# CSC418 A3 Report

998449497
Clark Chen

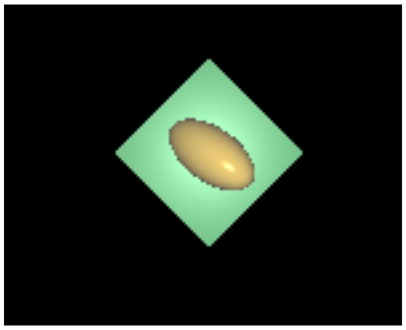
Raytracer, as last assignment of CSC418, offered plenty challenges and experiences. The first part of assignment is requiring us to implement a simple raytracer with basic functionalities. As for part2, people can choose to extend the raytracer or come up with a brand new graphic project.

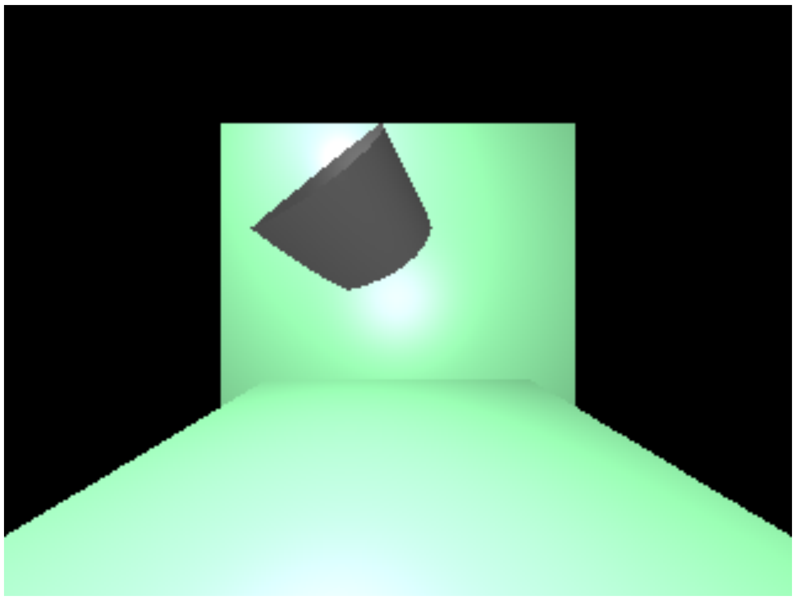
For part one, I finished all basic features, includes ray casting, ray-sphere intersection, ray-sphere intersection, Phong shading for a point light source and Anti-aliasing. For part two, I fully implemented ray-cylinder intersection, reflection and refraction and simple shadow effects, but failed on motion blur and texture mapping.

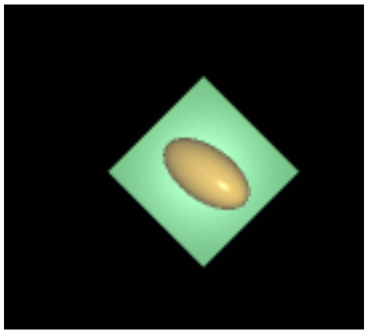
Regards to part one, I first implemented ray casting among all pixel in order to spawn correct ray for each pixel with recursively calling *shaderay()* to decide the final color. Meanwhile, I also implemented Anti-aliasing with Super Sampling, which sub divided each pixel into 2*2 grid, and casting rays at each individual grid. Then average out the amount of light and then decide the final color.

