Lecture 23:
Before Fall Break Loose Ends

November 21, 2019
Five Topics Today

• Reflections on Debugging in CS 410
• Z-buffers and psydo-depth
• Thin Lens Modeling Kept Simple
• Six Degree of Freedom Mapping
• Overview Programming Assignment 5
CS 410 and Debugging

• About Debuggers
  – Need them for segmentation fault line numbers
  – Otherwise, a mixed blessing (too much info.)

• Small scale testing
  – Render something simple!
  – Instrument your code (means print statements)
  – Compute it two ways

• Bigger more complex issues
  – Spreadsheets can be very useful!
Projection Pipeline and Depth
Render this Rectangle

Z-Buffer

Red Plane

Green Plane

Blue Plane

What colors and when.
Using a Z-Buffer

• Record depth at every vertex
• For every pixel in polygon (previous lecture)
  – Interpolate to get depth at specific pixel.
  – Is depth less than currently recorded?
    • Yes: Record in Z-Buffer and paint pixel
    • No: Move along, nothing to do here
• “Paint” is shorthand for compute the surface illumination for that position on the polygon.
About depth: the z-value

• Z-buffering based upon pseudo-depth is key to modern polygon rendering.

• Depth already revealed in SageMath notebook on the Canonical View Volume.

• Here let us briefly dive into the calculation of pseudo-depth using essentially that example.
SageMath Notebook

- Emphasize the z coordinate of transform

\[
P_{cc} = \begin{bmatrix}
\frac{u_{max}+u_{min}}{2} & \frac{u_{max}-u_{min}}{} & 0 & \frac{u_{max}+u_{min}}{} & 0 \\
\frac{v_{max}+v_{min}}{2} & \frac{v_{max}-v_{min}}{} & 0 & \frac{v_{max}+v_{min}}{} & 0 \\
\frac{2 \text{ far near}}{\text{ far near}} & \frac{\text{ far near}}{\text{ far near}} & z & \frac{2 \text{ far near}}{\text{ far near}} & \frac{\text{ far near}}{\text{ far near}} \\
1 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

and the z term only

\[
p_{z} = \frac{2 \text{ far near}}{\text{ far near}} - \frac{(\text{ far near})z}{\text{ far near}}
\]
First the Symptom

Near = -25
Far  = -75

Near =  -25
Far  = -750

Near =  -25
Far  = -7500

Remember, the house lies between z of 30 and 54 in world coordinates.

Even pushing the far clipping plane 2 orders of magnitude further back from -75 still results in the house occupying most of the pseudo-depth range between 0 and 1.
Back to the Math

• Camera at origin no world cam. rotation

\[
\begin{pmatrix}
\frac{(u_{\text{max}}+u_{\text{min}})z}{u_{\text{max}}-u_{\text{min}}} \\
\frac{(v_{\text{max}}+v_{\text{min}})z}{v_{\text{max}}-v_{\text{min}}} \\
\frac{2 \text{near}}{\text{far}\text{-near}} - \frac{(\text{far}+\text{near})z}{\text{far}\text{-near}}
\end{pmatrix}
= \begin{pmatrix}
\frac{2 \text{near}}{u_{\text{max}}-u_{\text{min}}} \\
0 \\
0 \\
0
\end{pmatrix}
\begin{pmatrix}
0 \\
\frac{2 \text{near}}{v_{\text{max}}-v_{\text{min}}} \\
0 \\
0
\end{pmatrix}
\begin{pmatrix}
\frac{\text{far}+\text{near}}{\text{far}\text{-near}} \\
0 \\
0 \\
1
\end{pmatrix}
\begin{pmatrix}
0 \\
0 \\
0 \\
z
\end{pmatrix}
\]

\[
P_{cc} = \begin{pmatrix}
\frac{u_{\text{max}}+u_{\text{min}}}{u_{\text{max}}-u_{\text{min}}} \\
\frac{v_{\text{max}}+v_{\text{min}}}{v_{\max}-v_{\text{min}}} \\
\frac{2 \text{near}}{\text{far}\text{-near}} - \frac{(\text{far}+\text{near})z}{\text{far}\text{-near}}
\end{pmatrix}
\begin{pmatrix}
\frac{2 \text{near}}{\text{far}\text{-near}} \\
\frac{\text{far}+\text{near}}{\text{far}\text{-near}}
\end{pmatrix}
\]

and the z term only

\[
p_{z} = \frac{2 \text{near}}{\text{far}\text{-near}} - \frac{(\text{far}+\text{near})z}{\text{far}\text{-near}}
\]

Pseudo-depth
\(pz\) At near and far

- Equation:
  \[
  pz = \frac{2 \cdot far \cdot near}{(far - near) \cdot z} - \frac{far + near}{(far - near)}
  \]

Let \(z\) equal \(near\)

\[
pz = \frac{2 \cdot far \cdot near}{(far - near) \cdot near} - \frac{far + near}{(far - near)}
\]

\[
pz = \frac{2 \cdot far - far - near}{(far - near)}
\]

\[
pz = \frac{far - near}{(far - near)}
\]

\[
pz = 1
\]

Similarly ...

