Asteroid Blizzard

Eden Zik and Kahlil Oppenheimer

We created an asteroid dodging game, with a reverse premise. Instead of the goal being to dodge asteroid, the player's objective in "Asteroid Blizzard" is to crash into them in full force. The more crashes a player achieves in a 3D world, the more points they earn. This game uses the standard WASD navigation keys, along with the space bar to generate thrust and "b" to stop.

The description is as follows:
"Asteroid Blizzard is a space game unlike any other... you are a crazy pilot trying to crash into as many asteroids as possible with your super strong ship. You change direction with either a, w, s, d or the arrow keys. You accelerate with space bar and decelerate with b. The more Asteroids you crash into, the more points you get!"

In this game, we focused most of the efforts on creating a visually elegant world that is focused around the player. The player is an abstract shape that can turn in any direction, and apply thrust. The world consists of a sphere which moves along the player, and clusters of stars (hence "blizzard") which move about the player.

There are several components to this assignment, all of which required some engineering.

1. Initial concept - Eden
    a. The concept of this assignment initially involved a plane on which all of the action took place. Inspired by resposities seen online, such as this, we decided to model the world totally in 3D.
    b. The idea of this is more around the concept of an infinite 3D world, which is a theme we found interesting.
2. World layout - Eden
    a. The layout of this world is spherical, with the player in the center.
    b. The trick of the model was to have a sphere with an internal texture, as seen in the example link, and have the entire sphere move along with the player. This gives the illusion of an infinite space, which can be explored endlessly.
3. Avatar - Kahlil/Eden
    a. The avatar is a simple and elegant pyramid like mesh. It is designed to allow the player to fly while still integrating well with the physics engine in the game.
    b. The avatar is able to move around, and its movement lead to rotation of the camera appropriately. It is able to exert exert thrust as well as pause, all using the physics engine described below.
4. Key Movements - Eden/Kahlil
    a. We used the keyboard binding "switch statement" demonstrated in class to create unique bindings for each key, enabling us to move around the scene.

    b. We then added the ability to combine keys to create thrust while changing direction, for example.
5. Asteroids - Kahlil
    a. Asteroids were simple meshes in random colors, with an initial velocity in the space
    b. The Asteroids are also able to collide with the player, as described.
6. Physics - Kahlil
    a. Rather than implementing our own naive physics mechanisms, we used an off the shelf library for modeling collisions called PhysiJS.
    b. We were able to integrate all objects in the scene to be able to collide with each other, and apply a force that will move the player's avatar in the opposite direction.
7. Sound - Kahlil
    a. We added funny music in the background
8. Score keeping - Eden
    a. Upon a collision, score is added to the player indicated in the top right.
9. Star Blizzard - Eden / Kahlil
    a. We used oncoming points to model stars, which auto generate as the player moves around the scene.
    b. Each star is a white rectangle, given an initial velocity. It is called a star forge, and is based on a repository we found.
10. Landing page - Eden
    a. A landing page to showcase our work
    b. The landing page contains a link to our repository as well.