Raytracer Report

s2281597

Module 1

- Blender export script – tested with zero, one and multiple of each object to ensure all are exported correctly into the lists in JSON format. The values in the output JSON were compared with the values within Blender to ensure they were accurate.

- Camera data reading – initially implemented for one camera in the scene, then improved to read details from different cameras into objects using an indexing system.
    - Issues with multiple cameras found when calling the method for multiple objects would read in the same data.

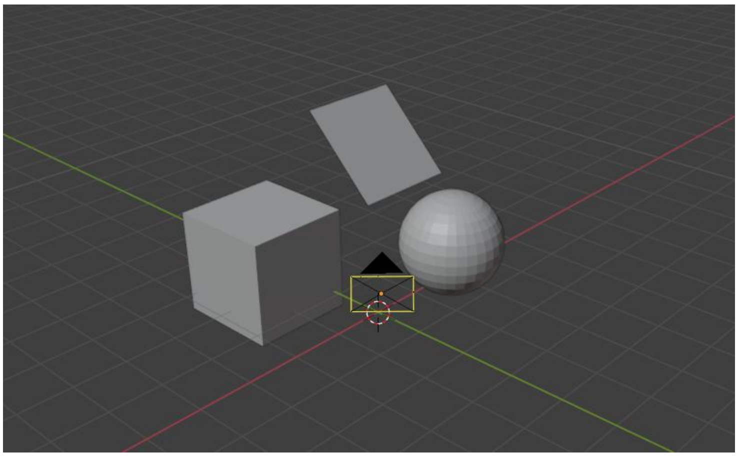
- Pixel-to-ray conversion – tested with a Blender script that creates a visualisation of the output ray, to ensure it originates from the camera and heads towards the correct point.

- Image read and write – tested with existing .ppm images, editing and saving them with different file names and checking that the pixel colours changed accurately.

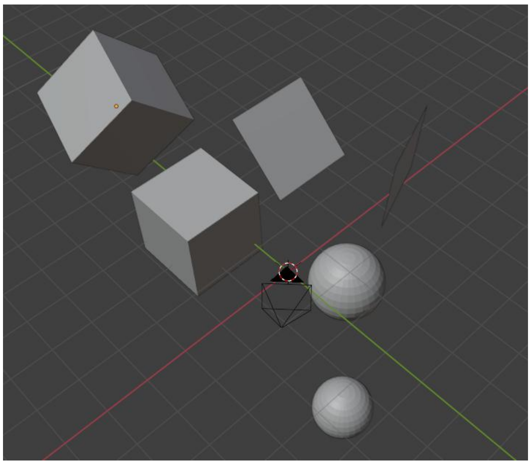
Module 2

- Ray intersection:
    - Intersection checker – tested by creating Blender scenes with each shape type, transforming them and checking if a ray intersected with it via my ray visualiser (ray_vis.py). This result was compared with the module Boolean output.
    - Shape hierarchy – "Shape" superclass was created and all existing shape classes were put as subclasses. Testing involved defining class methods and properties that were only declared in the superclass, overriding them with each shape's appropriate functionality
    - Hit structure – tested by checking the "hs" (hit structure) property of each ray after intersection, ensuring each has properly stored the intersection point and distance along the ray, and that these objects have been returned correctly.

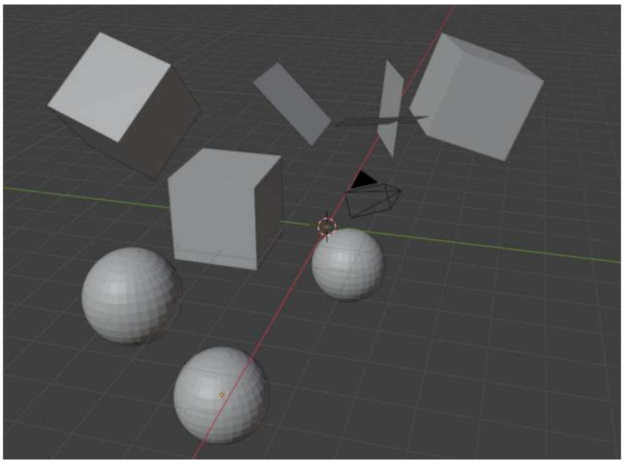
- Acceleration hierarchy – tested by checking if accelerated and unaccelerated methods achieved the same number of intersections and with the same types of objects for the same scene. Runtime was then compared for scenes with varying numbers of objects, as seen below:
  - Scene 1 – 3 objects (1 plane, 1 sphere, 1 cube):



