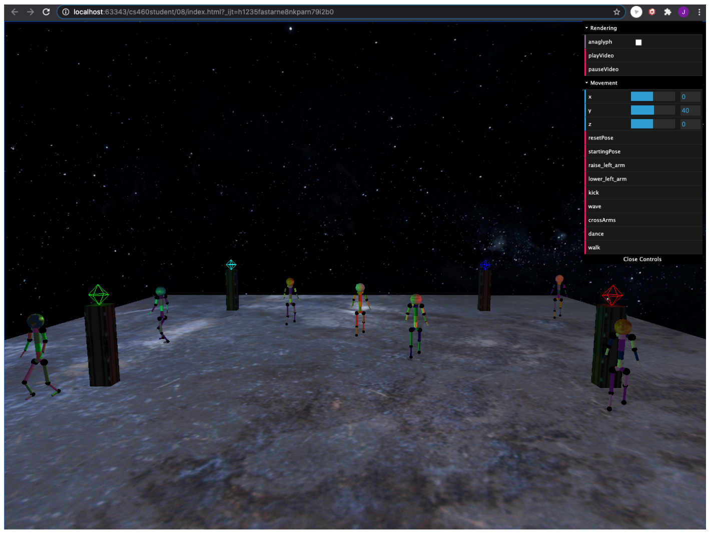
CS460 Fall 2020
Github Username: JamesEdMichaud
Due Date: 11/20/2020

Assignment 8: The Walking Robots!

We will create walking robots that switch directions when they bump into each other.



Starter code for assignment 8. After pulling from upstream, there is the folder `08` in your fork. Please copy `index.html` and `robot.js` from assignment 7 over or use Daniel's solution from `https://cs460.org/shortcuts/31`.

Part 1 (4 points): Please create a `Robot.prototype.walk` method that sets `this.movement` to "walk". Walking is very similar to kicking, which we implemented previously, but with less rotation. Also, please create a new `dat.GUI` button that calls the walk method for all existing robots in the scene.
robot.js, line 768 and index.html lines 98, 146

Part 2 (10 points): Please change `Robot.prototype.onAnimate` to catch if `this.movement == "walk"` and add functionality to **slerp the left upper leg by 45 degrees along the X-axis**.
robot.js, lines 786-801 and 565-662. These line numbers apply to this part and the next 3 parts.
My robot does quite a bit more than a single rotation... I implemented 4 steps to the walking motion and used the pelvis bone quaternion to trigger step progression.

Part 3 (10 points): Please change `Robot.prototype.onAnimate` to catch if `this.movement == "walk2"` and add functionality to **slerp the right upper leg by 45 degrees along the X-axis**.
See part 2.

Part 4 (10 points): In both cases, add functionality that slerps the other leg back to identity. So on "walk", slerp the right upper leg to identity, and on "walk2", slerp the left upper leg to identity.
See part 2.

Part 5 (10 points): Now we have two components. Please combine them as follows: in the "walk" block, check for `this.right_upperleg.quaternion.w` - if the value is smaller then 0.93 set this.movement to "walk2". In the "walk2"

block, check for `this.left_upperleg.quaternion.w` - if the value is smaller then 0.93 set this movement to ”walk”. The robot should now be able to move the legs as if it was walking if you call `r.walk();`
See part 2.

Part 6 (10 points): Create `Robot.prototype.onStep` and call `this.onStep()` at the end of each of the ”walk” and ”walk2” blocks from above. In `Robot.prototype.onStep`, use the following code to move the robot: `this.root.translateZ(10);`
robot.js, lines 664-704. I named it ’adjustPosition’ because it does more than just move the robot forward.

Part 7 (20 points): Now, please add safety that the robot does not walk off the plane. Whenever it reaches the ends of the plane (think about which coordinates of `this.root.position` to check), **turn the robot by 180 degrees along the Y axis**. This does not require quaternions but can be done using the `this.root.rotateY` functionality of Three.js.
robot.js, lines 739-754

Part 8 (25 points): Add functionality to `Robot.prototype.onStep` that loops through all the robots in the scene and checks if the current robot (based off `this.root.position`) is close to another one with the `this.root.position.distanceTo()` method. If the robot is close, rotate by 180 degrees. Please don’t forget to exclude the current robot during checks, else it will think it is close to itself and continuously spin without moving forward. You can use the `this.root.position.equals` method to detect equal positions.
robot.js, lines 739-754

Part 9 (1 points): Please update the screenshot above with your own and then post the github pages url here:

`https://jamesedmichaud.github.io`

Bonus (33 points):

Part 1 (10 points): Load a mesh and add two extra point lights to the scene.
I added 5 meshes (technically 2, but one is repeated 4 times).
There’s a skybox of stars line 171 of index.html. I used a very large sphere and used the material’s backside to display the texture.
There are also 4 pillars around the floor, line 178 of index.html. I found a sci-fi texture online and used uv mapping to map only part of the texture onto each pillar (box geometry).
On top of each pillar there is a point light. Red, Blue, Green, and Cyan (blue/green), line 231 of index.html. I used PointLightHelpers to make their positions obvious (I hope that’s ok). To make the light more apparent I removed the main directional light.

Part 2 (23 points): Don’t let the robots walk into the mesh by extending `Robot.prototype.onStep` to compare with the mesh’s bounding box.
Line 755 in robot.js.