John Russell Strauss
902455731
CS 6491
Fall 2019

Tetrobot Phase Two

All relevant Computer Graphics related code is contained within assets\js\components\scene.js.

For phase two, I have built on the functionality of the first phase by adding new methods and refactoring my faulty rotations from phase one. First off, I have added reset functionality (esc key) in order to reset the scene, camera, and position of the robot. This helps the developer test more easily and the user to try again. It is still a work in progress with known issues, however, with my rotation logic working correctly, I can now accurately move the robot. This logic is contained within rotatePointAboutLine(), line 345. I have refactored the logic to calculate the opposite edge midpoint as the robot moves by calculating shared vertices from the current robot position to the previous robot position. Then in combination with the top vertex of the robot, I can calculate a direction parallel to the floor to use as my direction vector. The current issue is with the angle of rotation with each step. My next requirement is to correctly calculate this rotation angle as the robot moves so that he can stay along the floor.

The logic of this rotation method is as follows. Apply the following transformations to both your rotation axis and your point or object in space:

1. Move your rotation axis so that one endpoint lies on origin.
2. Rotate space along the X-axis so that the rotation axis lies on the XZ-plane.
3. Rotation space along the Y-axis so that the rotation axis lies perfectly along the Z-axis.
4. Now perform your original desired rotation along the Z-axis.
5. Reverse the Y-axis rotation previously applied.
6. Reverse the X-axis rotation previously applied.
7. Undo the translation to the origin.
8. Your point is now rotated through 3D space.

I then wrote a method (rotateGeometryAboutLine(), line 411) that took a list of points and applied the rotations to all vertices in the geometry input along the desired axis. This resulted in the entire object rotating through space about any axis of choice defined by two points. From this point, I will be poised to rotate the robot correctly on each step and write easing functions to modify acceleration during the animation loop as a function of time.

Sources

- http://paulbourke.net/geometry/rotate/