Brock University
Faculty of Mathematics & Science
Department of Computer Science

COSC 3P93: Parallel Computing

4P78 Project Step 2

Prepared By:

Parker TenBroeck 7376726
pt21zs@brocku.ca

Instructor:
Robson De Grande

October 8, 2025

1 Intro

2 Algorithm Overview

2.1 Projection

$$
\begin{aligned}
&fov = \text{the field of view for our camera}\\
&far = \text{the farthest point our camera can see}\\
&near = \text{the closest point our camera can see}\\
&M_{projection} = \begin{bmatrix} \frac{1}{aspect*\tan(\frac{fov}{2})} & 0 & 0 & 0 \\ 0 & \frac{1}{\tan(\frac{fov}{2})} & 0 & 0 \\ 0 & 0 & -\frac{far+near}{far-near} & -\frac{2\times far\times near}{far-near} \\ 0 & 0 & -1 & 0 \end{bmatrix}
\end{aligned} \tag{1}
$$

2.2 View

$$
\begin{aligned}
&P_{target} = \text{Where the camera is looking}\\
&P_{eye} = \text{Where the camera is}\\
&\overrightarrow{V_{world\ up}} = \text{the up direction of the world } \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}\\
&\overrightarrow{V_{cam\ forward}} = \langle P_{target} - P_{eye} \rangle\\
&\overrightarrow{V_{cam\ right}} = \left\langle \overrightarrow{V_{cam\ forward}} \times \overrightarrow{V_{world\ up}} \right\rangle\\
&\overrightarrow{V_{cam\ up}} = \left\langle \overrightarrow{V_{cam\ right}} \times \overrightarrow{V_{cam\ forward}} \right\rangle\\
&M_{view} = \begin{bmatrix} \overrightarrow{V_{cam\ rightx}} & \overrightarrow{V_{cam\ righty}} & \overrightarrow{V_{cam\ rightz}} & \overrightarrow{V_{cam\ right}} \cdot P_{eye} \\ \overrightarrow{V_{cam\ upx}} & \overrightarrow{V_{cam\ upy}} & \overrightarrow{V_{cam\ upz}} & \overrightarrow{V_{cam\ up}} \cdot P_{eye} \\ -\overrightarrow{V_{cam\ forwardx}} & -\overrightarrow{V_{cam\ forwardy}} & -\overrightarrow{V_{cam\ forwardz}} & \overrightarrow{V_{cam\ forward}} \cdot P_{eye} \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{2}
$$

2.3 Model

$$
\begin{aligned}
&P_{pos} = \text{world space location of the object} \\
&\overrightarrow{V_{scale}} = \text{x, y, z scale of object} \\
&A_{rotation} = \text{euler angles of objects rotation, raw,pitch,roll} \\
&M_{yaw} = \begin{bmatrix} \cos(yaw) & -\sin(yaw) & 0 \\ \sin(yaw) & \cos(yaw) & 0 \\ 0 & 0 & 1 \end{bmatrix} \\
&M_{pitch} = \begin{bmatrix} \cos pitch & 0 & \sin(pitch) \\ 0 & 1 & 0 \\ -\sin pitch & 0 & \cos pitch \end{bmatrix} \\
&M_{roll} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(roll) & -\sin(roll) \\ 0 & \sin(roll) & \cos(roll) \end{bmatrix} \\
&M_{rot} = M_{yaw} \cdot M_{pitch} \cdot M_{roll} \\
&M_{scale} = \begin{bmatrix} \overrightarrow{V_{scalex}} & 0 & 0 \\ 0 & \overrightarrow{V_{scaley}} & 0 \\ 0 & 0 & \overrightarrow{V_{scalez}} \end{bmatrix} \\
&M_{rot\ scale} = M_{rot} \cdot M_{scale} \\
&M_{model} = \begin{bmatrix} M_{rot\ scale11} & M_{rot\ scale12} & M_{rot\ scale13} & P_{pos_x} \\ M_{rot\ scale21} & M_{rot\ scale22} & M_{rot\ scale23} & P_{pos_y} \\ M_{rot\ scale31} & M_{rot\ scale32} & M_{rot\ scale33} & P_{pos_z} \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{3}
$$

2.4 Clip

$$
\begin{aligned}
&V_{world} = M_{projection} \cdot M_{view} \cdot M_{model} \cdot V_{local} \\
&V_{clip} = M_{projection} \cdot M_{view} \cdot M_{model} \cdot V_{local} \\
&M_{normal} = {M_{model}^{-1}}^T \\
&\langle N_{world} \rangle = M_{normal} \cdot \langle N_{local} \rangle
\end{aligned}
$$

3 Paralization Opportinuty

3.1 Per Triangle

4 Libraries Used

Used `tinyobjloader` [3] for loading and parsing OBJ and MTL files.

Used `stb_image` [1] for loading and parsing texture files into linear RGB.

Used `stb_image_write` [2] for writing rendered frames to disk.

References

[1] S. Barrett, "stb image." [Online]. Available: https://github.com/nothings/stb

[2] ——, "stb image write." [Online]. Available: https://github.com/nothings/stb

[3] S. Fujita, "Tiny obj loader." [Online]. Available: https://github.com/tinyobjloader/tinyobjloader