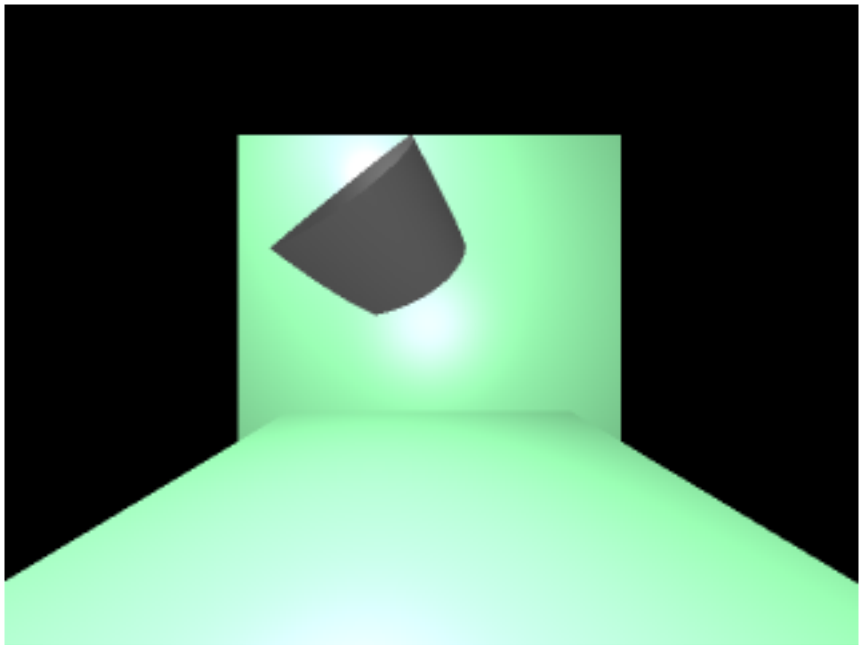
Group 1: picture without anti-aliasing

Group2:picuter with aa anti-aliasing







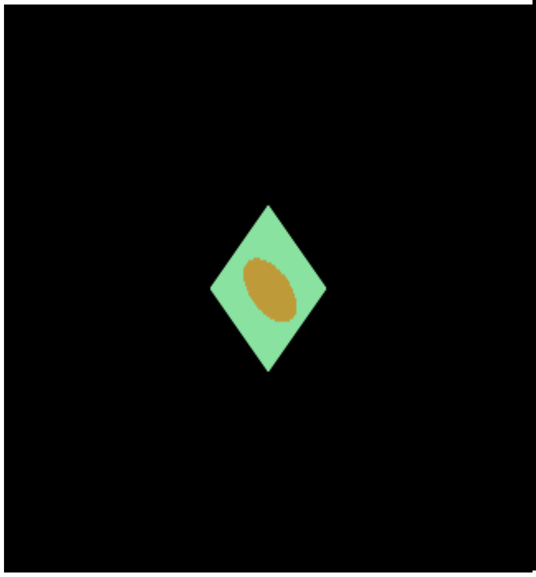
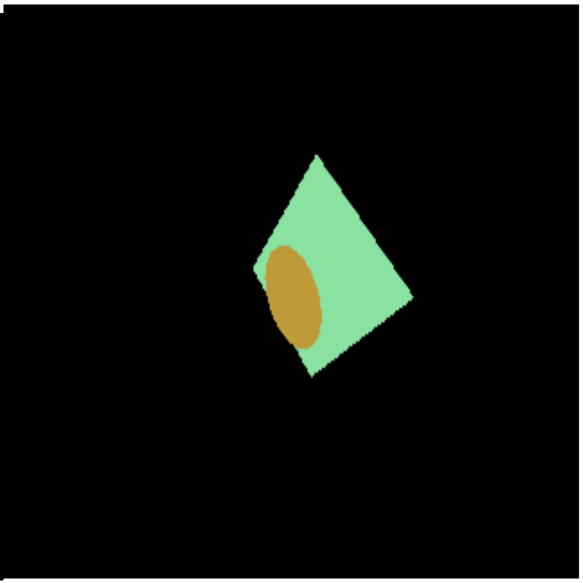


From above results, we can tell anti-aliasing is applied on second group of pictures