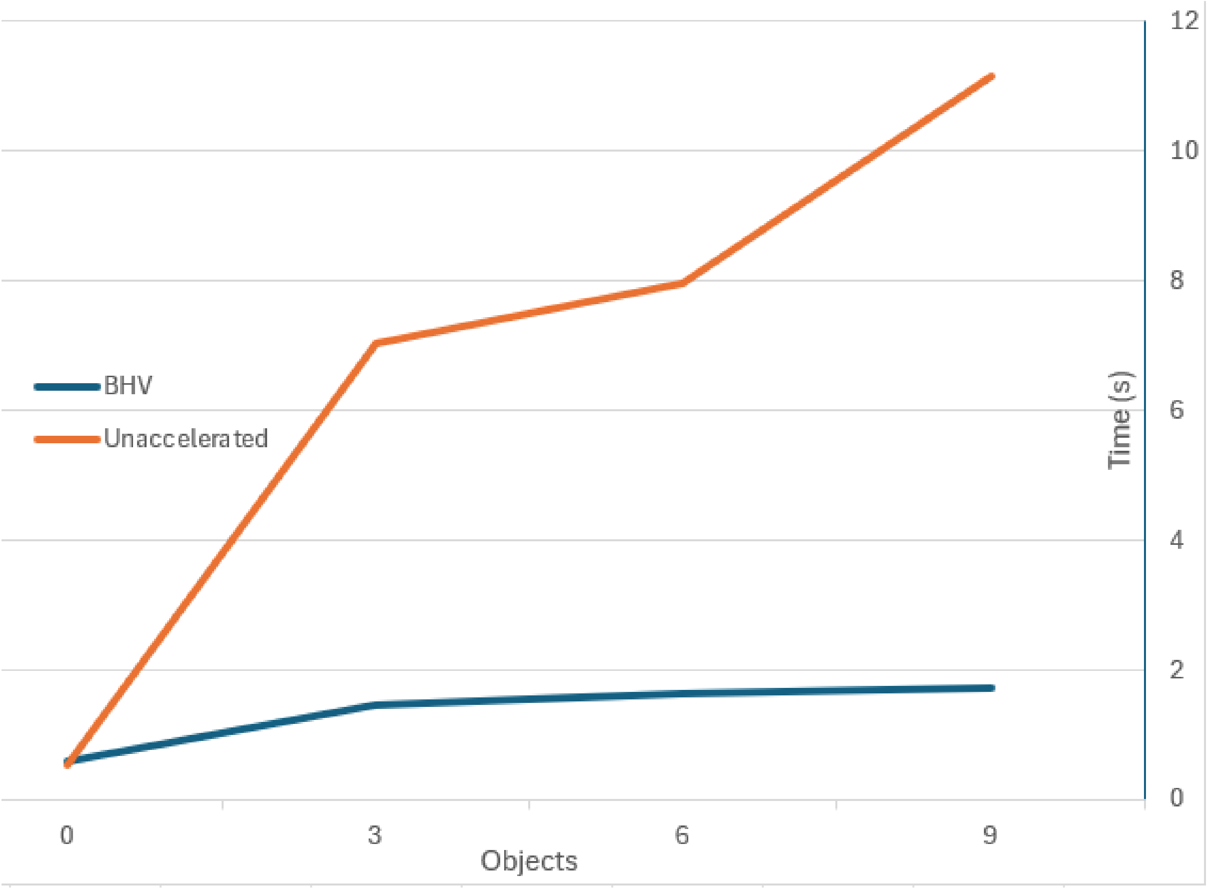
  - Scene 2 – 6 objects (2 planes, 2 spheres, 2 cubes):



  - Scene 3 – 9 objects (3 of each):



- Performance of unaccelerated interception checking vs BVH accelerated checking:



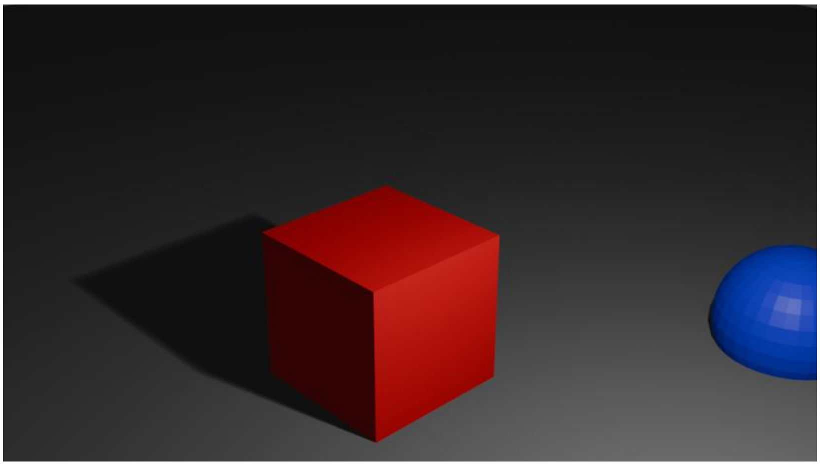
Module 3

- Whitted-style raytracing:
    - Added material property extraction to the Blender Export script
    - Implemented Blinn-Phong shading
    - Implemented recursive reflected and refracted rays that check for intersection – tested with various levels of reflection/refraction depth

