

Ray Tracing Project

The significant of this ray tracing project is that it had been done 5 basic features and 7 extension features, what is more the ray of object/components gets perfect reflection and refraction without any execution errors. However, there could be some more improvements such as Camera motion, constructive solid geometry, and bounding volumes. The estimate of the time taken by the program to run on lab computer is 35 seconds, and 2 min 15 seconds for run on my own laptop which is the worst case.

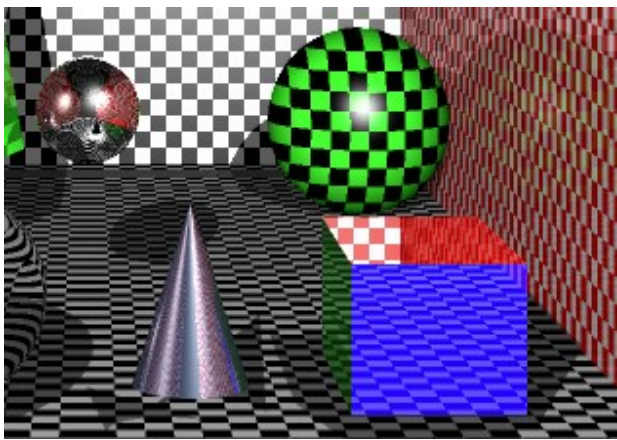
The features which achieved are listed below:

	Features				
Basics	Lighting	Shadows	Reflections	Box(Cube)	Textured floor
Extensions	Cylinder	Cone	Tetrahedron	Multiple Light Sources	
	Refractions	Textured Sphere		Anti-aliasing (Super Sampling)	

1. Lighting and Multiple Light Sources :

The lighting system of this ray tracing project is that included two lights inside of the scene. The two light provided two different shadows. The position of first light is $[-10, 15, -5]$ and the second light is $[16, 15, -50]$ so it like one on front and other one on the back. The shadow generates separately which give the two shadow works individual effect.

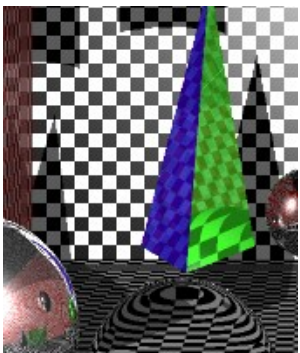
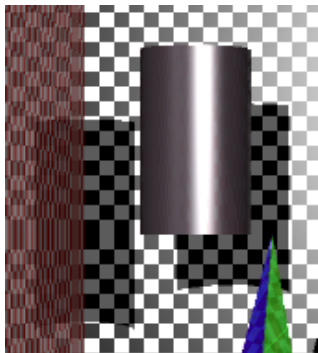
2. Shadows:



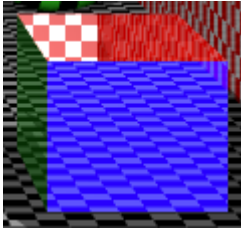
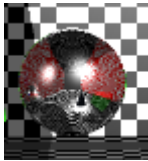
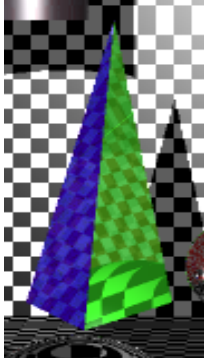
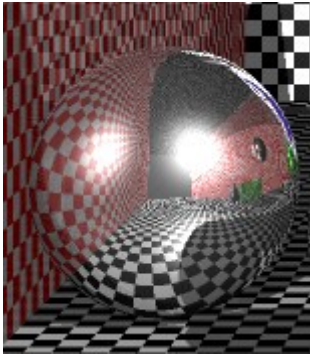
Since the scene had two individual light sources, hence the number of the shadow from the object should be two, which are showing on the picture.

The way of the lighting system providing shadow is trace a ray from the point of intersection towards the light source if the shadow ray hits the object surface and if the point of intersection is between the light source and the object then create only the ambient contribution of light.