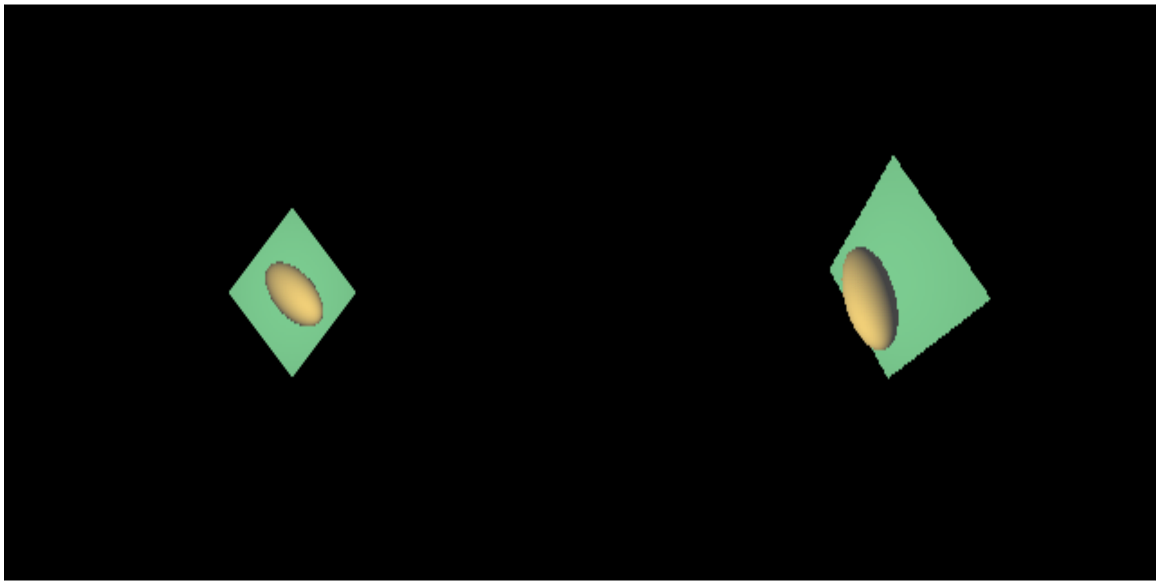
After basic ray casting, I implemented ray-unit square intersection, ray-unit sphere intersection in the scene object file and ray-cylinder intersection function. I used implicit equations to decide ray-object intersections among those functions, and transformed to appropriate coordinates by using *worldtomodel* and *modeltoworld* matrix conversions. With several condition checks in order to prevent over rendering (e.g ray.intersection.none=false), all objects are rendered correctly.

Next part is light source, I need to implement for three major rendering types. So I build up a *struct* with three rendering options, and calculate and assign to appreciate rendering option before calling render function.  Here are some results.