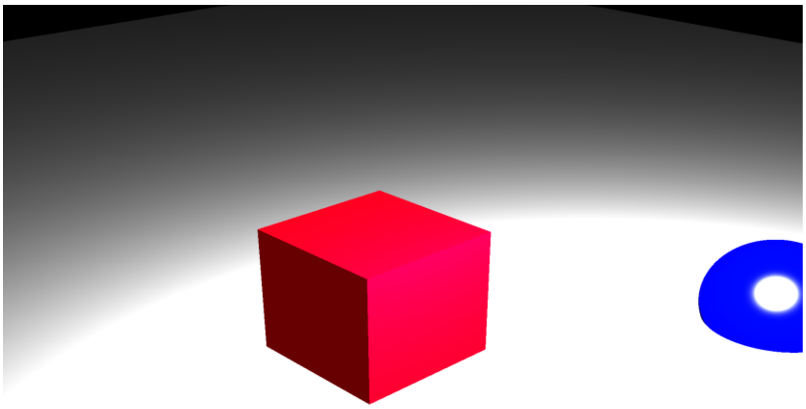
- Anti-aliasing – implemented with Monte Carlo sampling within each pixel, tested with increasing numbers of samples to investigate effect on jagged edges

- Texture mapping – tested with checkerboard ppm image mapped to each object type in Blender

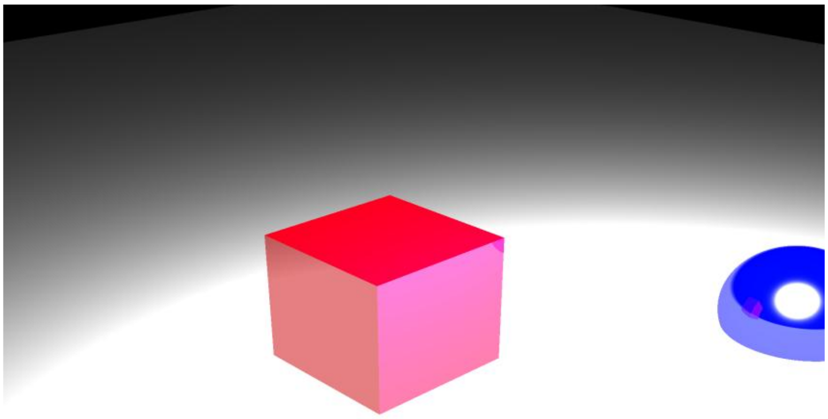
- Blender render of scene:



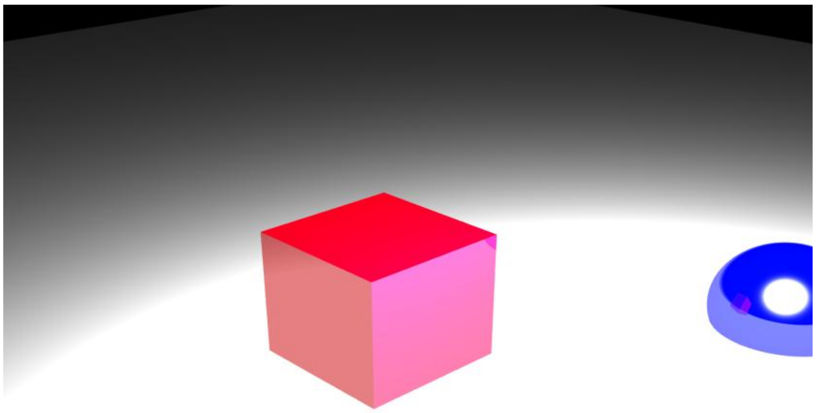
- Project render of scene – reflect/refract depth = 0, no anti-aliasing:



- Project render of scene – reflect/refract depth = 5, no anti-aliasing:



- Project render of scene – reflect/refract depth = 5, anti-aliasing with 4 samples/pixel:



- Project render of scene – reflect/refract depth = 5, anti-aliasing with 4 samples/pixel, checkerboard texture mapped to cube: