Sean O’Hagan

GyroScope

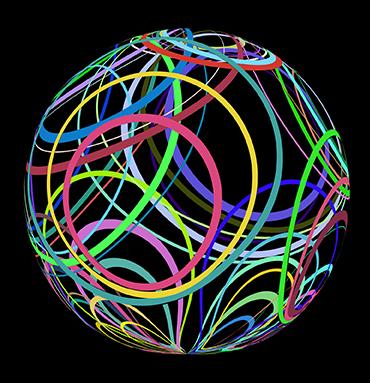
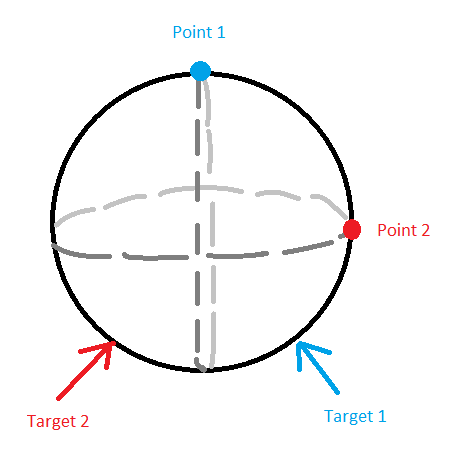


Image Designed by Kjpargeter / Freepik

# Executive Summary

For this project, I will be creating an educational visualizer for the various rotation techniques that can be applied to 3D objects. The “core” of this project is a (cross-platform) imaging app with a single, stationary 3-dimensional plane upon which a single object can be rotated. In the application’s menus, the method of rotation (Euler angle, Matrix, Quaternion) will be selectable and tooltips will provide explanations for how the math is being calculated behind the scenes (including advantages/drawbacks of each method).

In order to move beyond tooltips and tangibly demonstrate the workings of each rotation technique, the application will center around a simplistic game: there will be one single object, floating centered in the plane, with 2 distinctly colored points on its surface, placed at intersections of the object’s canonical x, y, and z axes. In the plane, target points will appear - the object must be rotated so that its colored points align with the correct target point (checks will occur to ensure target points are valid - requiring rotations, but not scales or transforms for the points to align).



This visual/interactive space will help contextualize concepts like gimbal lock and composed rotations - the areas where each rotational method has the clearest advantages and disadvantages.

Beyond (hopefully) being an engaging educational experience, I intend for this application to serve as a first draft/proving ground for an extensible general-purpose 3D rendering engine, and will architect it in such a way that allows for future extensibility.

# Development Tools

This project will be developed entirely in C++ and GLSL (OpenGL Shader Language). The GLFW and GLAD libraries will be used to provide a cross-platform rendering context and automatically map the OpenGL function prototypes to the machine’s specific graphics drivers, respectively. The Dear ImGui library will be used to provide an immediate-mode GUI framework upon which the application’s menus will be developed.

Shaders, shader management systems, and math libraries will be hand-written.

# Technical Background

I am extremely comfortable working in C/C++ for low-level systems architecture - I’ve been using the language in this context for personal projects for 5 years, and professionally since 2022. I have some experience with linear algebra and 3D rendering in an OpenGL environment - not so much that the specifics of this project touch anything I’ve directly worked on before, but enough that I won’t be approaching the larger concepts for the first time. Similarly, I’ve developed and delivered a number of applications with retained-mode GUIs using a variety of frameworks - I have experience with the larger concepts, but have not worked specifically in immediate-mode before.

# Project Risks

The largest risk to this project is the labor cost of performing the mathematical calculations for quaternion rotation on the “correct” hardware (if you’re an optimization buff). I am confident that I can implement a math library for quaternion rotations, running on the CPU, within the allotted time. To move these calculations to the GPU would require the use of compute shaders, which involve a great deal more low-level control of the GPU processing apparatus than performing calculations within the normal rendering pipeline, and are a very separate field of research from the areas covered by the majority of this project.

I plan to mitigate this risk by considering full hardware acceleration a “stretch goal” of the project. My focus will be on implementing the requisite math using a combination of CPU and GPU processing, moving only those tasks to the GPU that are directly supported by the rendering library as-is. If I am able to complete the “core” of this project early, only then will I begin work to write compute shaders and move the more complex mathematical operations GPU-side.

# Time & Cost Estimate

| **Item** | **Hours** | **Unit Cost**  **or % of Programming Time** | **TOTAL** |
| --- | --- | --- | --- |
| **Development Tools** |  |  | $0 |
| **Misc. Expenses** |  |  | $0 |
| **Programming Time (In Hours)** | 100 | $110 | $11,000 |
| **Project Management** | 20 | $90 | $1,800 |
| **Administration** | 10 | $80 | $800 |

**TOTAL COST ESTIMATE:** **$13,600**

# Future

The larger intent behind this project is to create a groundwork architecture that can be expanded into an educational renderer for general-purpose 3D topics. If all goes well, in the future, I plan to be able to add scaling/transform functions, multiple objects, and user-customizable filters and image processing.

Outside of a hobby project, though, my company currently provides a real-time data processing and display application that is very capable for high speed image processing research, but also horrendously outdated and not at all user friendly in many critical areas. My hope is to push for an internal project to revamp critical areas of that application, using this project as a proof-of-concept for what a modern, cross-platform research tool could look like with the right investment.