OVERVIEW:

- This an iOS App developed using Swift and GLSL in Xcode.
- It draws Poly-lines with every touch on the screen and also locates its centroid as soon as there are more than two edges. Individual vertex, edge or centroid can be translated to another point on screen.

TO RUN :

- Open the a03.xcodeproj file in Xcode.
- Run the project after setting the active scheme as iPhone6
- The can be directed to an actual handset by selecting the connected iPhone as the active scheme.

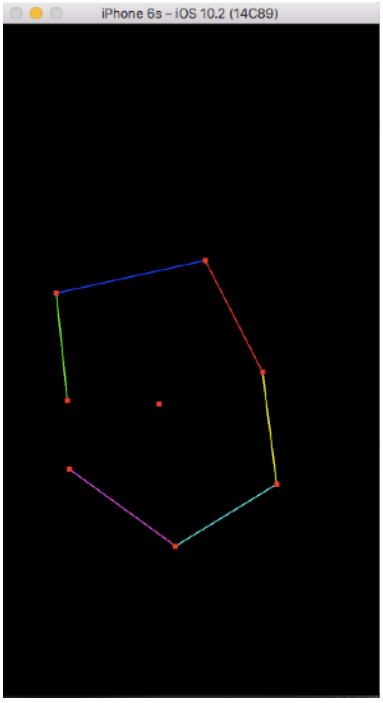
BEHAVIOR:

- When the app launches in simulator, any tap on the screen will add a vertex(using GL_POINTS) at the tapped coordinates on screen.
- As soon as there is more than one touch, each touch will add vertex and draw edges(using GL_LINE STRIP) from the last vertex to the current vertex resulting in a poly-line.
- As soon as there are two or more lines, it will also display its centroid drawing one point (using GL_POINTS).
- This Poly-line is flexible and any edge or vertex can be grabbed and moved/translated. But this behaviour is retained until the centroid has not been grabbed and moved.
- Once the centroid is grabbed and moved, neither vertices can be added/translated nor edges can be translated.
- Additionally, after one translation of the centre of mass, any touch start or touch moved event on the screen will cause only rotation of the poly-line about its centre of mass.
- This project is a combination of translation and rotation done on CPU side without using matrices.

THE CODE

- In Xcode go to the GameViewController.swift from the Project Navigator. This file has the 90% of the logic including touch-listeners and proximity testing.
- After all the touch inputs and proximity calculations, this file passes the uniform variables and attribute values to vertex shader.
- The vertex shader (written in GLSL) calculates the orthogonal projection matrix and multiplies it to the vertex coordinate passed by the CPU code and sends them to the fragment shader.
- Fragment shader has been kept simple for this project and does not include any extra task.

SCREENSHOT:



DEPENDENCIES:

- The dependencies are: it might not draw the solid lines for some models other than iPhone6.

LIMITATIONS:

- In this project no matrix transformations have been used. To implement translation of control points, traditional insert and delete functions have been used on vertex arrays. Hence greater part of the work is being done by CPU which can be delegated to GPU using matrix transformations.