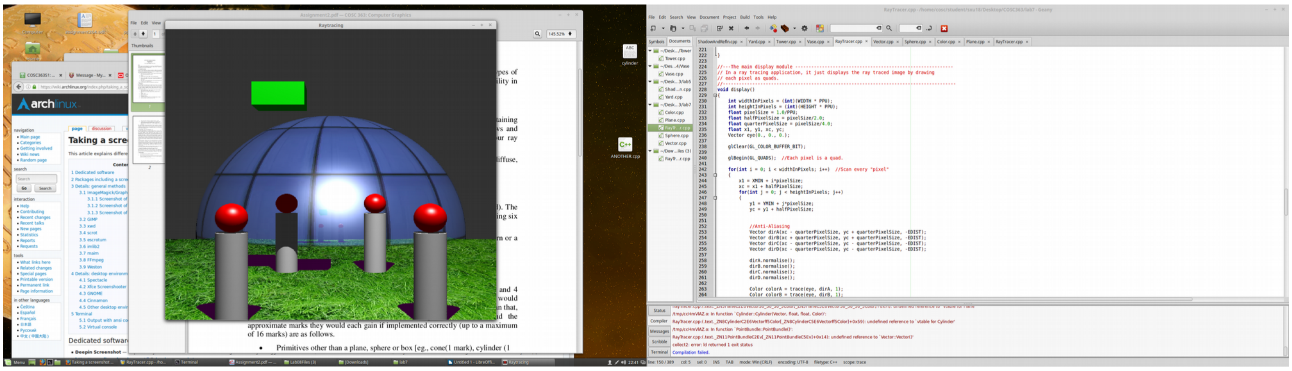
Assignment 2 report

Nmae: Suyuan
ID: 47784151

This assignment is working on building image models and then mapping every element of the image to each pixel I build for completing ray tracing.

Generally, the screen shot image below is what my programme generated in about 12 seconds.



```
}

//---The main display module -----------------------------------------------------
// In a ray tracing application, it just displays the ray traced image by drawing
// each pixel as quads.
//-----------------------------------------------------------------------------------
void display()
{
    int widthInPixels = (int)(WIDTH * PPU);
    int heightInPixels = (int)(HEIGHT * PPU);
    float pixelSize = 1.0/PPU;
    float halfPixelSize = pixelSize/2.0;
    float quarterPixelSize = pixelSize/4.0;
    float x1, y1, xc, yc;
    Vector eye(0., 0., 0.);

    glClear(GL_COLOR_BUFFER_BIT);

    glBegin(GL_QUADS);  //Each pixel is a quad.

    for(int i = 0; i < widthInPixels; i++)  //Scan every "pixel"
    {
        x1 = XMIN + i*pixelSize;
        xc = x1 + halfPixelSize;
        for(int j = 0; j < heightInPixels; j++)
        {
            y1 = YMIN + j*pixelSize;
            yc = y1 + halfPixelSize;


            //Anti-Aliasing
            Vector dirA(xc - quarterPixelSize, yc + quarterPixelSize, -EDIST);
            Vector dirB(xc + quarterPixelSize, yc + quarterPixelSize, -EDIST);
            Vector dirC(xc - quarterPixelSize, yc - quarterPixelSize, -EDIST);
            Vector dirD(xc - quarterPixelSize, yc - quarterPixelSize, -EDIST);

            dirA.normalise();
            dirB.normalise();
            dirC.normalise();
            dirD.normalise();

            Color colorA = trace(eye, dirA, 1);
            Color colorB = trace(eye, dirB, 1);
```

```
/tmp/ccHmVlAZ.o: In function `Cylinder::Cylinder(Vector, float, float, Color)':
RayTracer.cpp:(.text._ZN8CylinderC2E6Vectorff5Color[_ZN8CylinderC5E6Vectorff5Color]+0x59): undefined reference to `vtable for Cylinder'
/tmp/ccHmVlAZ.o: In function `PointBundle::PointBundle()':
RayTracer.cpp:(.text._ZN11PointBundleC2Ev[_ZN11PointBundleC5Ev]+0x14): undefined reference to `Vector::Vector()'
collect2: error: ld returned 1 exit status
Compilation failed.
```

1. I set two light sources for my program, they are light1 and light2, both of them are located above of the other models, and are made to generate shadow together.
However, In order to make it easier and avoid to much repeated work, I did not make the second light source(light2) to implement other features as what light does, such as specular reflection.

2. The biggest located behind is the only reflective object in my program.

```
if(q.index == 0 && step < MAX_STEPS){
        Vector reflectionVector = pos - q.point;
        float nDotv = n.dot(reflectionVector);
        reflectionVector = (n*2)*nDotv - reflectionVector;
        reflectionVector.normalise();

        Color reflectionCol = trace(q.point, reflectionVector, step+1);
        colorSum.combineColor(reflectionCol, reflCoeff);


}
```

the code segment is how I make it reflects others objects. The main idea is calculating the reflection ray first, the closest point on this ray blocked by other object will be the point that affects the current pixel's colour.

Although my program only has one reflective object, I write my code to recursively call Trace() function. Because every other reflective has to be considered as well.

3. there is a box consists of 6 planes in my program. Every side is created individually by using Plane class.

5. Two textures are implemented, one is applied to the biggest sphere, and the other is the ground.

```
Vector a = ground_vector1;
Vector b = ground_vector2;
Vector c = ground_vector3;
Vector d = ground_vector4;

float u = ((q.point - a).dot(b-a)/(b-a).length()/(b-a).length());
float v = ((q.point - d).dot(a-d)/(a-d).length()/(a-d).length());
col = texture_ground.getColorAt(u,v);
colorSum.combineColor(col);
```

the code segment is how I mapped the vectexf the texture image to each corresponding vectex the ground plane.

6. I created another Object class's sub class, which is Cylinder.
To build this, I have to calculate out every point that intersects with the vector from eye.

The way to solve this is using the equation about variable t from lecture note, then transform it and make use of root representation of quadratic equations. Once I get two possible values of t. the additional comparisons are needed, as only the closest one is required.

7. Anti-aliasing.
Anti-aliasing is ….

Sorry that I can not finish my report, and my program part of the assignment was not cared enough because other courses' assignment cost a lot.