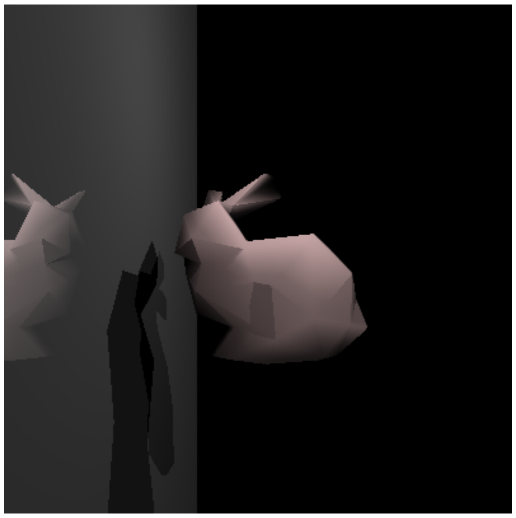
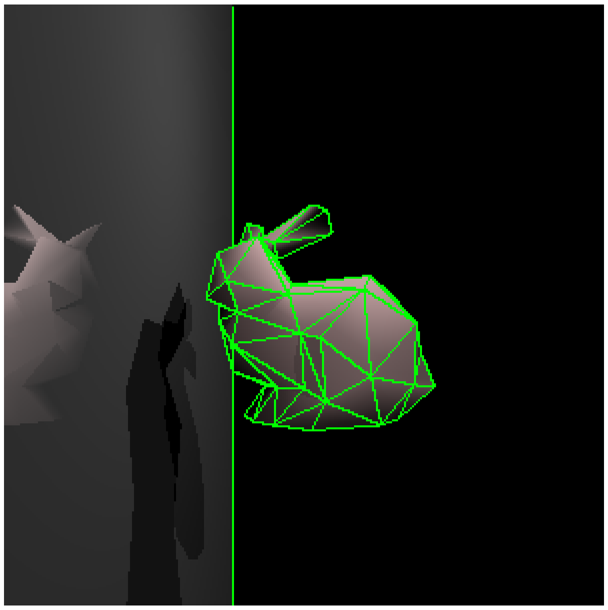
1. Briefly describe your progress. Address all items that were in your project proposal for this milestone.
2. Include some form of visual evidence of the results you've achieved so far: this could be output images, result videos, or screenshots of your progress; if your project is WebGL-based and opening your HTML file shows your current results, that will suffice, but make sure it's clear which file I should look at.
3. Provide instructions for running your code, and a description of what I should expect to see when I do so.
4. If the project's goals need to change at all (e.g., to adjust scope to account for unforeseen challenges, or to further clarify goals), provide an updated set of goals for your final deliverable. Explain each change with respect to your original goals.

Turn-Ins

Anti-aliasing (Jackson), edge-detection (Josh), refraction (Shahu)

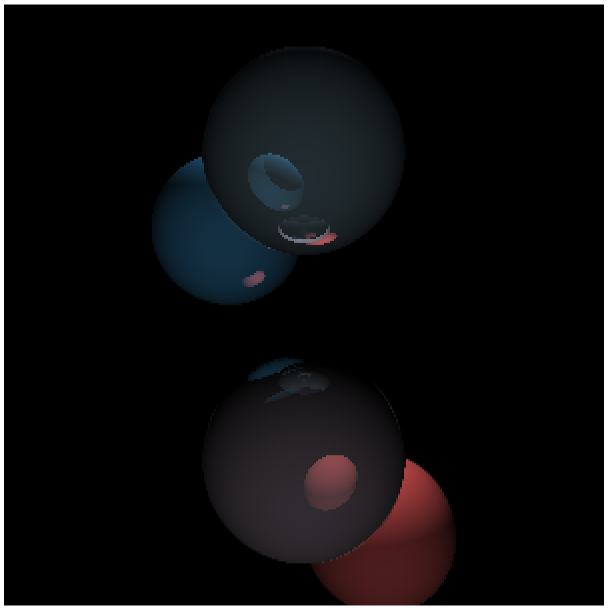


The anti-aliasing system supports uniform, random, and stratified supersampling. The main function now takes two additional arguments: the anti-aliasing mode and the number of samples. Anti-aliasing modes are as follows: 0 - no AA, 1 - uniform AA, 2 - random AA, 3 - stratified AA. The anti-aliasing system iterates over the edge mask, resampling the scene at a given index if that index is marked as an edge and updating the pixel color. The number of samples is equal to the square of the sample number argument. Performing anti-aliasing only at the edges of objects allows for very high sample rates with relatively little performance decrease.



For **Milestone 2**, the introduction of a locality parameter in the edge detection will allow for more noticeable anti-aliasing effects on objects, as the AA only offers small improvements with the current 1-pixel-wide edges.

The edge detection algorithm maintains O(n) efficiency to produce a boolean mask which is fed into the anti-aliasing algorithm. The anti-aliasing will only occur at the true indices of the mask. Left, is my current edge detection algorithm output. I will feed the boolean mask this method produces to the anti-aliasing algorithm to provide a smoother picture at rough edges. For **Milestone 2** I will improve the edge detection algorithm to include reflections and shadow edges. I will also create a locality parameter to grab pixels that are L away. These efforts all work to intelligently select pixels to perform anti-aliasing on.



The refraction calculation system works by first checking the angle at which a light ray hits the surface of the object containing the refractive material. If the light ray hits the surface at a "shallow" angle, the system uses *Snell's law** to calculate the direction of the refracted ray. Otherwise, if the light ray hits the surface at a "steep" angle, the ray undergoes a process known as *Total Internal Reflection* (TIR), in which the ray is completely reflected (away from) the object instead of refracted (passing through) the object. **Snell's law* generally calculates how light rays will bend as they pass from one object to another object with a different refractive index. The system then determines the final color of the pixel by combining the color of the refracted light ray with the base color and transparency of the object the ray hit. The outcome of this algorithm is seen on the left. For **Milestone 2**, I will enhance the refraction calculation system by implementing *Bounding Volume Hierarchies (BVH)*, which would reduce the number of times the system has to run to calculate all refractive surfaces, thereby increasing performance as scenes become more complex.

Milestone 2 Goals

- Edge Detection for shadows and reflections.
- Oren Nayer shader model
- Rectangular area light using the same 3 sampling methods. Use the **Strategy** design pattern to call the 3 sampling methods implemented with an interface.
- Bounding Volume Hierarchies