Assignment 2

3D Projections

**Task**

Implement 3D line drawing by projecting models onto the view-plane. You will use HTML's Canvas 2D API.

**3D Projections (to earn a C: 45 pts)**

* Implement perspective projection for 3D models: **35 pts**
  + Transform models into canonical view volume
    - Implement the matrix functions in transforms.js
  + Implement Cohen-Sutherland 3D line clipping
  + Project onto view plane
  + Draw 2D lines
* Implement camera movement to change the view of a scene: **10 pts**
  + A/D keys: translate the PRP and SRP along the u-axis
  + W/S keys: translate the PRP and SRP along the n-axis

**Additional features (to earn a B or A)**

* Implement parallel projection for 3D models: **5 pts**
  + Follows same steps as perspective
* Generate vertices and edges for common models: **5 pts**
  + Cube: defined by center point, width, height, and depth (1 pt)
  + Cone: defined by center point of base, radius, height, and number of sides (1 pt)
  + Cylinder: defined by center point, radius, height, and number of sides (1 pt)
  + Sphere: defined by center point, radius, number of slices, and number of stacks (2 pts)
* Allow for models to have a rotation animation: **5 pts**
  + Can be about the x, y, or z axis
  + Defined in terms of revolutions per second
* Left/right arrow keys: rotate SRP around the v-axis with the PRP as the origin: **5 pts**

**Scene**

Scenes will be defined as a JavaScript object. The scene will contain both view parameters and a description of the models.

view:

* type (perspective / parallel)
* prp
* srp
* vup
* clip (array - left, right, bottom, top, near, far)

models (array):

* type = generic
  + vertices (array of Vector4)
  + edges (array of lines)
    - line: array of vertex indices
* type = cube
  + center (Vector4)
  + width
  + height
  + depth
* type = cone
  + center (Vector4)
  + radius
  + height
  + sides
* type = cylinder
  + center (Vector4)
  + radius
  + height
  + sides
* type = sphere
  + center (Vector4)
  + radius
  + slices (think number of longitude lines on a globe)
  + stacks (think number of latitude lines on a globe)
* All modes also optionally may have an 'animation' field
  + animation
    - axis (x, y, or z)
    - rps (revolutions per second)

\*Note: sample models can be found in the starter code.

**Starter Code**

Starter code is available on GitHub: [cg-3dprojections (Links to an external site.)](https://github.com/tmarrinan/cg-3dprojections). Please **fork** your own version of the code, then enable GitHub Pages in the project's settings (change *Source* from *None* to *master branch*).

**How this Relates to My Capstone Project**

This assignment relates to the Desktop Application Project because I need to perfectly size and place the wireframe objects just like the buttons and drop-down menus in the FXML files. If I do not properly size or place the wireframe objects, buttons, or drop-down menus, they can be too big, small, or be somewhere offscreen, which causes problems for the user.