Let \(z\) equal \(far\)

\[
pz = -1
\]
Plot actual Depth to Pseudo-depth

Far = -75
Plot actual Depth to Pseudo-depth

Far = -750
Plot actual Depth to Pseudo-depth

Far = -7,500
There are various ways to interpolate in order to arrive at an estimated z-value for an interior point on any given triangle.

Common is to first interpolate up the sides and then to interpolate across.
Z-Buffer Summary

• A Z-buffer is an array of doubles
• Size of the frame buffer / image
• Initialized to -1.0, i.e. far clipping plane
• Now consider a specific triangle
• For each pixel to be filled
  – Interpolate pixels z-value
  – Test if larger then what is in the Z-buffer
  – If yes then “paint” that pixel for that triangle
What if you Want Depth?

- Mapping may be inverted.

\[ pz = \frac{2 \times \text{far} \times \text{near}}{\text{far} - \text{near}} \times z - \frac{\text{far} + \text{near}}{\text{far} - \text{near}} \]

\[ z = \frac{2 \times \text{far} \times \text{near}}{\text{far} - \text{near}} \times pz + \text{far} + \text{near} \]

There are worse things then checking your work in a symbolic math package.
Thin Lens Modeling

Screen

Lens

Optical Axis

Aperture
Thin Lens Model

- Parallel rays on one side converge at focal point on the other side.
- Rays diverging from the focal point become parallel.
Thin Lens Model

• Thus many paths join together.
Thin Lens Constraints

#1 All rays emanating from a single point in space must converge on a single point in the image plane (definition of focus)

#2 Any ray entering the lens parallel to the axis on one side goes through the focus point on the other side

#3 Any ray entering the lens from the focus point on one side emerges parallel to the axis on the other side
Fundamental Equation of Thin Lenses

Note: $P$ is “not too far” from optical axis
Fundamental Equation (II)

- The ray PQ (parallel to the optical axis) must be deflected to pass through FR by property #2.
- The ray PR must be deflected so that it becomes parallel to the optical axis by property #3.
- After deflection, PQ & PR must intersect at p, by property #1.
- Now, use similar triangles....
Fundamental Eq. (III)

Substitute for $y$ and solve: $f^2 = zZ$, or

$$\frac{1}{Z + f} + \frac{1}{z + f} = \frac{1}{f}$$
Out of Focus Images

What happens when \( \frac{1}{Z+f} + \frac{1}{z+f} \neq \frac{1}{f} \)

Out-of-focus image planes

Optical Axis \( F_L \)

Spherical Blurring

Out of Focus Images

What happens when \( \frac{1}{Z+f} + \frac{1}{z+f} \neq \frac{1}{f} \)
Same picture as before – new question:
Given a point $p$ in the camera (a pixel), where in the world is $P$ such that all rays through the lens from $P$ to $p$ focus perfectly?
Point of Focus

• The position of $P = (X, Y, Z)$ can be calculated by finding the intersection of two rays that converge at $(a, b, z)$

• One ray goes through the left focal point, strikes the lens at $(a, b, L)$, and proceeds parallel to the optic axis to $(a, b, z)$

• One ray goes from $(X, Y, Z)$ parallel to the optic axis until it strikes the lens, and is then reflected through the right focal point to $(a, b, z)$
Keep It Simple

• Arbitrarily set a depth for perfect focus.

Compute the 3D target point by going an amount tau along the ray from the pixel. Instead of firing one ray from the pixel 3D position to the target, fire k from slightly different pixel coordinates sampled around the true position.
Keep It Simple

• Showing a couple of sample rays.

The size of the sample pattern is exaggerated above.
Texture Coordinates Solve Transform

This is a quick workup of solving for texture coordinates given three pairings. In the next equation the six degree of freedom transformation mapping from a 3D coordinate on a surface and a 2D coordinate in a texture map is expressed using six free variables indicated ‘a’ through ‘f’.

Ross Beveridge, November 21, 2019

\[
\begin{align*}
\begin{bmatrix}
 u \\
 v \\
\end{bmatrix}
&= 
\begin{bmatrix}
 a & b & c \\
 d & e & f \\
\end{bmatrix}
\begin{bmatrix}
 x \\
 y \\
 z \\
\end{bmatrix}
\end{align*}
\]