Jeffery Bouchard

Alex Ward

CS 480

Project 9 Air Hockey

**Overview:**

This is a basic two player Air Hockey game using C++, Open GL and GLUT. Use the w, a, s, d keys and mouse to move the paddles in order to hit the puck in the opposing side’s goal. First player to score 7 points wins the game.

* *Extra Credit*:
* Text2D display
* Game replay

**User Manual:**

* **Controls**:

Menu: right mouse button

* Play – Starts the game.
* Pause – Pauses the game.
* Reset – Resets the board.
* Reset Game – Resets scores and the board.
* End Game – Exits the program.

**Player 1:**

Move Right: move **mouse** to right

Move Left: move **mouse** left

Move Up: move **mouse** up

Move Down: move **mouse** down

**Player 2:**

Move Right: press ***d*** key

Move Left: press***a*** key

Move Up: press ***w*** key

Move Down: press ***s*** key

**Camera Controls:**

Free Roam Camera: Press **F** to set the camera to be movable.

Follow Player One: Press **1** to set the camera to follow player one.

Follow Player Two: Press **2** to set the camera to follow player two.

Move Camera:

**Arrow keys** to move the camera in the x and z direction.

**Period** and **Forward Slash** to move the camera in the y direction.

Reset Camera: Press **F** again to reset the camera.

**Instruction Manual/Walkthrough:**

* 1. Begin the game by pressing the right mouse button and selecting *play* as shown by Fig. 1 below.

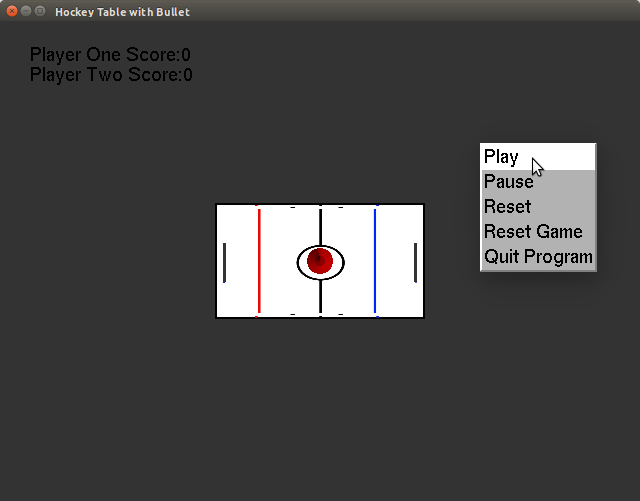


Figure 1: Starting the game

* 1. Once the game has begun, press the right mouse button and select *pause* to pause game play, as shown by Fig. 2.

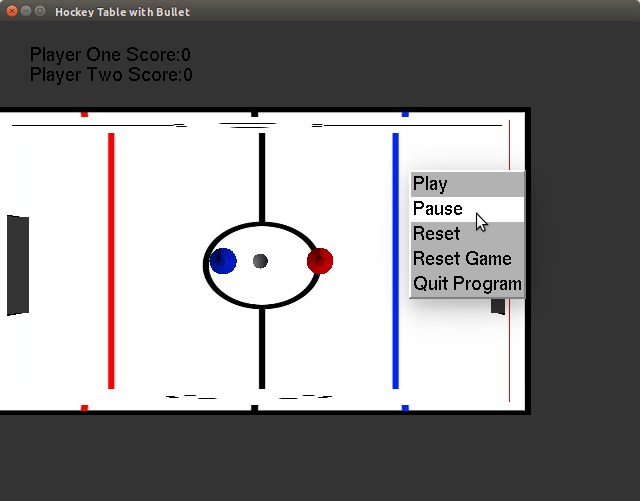
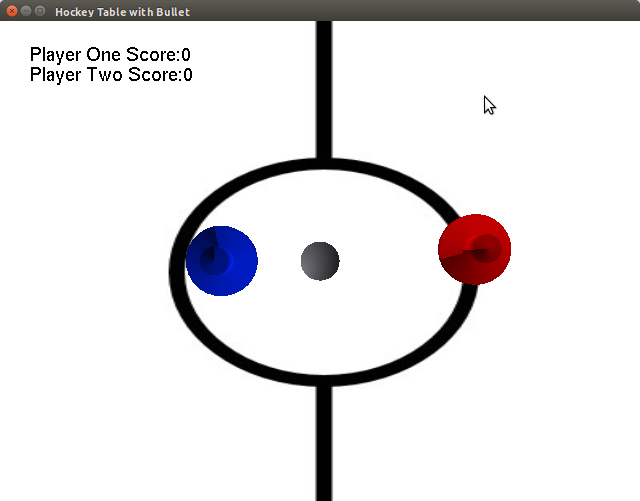