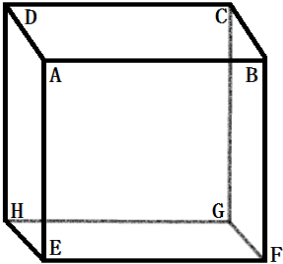
The colour of lights are create by phone shading which a function inside of color.cpp

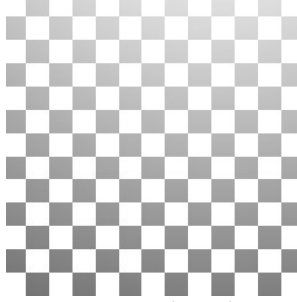
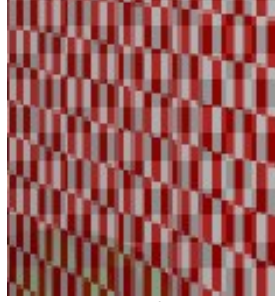
4. Reflections:

 <p>Box</p>	 <p>Sphere</p>	<p>The theory of reflections is found the object index which is q.point, if the surface of this object is reflective then a secondary ray along the direction of reflection is traced.</p> <p>And also if this secondary ray meets a surface of this object at point intensity then add it to the pixel colour. After finish tracing return the sum of pixel colour.</p> <p>The colour of the pixel is $I = I_A + p_r I_B$</p>
 <p>Tetrahedron</p>	 <p>Sphere</p>	


5. Box(Cube):

 <p>(a) Cube</p>	<p>This cube (a) is draw by 6 planes. The 6 planes are generated by 8 coordinates as showing in (b), the coordinates listed below:</p> <p>A[9,-14,-80] B[15,-14,-80] C[15,-14,-90] D[9,-14,-90] E[9,-19,-80] F[15,-19,-80] G[15,-19,-90] H[9,-19,-90]</p>
 <p>(b) Structure of Cube</p>	

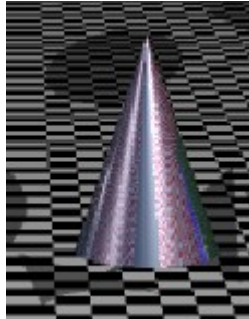
6. Textured plane:

 <p>(A) Floor/Top/Back</p>	 <p>(B) Left/Right</p>	<p>The textured plane has two colours one is black & white for floor and top, other one is red & white for left and right side.</p> <p>The algorithm of this texture is sum of the object position coordinates mod 2 which means if it's even then it set up White if it's odd set up Black/Red.</p>
---	---	--

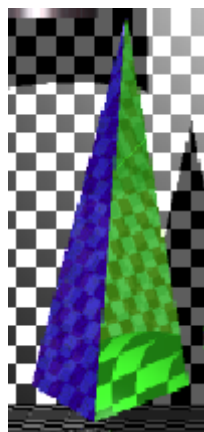
7. Cylinder:

 <p>Cylinder</p>	<p>The calculation of cylinder is basic on quadratic function, and the cylinder at the origin with axis along the center.y axis, radius and height.</p> <p>According to the formula: $x^2 + z^2 = R^2 \quad 0 \leq \text{center.y} \leq \text{height}$</p> <p>Quadratic function: $t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$</p> $a = d_x^2 + d_z^2$ $b = 2 \times d_x(x_0 - x_c)^2 + d_z(z_0 - z_c)^2$ $c = (x_0 - x_c)^2 + (z_0 - z_c)^2 - R^2$ <p>After intersected, normalise the vector at(x, y, z) by $n = (x - x_c, 0, z - z_c)$</p>
--	--

