COSC363: Assignment 2

Report

Max Bastida
89476639

Description.

The ray tracer can display geometric shapes with lighting and visual effects. The defined shapes are spheres, planes, pyramids and cylinders, and the effects are transparency, refraction and reflection. These effects can be applied to the shapes in different combinations, shown in Fig 1 and Fig 2.

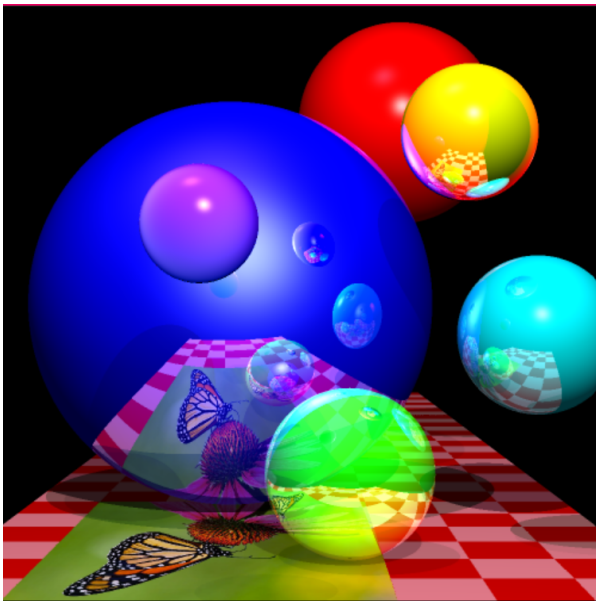

Fig 1: Rendering of combined effects

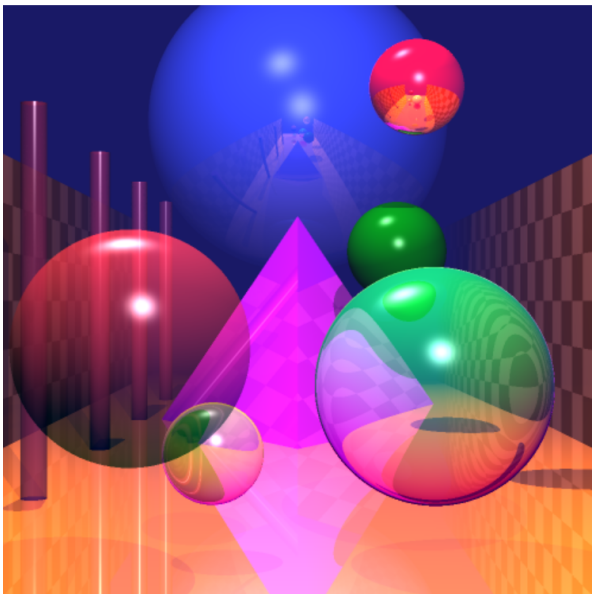

Fig 2: Rendering of multiple different shapes

Shortcomings

A shortcoming of this ray tracer is that shadows cast by transparent and refractive objects are not coloured to match the colour of the object casting the shadow. Another shortcoming is that although there is a class that can be used to simplify the creation of pyramids, each pyramid is rendered as four separate planes rather than as an individual SceneObject, and as such cannot be rendered with refractive effects, because planes do not refract. The ray tracer also takes a long time to run. The image shown in Fig 2 took 3 minutes to render when run in Visual Studio on a laptop.

Extra Features: Cylinders.

Cylinders can be rendered in the ray tracer (shown in Fig 2) using a Cylinder class that was created and inherits the methods from SceneObjects. The intersection point of a cylinder is found by solving for t in the formula described in the lectures. This was done by using the quadratic formula such that:

$a = dx^2 + dz^2$, $b = dx(x0 - xc) + dz(z0 - zc)$, and $c = (x0 - xc)^2 + (z0 - zc)^2 - r^2$

This was then used to find the two roots of the equation t1 and t2, and therefore the distance to the closest point. These points were then checked against the height of the cylinder. In cases where the closest root was outside the bounds of the cylinder, but the further root was

within, the cap of the cylinder was rendered. This was calculated using the formula given in the lectures.

Extra Features: Refraction.
The ray tracer is able to render refractive objects. This was implemented by tracing secondary rays through an object and modifying the direction of the ray based on the refractive index of that object. First a ray, g, is traced through the object from the point where the primary ray hit the object. Then another ray, h, is traced from where g hits the object to the rest of the scene. The directions of g and h are found using the glm::refract function and the eta calculated by the object's refraction index.

Extra Features: Multiple Light Sources
The ray tracer renders scenes with two light sources. To calculate the lighting of an object, first the lighting provided by each light source is calculated without the ambient light. This is done with the use of the SceneObject lighting method (modified to remove ambient light) and then calculating any shadow cast by that light source. Then the two light values are added together with the ambient light to produce the combined lighting effect.

Extra Features: Anti-aliasing
Anti-aliasing is used to make the image appear smoother. In places where there are distinct changes in colour, most notably on the edges of objects, because the scene is rendered in pixels, there will often be harsh, pixelated edges in the rendered image (Fig 3). Anti-aliasing is achieved by sampling a number of points per pixel and colouring the pixel as an average of those points. This softens the transitions between colours and smooths the edges of an image (Fig 4).

