John Russell Strauss
902455731
CS 6491

For phase one of this project, I committed myself to using a different technology than the provided Processing example and included methods provided by Jarek. In the first 2 weeks my main priority was to determine if this was a viable route, and I decided that while potentially risky, being forced to develop my own methods to perform many geometrical transformations would promote deeper understanding and strengthen my Computer Graphics muscles. The majority of the time thus far has been spent learning how to display, translate, and rotate geometries and the bare basics of learning how to work with the Three.js framework. I have spent time writing custom methods to create shapes, draw points, display and label axes, calculate midpoints, calculate triangle and tetrahedral centroids, create vectors, get distances, and various other basic functions. All code is original.

All relevant Computer Graphics related code is contained within assets\js\components\scene.js. You will see settings at the top and the rest divided logically in the functions below them. The majority of all geometric logic stems from the addTetrahedron method. This is where the tetrahedron is added and initially rotated to be placed flat on the floor centered on the origin. After rotating the tetrahedron so that its bottom face is parallel with the floor, it is oriented below the ground due to this rotation, so I perform some calculations to move it upward. I calculate the centroid (source below) and use that value to move the tetrahedron up to be flat on the floor. The centroid of the faces and the tetrahedron can be calculated by calculating the midpoints of the respective geometry that you are looking for, whether a triangle face or the whole tetrahedron, the only difference being whether you average 3 or 4 vertices. The math for this calculation is contained within the method called calculateCentroidLocation. Once aligned, I can begin making calculations for the tetrahedral movement. I created a method that will let me define a rotation axis by any two points and then rotate any geometry around the axis created by those points by any angle of my input. I have gotten parts of it working, but there are problems with this method that I am still trying to locate. I was able to get the correct rotations by changing the inputs, but I think it is causing compounding issues in the rotations and therefore preventing my program from functioning correctly. For example, every roll is correct on the first iteration. But after the first iteration, only alternating left and right directions function correctly. I believe this rotation function (threeStepRotation) is to blame, and this is the fist thing I need to fix before proceeding further. To see an example of an error-less working path, use LRLRLRLRL as your inputs to demonstrate the basic functionality.

The logic of this rotation method is as follows:

1. Take rotation axis V and move it to the origin. Perform same translation on object being rotated.
2. Rotate V onto the YZ plane. Measure this rotation as angle -Theta. You can calculate the value of the angle with arctan(Vx/Vz).
3. Rotate V to be co-linear with the Z-axis. Measure this angle as Phi. You can calculate this angle with arctan(Vy/sqrt(Vx^2 + Vz^2)). These are the singular components of V using Pythagorean theorem.
4. Now rotate object by this angle Phi. Since our rotation axis is now co-linear with the Z-axis, we can now rotate our object by our original desired angle. The remaining step involved performing the remaining 3 steps in reverse: rotate the object around the X-axis by -Phi, rotate around Y by Theta, and then perform the same translation in step 1, but in the negative direction.

Using this rotation and defining my rotation axis as the edge of one of the bottom faces of the tetrahedron (pivot), I can begin rotating the entire object to make it move from step to step. Since the equilateral triangle has interior angles of 60°, I can rotate it 120° (because 180° - 60°), or 2 * PI / 3. I then merely create a method for applying each direction, update the location, and pass the updated geometry location into the same method to take further steps.

Sources

- Calculate the centroid of a triangle and tetrahedron: https://youtu.be/Infxzuqd_F4
- Rotation Around an Arbitrary Axis – UC Davis Academics: https://www.youtube.com/watch?v=gRVxv8kWl0Q&t=1224s