8. Cone:

 <p>Cone</p>	<p>The calculation of Cone is basic on quadratic function, and the Cone with base at the origin with axis parallel to the center.y axis, radius and height.</p> <p>According to the formula: $x^2 + z^2 = r^2 \quad 0 \leq \text{center.y} \leq \text{height}$</p> $x^2 + z^2 = \left(\frac{R}{h}\right)^2 (h - y)^2$ <p>Quadratic function: $t = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad \tan = \left(\frac{\text{radius}}{\text{height}}\right)^2$</p> $a = d_x^2 + d_z^2 - (\tan^2 \times d_y^2)$ $b = [2 * (x - x_c) \times \text{direction.x}] + [2 \times (z - z_c) \times \text{direction.z}] + [2 \times \tan^2 \times (\text{height} - y + y_c) \times \text{direction.y}]$ $c = (x - x_c)^2 + (z - z_c)^2 - \tan^2 \times (\text{height} - y + y_c)^2$ <p>After intersected, normalise the vector at(x, y, z) by $n = (x, r \times \tan, z)$</p>
---	--

9. Tetrahedron:



Tetrahedron

Similar as plane.cpp basic the way to generate a plane with 3 point triangle plane instead of 4 point square plane.

Intersection formula: $t = \frac{(p-p_0) \cdot n}{d \cdot n}$

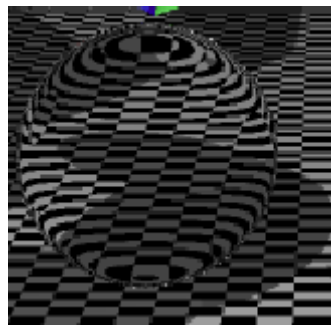
$ua = b-a$ $va = q-a$

$ub = c-b$ $vb = q-b$

$uc = a-c$ $vc = q-c$

If the point of ray inside the quad if and only if $(ua * va) \cdot n$, $(ub * vb) \cdot n$, and $(uc * vc) \cdot n$ are all positive then return Boolean value true, otherwise return false.

10. Refractions:



Air Bubble

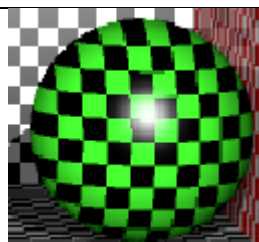
The theory of refraction working is when the surface of object is transparent, and then a secondary ray along the direction of refraction is traced. Moreover, if this secondary ray hits a surface of the object at a point with intensity, then add it into the pixel color.

The colour of the pixel is $I = I_A + p_c I_c$

If the ray hits inside the object, the refraction vector should calculated by $dir \times ratio - n \times [(ratio \times dir \cdot n) + Cos]$

If the ray get out of the object then the refraction vector should calculated by $dir \times ratio - (n \times -1) \times [(ratio \times dir \cdot n) + Cos]$

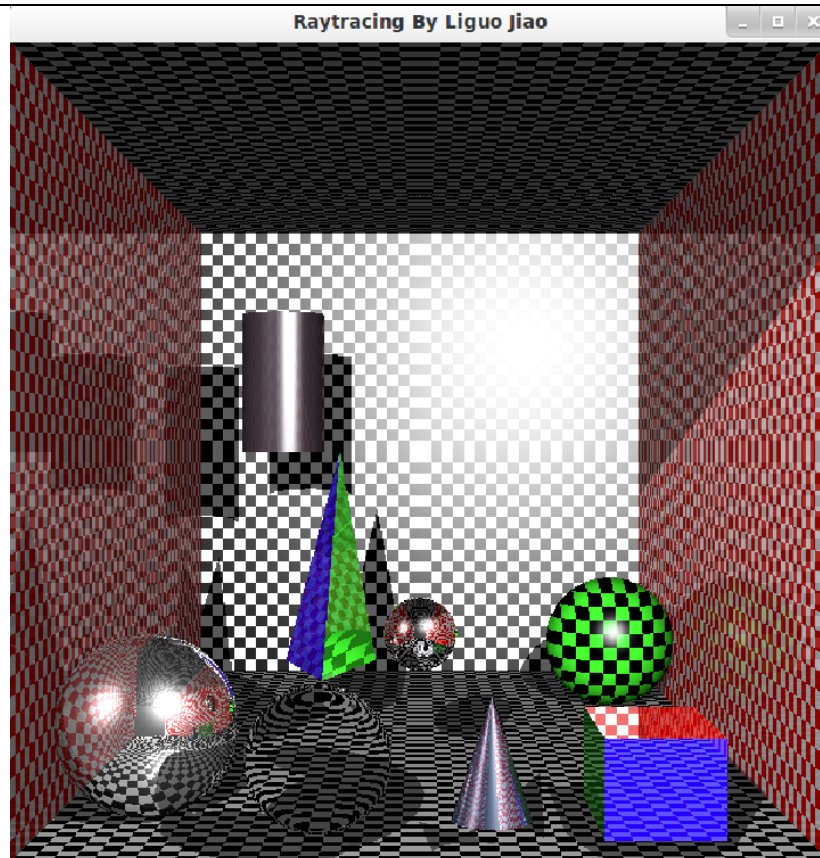
11. Textured sphere:



