# Project Proposal

I plan to create a mini 3D graphics software. For 2D graphics, the average user can easily can access to paint to draw these graphics. However for 3D graphics, the average user has to download and purchase 3D graphic modelling software such as Blender, 3DS Max which as costly and large to download. Hence it will be difficult for average users to use small and easily distributable 3D graphic software offline. Therefore, I want to make an easily accessible and small-sized 3D graphics software for the average user. Furthermore, the typical graphic software does not draw points and link those points together to form lines, curves and faces.

I intend to solve by building a mini 3D graphics engine using Tkinter. In order to build this 3D graphics engine, I have to learn and use various techniques such as

1. 3D rotation (3 separate rotation along XY, XZ and YZ axis)
2. Transform 3D vectors to screen x and screen y and vice versa (Matrix solving)
3. 3D display sequence by depth (sorting by depth)
4. Collision detection between mouse and 3D graphical objects (Ray Casting)
5. Create curves (Beizer curve)
6. Transparency (Tkinter stipple)
7. Load and Save files (file read and write)

There will be 3 main classes, point, line and polygon classes. Points contain the coordinates, color and name. Lines contain either 2 points for straight lines and 3 points for curves. The polygon classes will contain finite number of points.

For curves in polygon, the vectors are firstly mapped to screen X and screen Y. With both X and Y, the coordinates for the curve will be computed using the beizer curve formula taking the middle point as the control point. After that, the points will be passed into a polygon to draw the final curve.

As for collision detection, the list of points will be evaluated using Bounding Box and Ray Casting. For Ray Casting, the code will find pairs of points where the mouse Y is between these points. Then it will count the number of such points to the left of the mouse X. If it is odd, then the mouse in the object, if it is even, then the mouse is out of the object.

Converting screen X and Y to model X,Y,Z require the use for matrix solver. For the other way, it is a simple matrix multiplication process. 3D rotation is achieved by breaking it down into 3 different types of rotations, find the radius, angle and the change in angle to obtain the new coordinates.

# Technology Demonstrations

Refer to file Project\_Cube.py and Raycast\_Curve.py for demonstration of knowledge of 3D rotation, ray casting, and beizer curve.

# Competitiveness analysis

3D Function Grapher

Website: <http://www.math.uri.edu/~bkaskosz/flashmo/graph3d/>

|  |  |
| --- | --- |
| **Features** | **Anti-Features** |
| Curve drawing | Use equations to define the curves |
| Orthographic projection | Can only graph one at a time |
| Small size | No Save |

Blender

Website: <http://www.blender.org/>

|  |  |
| --- | --- |
| **Features** | **Anti-Features** |
| Allow editing of vertex | Grid |
| 3D Rotation | Camera |
| Local save | Animation |

3D Tin: 3D modelling for everyone (Online)

Website: [http://www.3dtin.com/#](http://www.3dtin.com/)

|  |  |
| --- | --- |
| **Features** | **Anti-Features** |
| 3D Rotation | Draw basic 3D shapes to form bigger 3D shapes |
| Draw curved objects | Online save |
| Change color | Grid |

# Storyboard

# 