Figure 2: Pausing the game



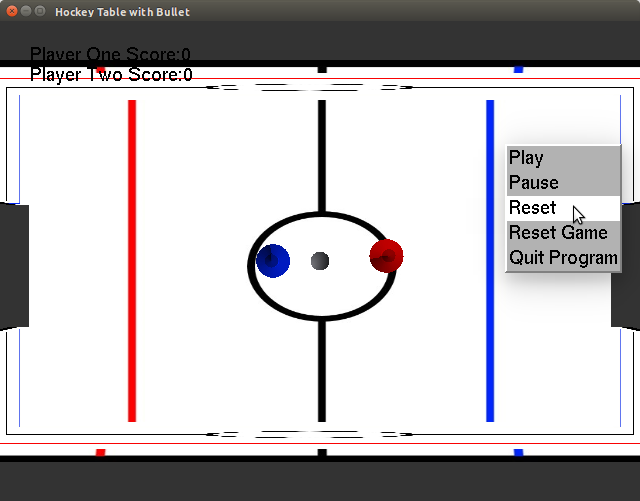
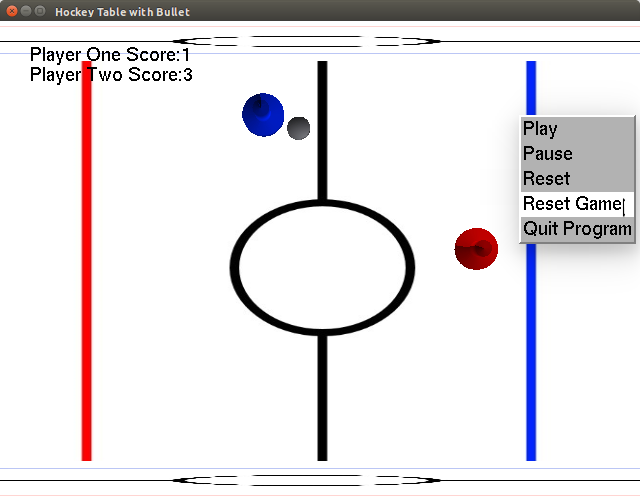
* 1. Player 1 (in red) must use the mouse to control the puck movement.
  2. Player 2 (in blue) must use the wasd keys to control the puck movement.

Figure 3: Resetting the puck and paddle position.

* 1. If at any point during game play the puck becomes stuck, the game can be easily reset. This is done by pressing the right mouse button and pressing the reset button, as shown by Fig. 3 below.
  2. After either player scores, the current score will be printed to the screen and the puck and paddles will be reset to the starting position.

Figure 4: An example of the score and the reset game menu option.

* 1. After a player scores 7 goals the game is over. The score will be reset to zero and the puck and paddles will reset to their starting position.

**Technical Manual:**

* **Compiling**:

1. Code must be compiled on a linux machine using the gcc compiler, ensure Magic++, Bullet, GLUT, and OpenGl libraries are all available.
2. Navigate to the build folder and type *make* to compile the code into an executable file.
3. Type *make clean* to override the existing executable and recompile completely.
4. Navigate to the bin folder and type *./table.exe* to run the compiled code and play the game.

* **Issues**:

1. Fluidity of the puck, and the paddles.
   1. The paddles and the puck are effected by gravity, but it is not as fluid as it could be. The puck and the paddles really don't bounce off the walls as well as we would have liked. The paddles and the puck also flip over rather often. We increased the mass of the objects and this made a slight difference, but as of now we don't have the best solution.
2. Texturing of the objects.
   1. The objects were textured with unique images. The board has a field that resembles an air hockey table. The issue is that it doesn't look as professional as it could. This is in part due to the lack of experience with blender and texture rendering. We are still happy with the way the objects turned out.
3. Collisions of the objects.
   1. The collisions are working, but on occasion the puck will fall through a triangle or a corner of the board. Not sure how to fix this. Rebuilt the board mesh several times and still same issue. It only happens when the puck falls weired on a corner or intersection of triangles.
4. The Bullet API.
   1. The bullet API was not helpful at all a majority of the time. It was easier to look at the code in a header file then to read the API.
5. Bullet walls disappearing.
   1. Don't know why, but on reset occasionally our boundary walls for the paddles will disappear. Not sure what is causing this issue.
6. World hunger
   1. A major issue.

* **What we would change**:
* The movement for the camera is not as fluid as we had hoped for.
* We would have added a menu, and a high scores if time allowed.
* The wasd keys controlling player 2 were also a bit of a pain as the system will react to one at a time.
* We also would have completed more extra credit.
* **Interesting Things**:

1. Saving a rigidbody pointer in our object class and creating a get rigidbody function helped a lot with manipulating and or reseting our objects.
2. All of our objects have rigid bodies created by outlining the mesh.
   1. The board is a btBvhTriangleMeshShape() which is used for static objects.
   2. The rest of our objects are btConvexTriangleMeshShape() which can be used for dynamic objects.
   3. If we didn't use the btConvexTriangleMeshShape() our objects would not collide even with the bit mask collisions set up.
3. We checked the position of our puck relative to our board as opposed to checking collisions with the puck and an invisible wall in the goal to keep track of score.