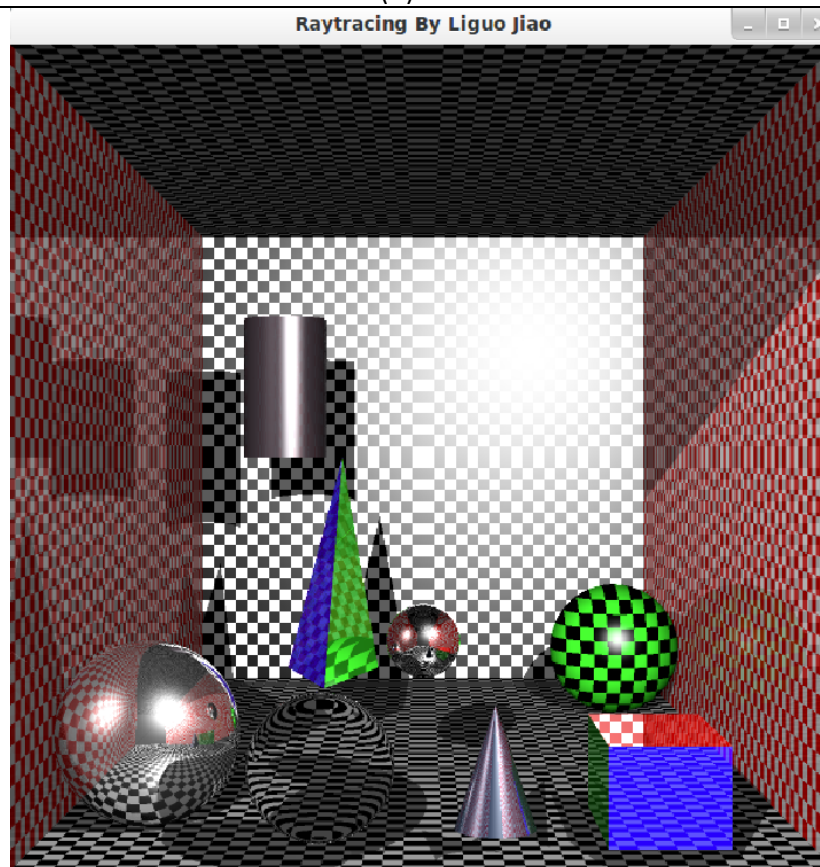
Textured Sphere

The way of generate the textured sphere is similar as textured plane which is sum of the object position x and negative position y if it's odd number then set up the colour is green, if it's even number then set up the colour is black.

12. Anti-aliasing(Super Sampling):



(A) Normal



(B) Super Sampling

The method of Super Sampling is to generate several rays through each square pixel and compute the average of the colour value.

Basically, the normal ray tracing is a ray into one pixel. In super sampling, the ray has to be split to 4 and split the one pixel to 4 small pixels, then match the 4 rays into the 4 small pixels and averaging the colour value. Finally, return the average. Since it is one pixel split into 4 small pixels, the operation time logically will take 4 times as long.

The differences between normal ray tracing (A) and Super Sampling (B) are showing on the left side.

According to the two samples we can conclude the super sampling could give the image a better shape and make the edge of each component smoother. The image noise of after super sampling is much less than the normal image.