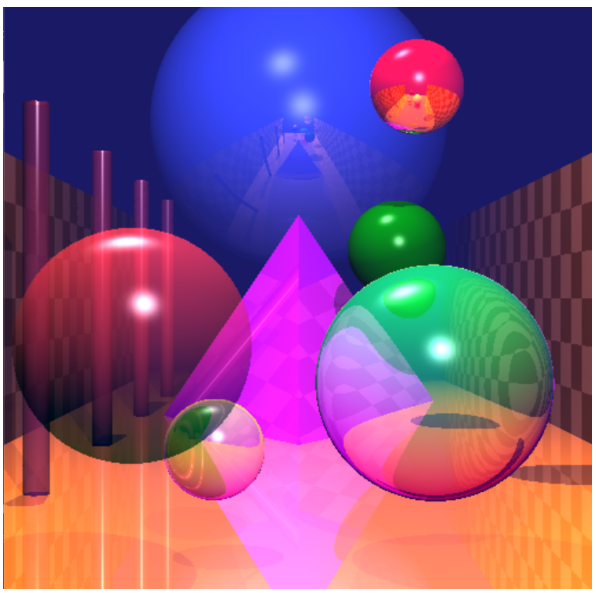

Fig 3: without anti-aliasing

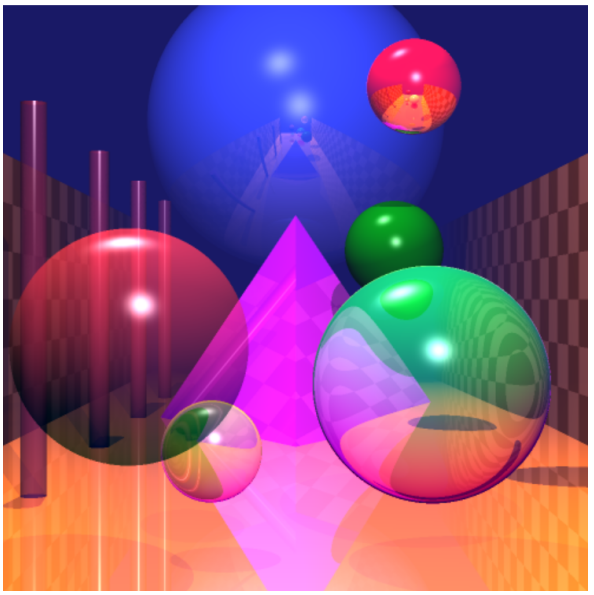

Fig 4: with anti-aliasing

Anti-aliasing was implemented in this ray tracer by dividing each pixel into four separate rays and colouring the pixel the average colour of those rays.

Extra Features: Fog

Fog was added to the ray tracer to make objects that were placed further away in the scene fade into the background. Fog was implemented by blending the colour of a ray with the background colour of the scene based on the z distance of the point the ray hit. The fog was defined by two points: fogz1, before which there was no fog, and fogz2, after which was entirely obscured. Between these two points the fog colour was blended with the ray colour to a ratio proportional to the distance ray.hit.z was into the fog. The effect of the fog is shown in Fig 5 and 6.

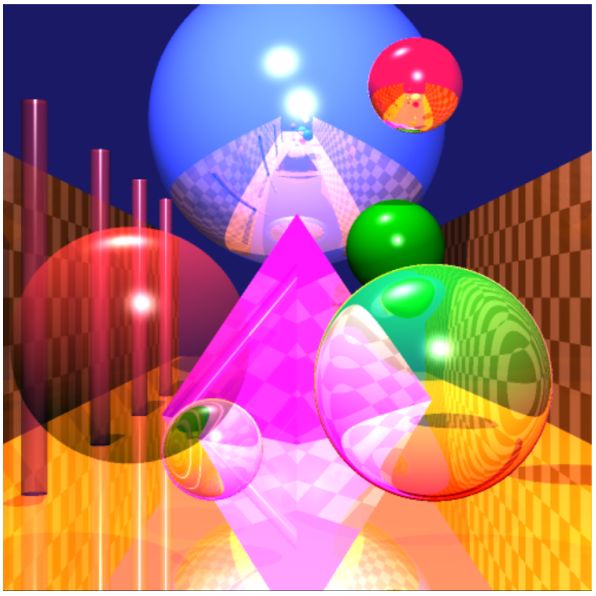

Fig 5: without fog

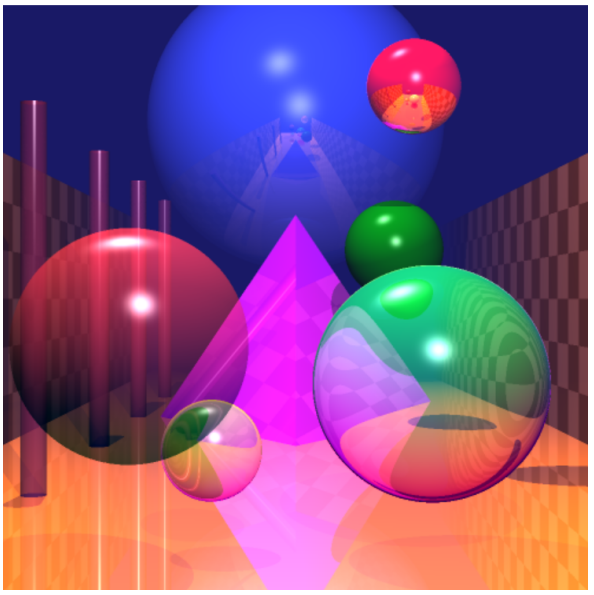

Fig 6: with fog

Instructions

To compile and render the image shown in Fig 6 with CLion:
Open the program folder in CLion.
When the folder is opened, it should detect the CMakeLists file and load the project. If this doesn't happen, right click on the CMakeLists file and select Load CMake Project.
Go to the Run menu and select Edit Configurations.
Select the RayTracer.out configuration and set the working directory to be the program folder. Apply and close the configurations menu.
Select Run.