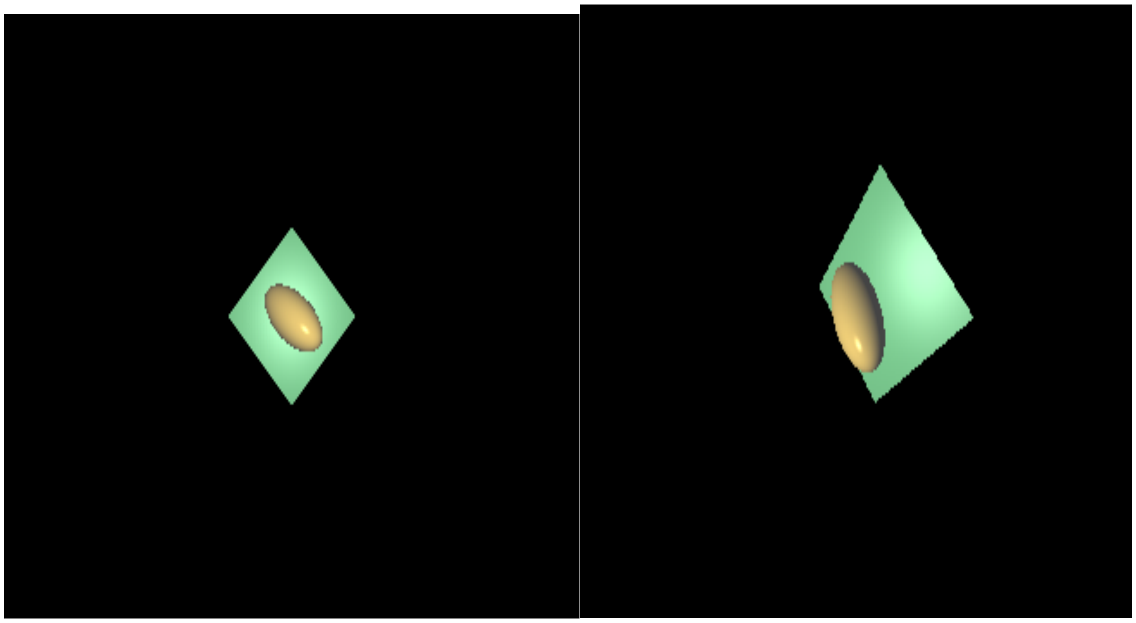
Group1 Signature

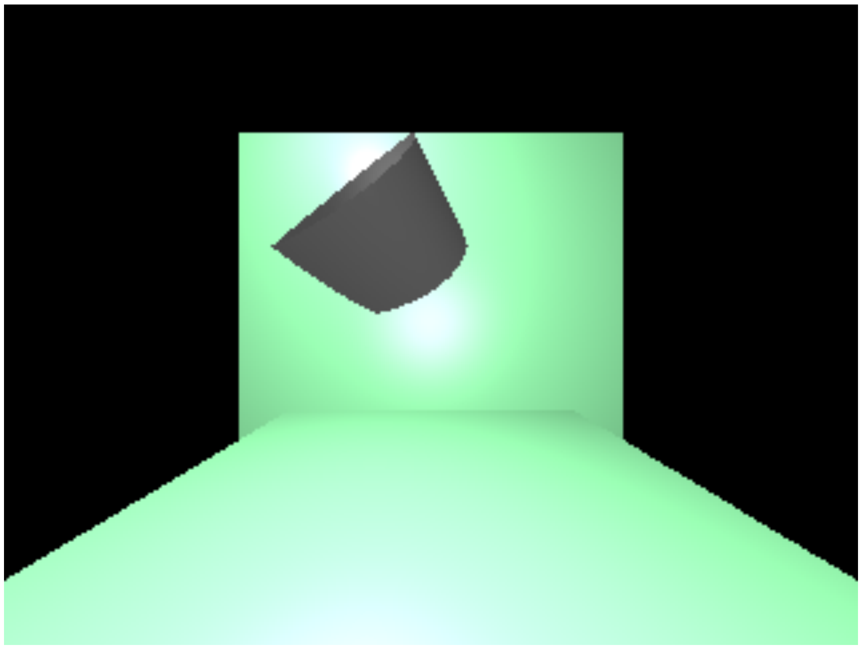
Group2 Diffuse



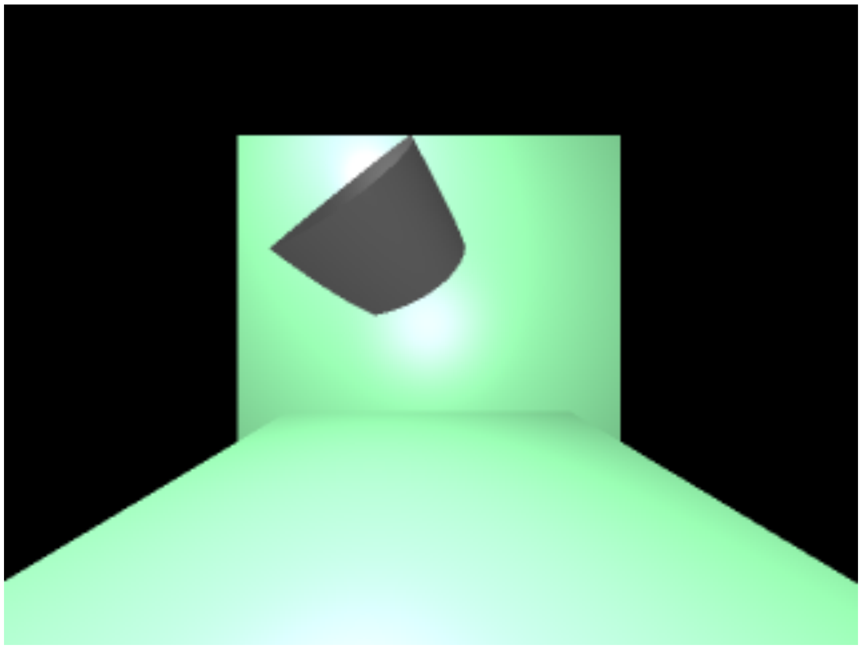
Group3 Phong

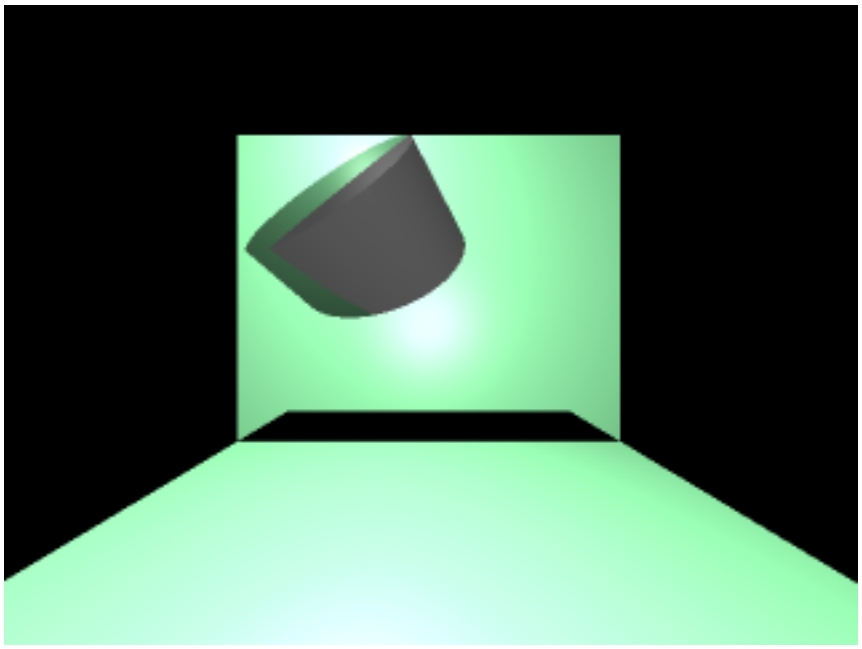


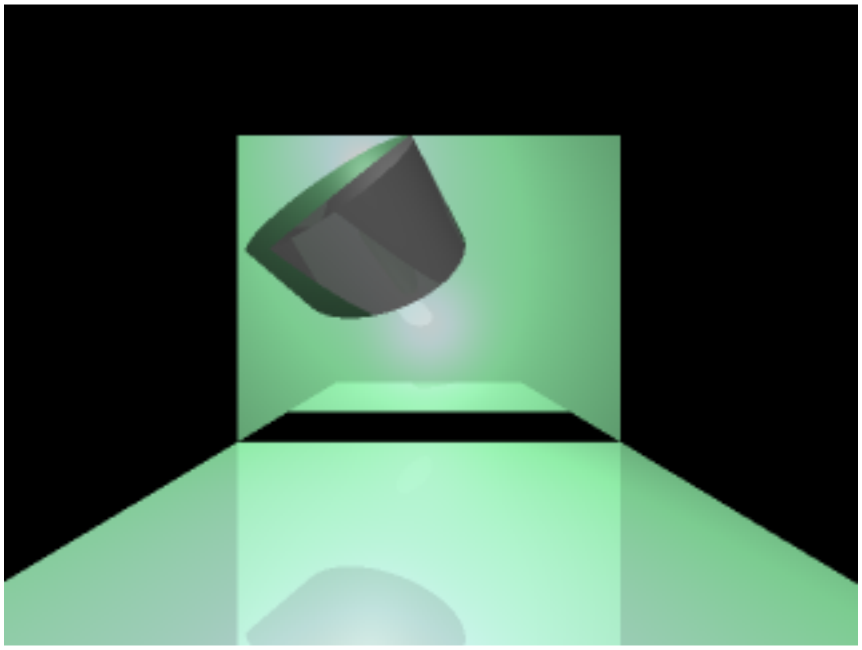
All basic features are implemented till now, so I start implementing advanced features. The first advanced feature I implemented is ray-cylinder intersection, which was finished with other shapes together in Part1. The second advanced feature I implemented is simple shadow effect from point light. Since I was going to implement area