Mathematical Foundations for 3DGraphicsEngine

Dominick Harasimiuk

July 29, 2018

1 Introduction

This document is intended to highlight the underlying workings of the graphics engine. Describing these mathematical workings enables the curious developer to implement their own code to reproduce the functionality of the engine. Additionally, it offers the benefit of allowing the community to improve on the backbone of this project if more suitable mathematical avenues are found.

This graphics engine takes input of arbitrary position vectors, $\mathbf{v} \in \mathbb{R}^3$, where $\mathbf{v} = \langle a, b, c \rangle$ for some $a, b, c \in \mathbb{R}$. These vectors are used as a means by which points in $\mathbb{R}^3$ can be located and manipulated. To retrieve points in $\mathbb{R}^2$ that can be displayed on a computer screen, the vectors are projected onto a chosen viewing plane, P. This plane contains the origin and is defined by a chosen normal vector, $\mathbf{n}$, that "points" at the observer. The details of P will be discussed in section 2. Once the position vectors are projected onto P, they are transformed into vectors in $\mathbb{R}^2$ using their relative magnitudes and interior angles. This process will be discussed further in section 3.

2 Projecting Vectors

This section describes how an arbitrary vector, $\mathbf{v} = \langle a, b, c \rangle$ is projected onto the plane, P, defined by normal vector, $\mathbf{n}$.

The viewing plane, P, with normal $\mathbf{n}$ can be defined as follows:

$$P = \{(x, y, z) \in \mathbb{R}^3 | n_1x + n_2y + n_3z = 0\}$$
$$\mathbf{n} = \langle n_1, n_2, n_3 \rangle$$

Notice that P will intersect the origin for any normal vector $\mathbf{n}$. If we allow the components of $\mathbf{n}$ to vary, we can achieve any desired viewing angle within the graphics engine. Now, suppose we have an arbitrary vector $\mathbf{v} = \langle a, b, c \rangle$ and we would like to project it onto P. This proceeds as follows:

We form projection $\mathbf{p}$ by adding a scalar multiple of $\mathbf{n}$ to $\mathbf{v}$:

$$\mathbf{p} = \mathbf{v} + k\mathbf{n}$$

We then stipulate that $\mathbf{p}$ must lie in P, hence its components must satisfy $n_1x + n_2y + n_3z = 0$:

$$n_1(a + kn_1) + n_2(b + kn_2) + n_3(c + kn_3) = 0$$

Solving for k, we find that $\forall \mathbf{v}, \mathbf{n} \in \mathbb{R}^3$:

$$\begin{aligned}
n_1(a + kn_1) + n_2(b + kn_2) + n_3(c + kn_3) &= 0 \\
n_1a + kn_1^2 + n_2b + kn_2^2 + n_3c + kn_3^2 &= 0 \\
n_1a + n_2b + n_3c + k(n_1^2 + n_2^2 + n_3^2) &= 0 \\
-\frac{n_1a + n_2b + n_3c}{n_1^2 + n_2^2 + n_3^2} &= k \\
-\frac{\mathbf{n} \cdot \mathbf{v}}{||\mathbf{n}||^2} &= k
\end{aligned}$$

This constant k allows for the projection of any vector onto the desired viewing plane. To illustrate that this methodology is workable, let us proceed with an example. Suppose we want to project the unit vector in the z direction, $\mathbf{k}$, onto P with $\mathbf{n} = \langle 1, 1, 1 \rangle$. This would proceed as follows:

We find the constant k:

$$\begin{aligned}
k &= -\frac{\mathbf{n} \cdot \mathbf{k}}{||\mathbf{n}||^2} \\
k &= -\frac{1 \cdot 0 + 1 \cdot 0 + 1 \cdot 1}{3} \\
k &= -\frac{1}{3}
\end{aligned}$$

We now compute the projection vector $\mathbf{p}$:

$$\begin{aligned}
\mathbf{p} &= \mathbf{k} + k\mathbf{n} \\
\mathbf{p} &= \mathbf{k} - \frac{1}{3}\mathbf{n} \\
\mathbf{p} &= \langle -\frac{1}{3}, -\frac{1}{3}, \frac{2}{3} \rangle
\end{aligned}$$

Now, if we visualize the plane P, the unit vector $\mathbf{k}$, and our projection $\mathbf{p}$, it becomes intuitively apparent that our projection is sensible. We can expect $x, y < 0$ while $z > 0$ given the orientation of P. We can also expect that $||\mathbf{p}|| < ||\mathbf{k}||$.

