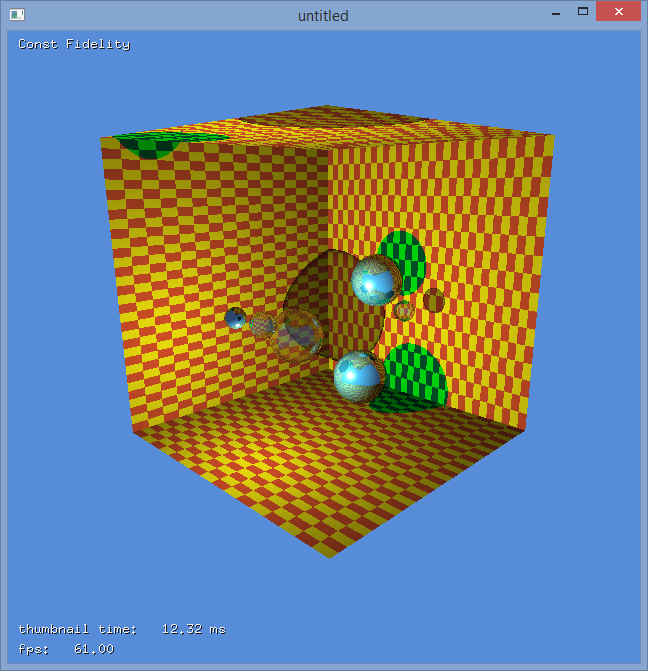
**Interruptible Rendering Project**

Structure overview:

All meaningful files are split between the Scene and Renderer; Scene manages the objects and surfaces seen in the scene itself, the buffers that OptiX renders to, and the ray generation programs used to render the scene. Renderer manages the main render loop of the Scene and generally handles all the implementation details unique to this project (constant fidelity vs interruptible rendering modes, writing pertinent data to file, handling keyboard input to manage these states, etc).

Scene overview:

In the scene, three red and yellow checkered walls serve as shadow receivers for different spherical objects in the scene. There are three reflective balls and three glass balls; the largest glass ball in the center rotates about a point near the center of the scene, changing the caustic shadow casted on the wall behind it. The animation loop of the large glass ball spins fairly rapidly for a small number of revolutions (about 2.5) and then pauses for a short time before resuming its spinning animation from the previously stopped position. There is a superficial bug at the moment in which after about 30 seconds the large glass ball stops animating entirely, and the program must be restarted to resume its animation. A zoomed out, slightly rotated screen capture of the scene is shown in constant fidelity mode.



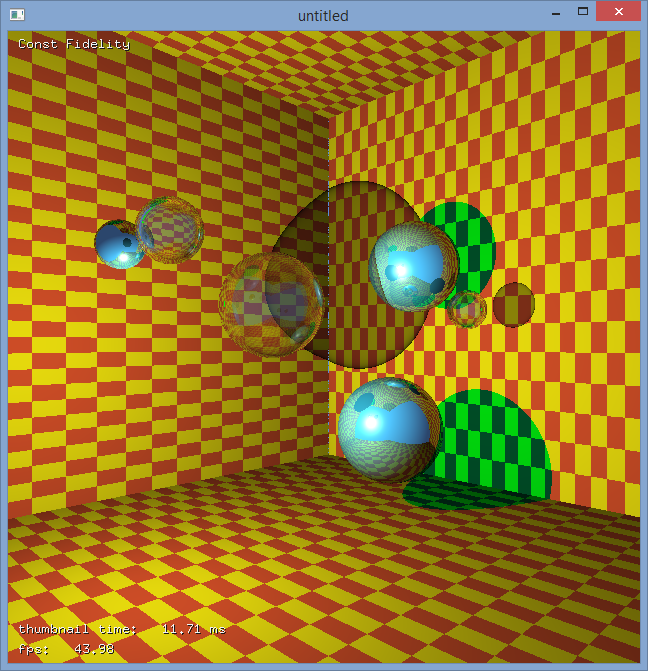
[Figure 1: A zoomed out version of the scene, rendered in constant fidelity]

Inputs to the program:

* Holding left click and dragging the mouse will rotate the camera view to provide different views of the scene from the same point of rotation. Holding right click and dragging the mouse left and right will translate the camera forward or backward in the scene. The combination of these mouse inputs lets the user move the camera to any location in the scene with any desired viewing direction.
* Pressing the ‘t’ key switches the render modes between the constant fidelity and interruptible rendering modes.
* Pressing the ‘a’ key will toggle the animation of the large glass ball in the center of the screen off and on. As mentioned previously, there is currently a superficial bug that limits the total amount of time the object will animate in scene before becoming unresponsive, although other inputs to the system (mouse, etc) still function properly.
* Pressing the ‘g’ key enables the rendering of the golden standard images against either the constant fidelity or interruptible rendering modes, depending on which is currently active. This writes .ppm image files to disk, assumedly for future comparison given the inputs provided to the program (reflected in the scene’s current state).
* Through code inspection, it appears that pressing the spacebar key is intended to destroy and repopulate the spheres in scene, although currently this crashes the program.

State of the rendering modes:

As seen in the image above, the constant fidelity render on my home PC renders at about 60 FPS in the zoomed out view, and roughly 40 FPS in the default view the program starts in. This is shown in the image below. In this mode, the large sphere ball animates in a tight orbit at a fairly rapid speed with acceptable frame rates.



[Figure 2: The default view of the scene, rendered in constant fidelity]

In the interruptible rendering mode, the below image is produced at roughly the same time period as the screen capture taken of the constant fidelity capture shown above. The spatial and temporal error values used in the render calculation are displayed on screen, and the displayed FPS hovers above and below 60 FPS. However, the large glass ball animates at a much slower rate in the scene, which is due to the animation rates being tied to the render rates instead of a global sense of elapsed time. As is clearly visible in the screenshot, the image quality of the interruptible rendering mode is drastically reduced compared to the constant fidelity mode. One point worth noting is that unlike in the videos Dr. Watson shared of what I assume are previous incarnations of the project, when the ball pauses in its animation loop the image quality does not improve.



[Figure 3: The default view of the scene, rendered in interruptible rendering mode]

Distance based error measures:

As best as I can tell, if any distance based error measures were used before, they have since been removed