lights with soft shadow, I thought simple shadow effect could be good for comparison. Then I implemented reflection and refraction, which are hardest parts so far. For both reflection and refraction, I set up a max number Maxref to limt the amount of new rays that introduced among reflection and refraction (e.g: 5). Then I add new variables to materials, which includes Transmittivity and speed of light with refraction, reflectivity with reflection. Here are a series of results indicted by different setting ups.



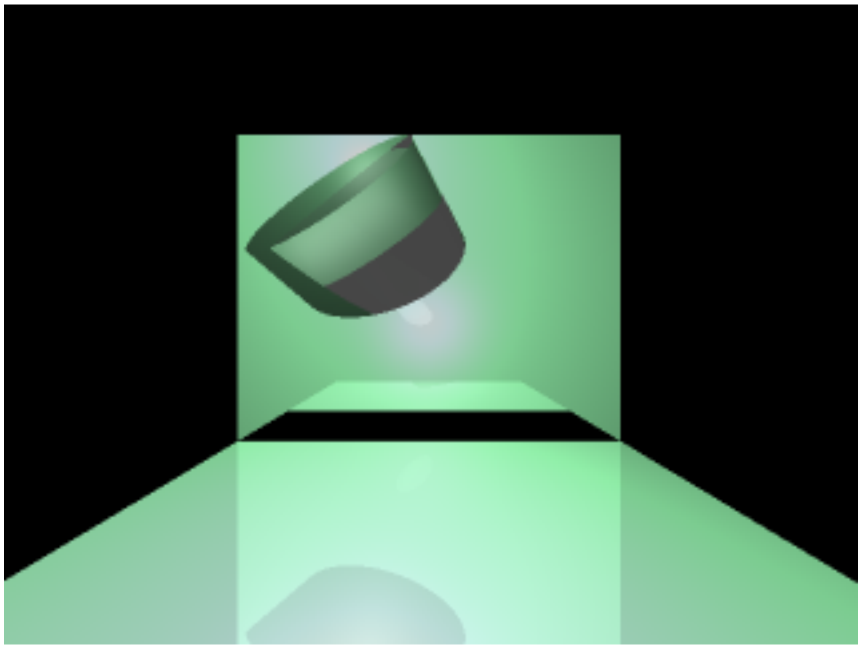
No feature


Anti-aliasing


Anti+Shadow



Anti+shadow+reflection



Anti+shadow+reflection+refration

I also tried to implement motion blur by randomizing object center respect their movement speed and time. Although the algorithm is applicable, but I found it's really hard to get correct result with mixture of other rays. I also tried to implement texture mapping by using triangular mapping. Again the result is not as good as I thought.

To conclude, I am slightly disappointed on my performance on this assignment. However this assignment definitely offered a lot experience for me, and I will try to do better next time.