3 Graphing Projections

This section describes how we transform a projected vector, denoted by $\mathbf{p}$, in order to display it on a two dimensional plane, referred to as the "screen" in order to distinguish it from the viewing plane (a plane in three dimensions) of prior discussion. We will discuss a limiting but still versatile case by fixing the projection of $\mathbf{k}$, the unit vector in the z direction,to lie along the y axis of our screen. We will call this projection $\mathbf{k}_p$. This enables us to plot all other projections relative to $\mathbf{k}_p$ using the varying magnitudes and angles between vectors.

Recall that if $\mathbf{v} = \langle a, b, c \rangle$, then

$$||\mathbf{v}|| = \sqrt{a^2 + b^2 + c^2}$$

Also, recall that for any $\mathbf{u}$, $\mathbf{v}$ we have

$$\mathbf{u} \cdot \mathbf{v} = ||\mathbf{u}|| ||\mathbf{v}|| \cos\theta$$

Thus, we can find the angle between two vectors as follows:

$$\theta = \arccos \frac{\mathbf{u} \cdot \mathbf{v}}{||\mathbf{u}|| ||\mathbf{v}||}$$

It is important to note that this will return a $\theta \in [0, \pi]$. This leaves us unsure of which side of the screen, relative to $\mathbf{k}_p$, an arbitrary projection $\mathbf{p}$ lies on. To clear up this ambiguity, we can do the following:

Letting $\mathbf{p} = \langle x_0, y_0, z_0 \rangle$, we construct a line in the xy plane on which the projection lies above

$$y = \frac{y_0}{x_0} x$$

If $x_0 < 0$ then we treat the region above the line as the "right" side of the screen, whereas if $x_0 > 0$, we treat the region above the line as the "left" side of the screen. If $x_0 = 0$, then we consider y_0. If $y_0 > 0$ then we assign the regions naturally, meaning that $x_0 > 0$ implies "right" and $x_0 < 0$ implies "left". If $y_0 < 0$, then we assign the regions opposite of the aforementioned assignment.

Now that we can determine an arbitrary projection's orientation relative to $\mathbf{k}_p$, we can plot it on the screen. Letting $\mathbf{p} = \langle x_0, y_0, z_0 \rangle$, this proceeds as follows:

We compute

$$\theta = \arccos \frac{\mathbf{k}_p \cdot \mathbf{p}}{||\mathbf{k}_p|| ||\mathbf{p}||}$$
$$y = \frac{y_0}{x_0} x$$

Depending on our values for x_0, y_0, we determine our region assignments and compute ϕ, which will act as our polar coordinate plotting angle (Note that the $\pm$ below is determined by region assignment)

$$\phi = \frac{\pi}{2} \pm \theta$$

Using $r = ||\mathbf{p}||$ we can plot our projection onto the screen with

$$x = r \cos \phi$$
$$y = r \sin \phi$$

4 Conclusion

Demonstrating the mathematics that allow the graphics engine to operate is the purpose of this document. That being said, it is important to acknowledge certain shortcomings of this mathematical methodology. One such shortcoming is the fact that this methodology does not support viewing vectors without fixing $\mathbf{k}_p$ to lie along the y axis of the screen. While this does still allow for plenty of rotation and viewing angles, it is nonetheless limiting and worth acknowledgment. Another shortcoming (albeit a potential one) is that this document does not currently highlight the process of dealing with edge cases that could potentially produce problems in the engine. Examples of these include **k** being the normal vector to the viewing plane, classification of **-k** into appropriate regions, and other miscellaneous cases that primarly have to do with the region selection criteria. As this project progresses, such edge cases will be investigated and resolved.

Thank you for taking the time to read this document!