3D Graphics

The rendering pipeline
Rasterizer
Find which pixel is inside which triangle

Emit fragments (candidates pixel)
Problem description:

Given:
- A triangle
- A 2D grid of x by y

Which cell center of the grid is in the triangle
What is the cell center coordinates?

Cell Center coordinates

Cell $i,j$

$i \in [0,w]$

$j \in [0,h]$

(1 minute : alone)
(2 minutes : neighbors)
(5 minutes : groups)
Cell Center coordinates in projected space

\[
x = \frac{(i + 0.5)}{w} \times 2 - 1
\]
\[
y = \frac{(j + 0.5)}{h} \times 2 - 1
\]
Inside - outside check

Is p inside the triangle?
Inside - outside check

Is \( p \) inside the triangle?

How do we know it?
The edge side function

The edge side function return:

A positive value on one side of the edge

A negative value on the other side of the edge

edgeSide(p,v0,v1)
The edge side function: Warning

Reversing the edge orientation change the returned sign

edgeSide(p,v1,v0)
How do we know $p$ is inside the triangle?

(1 minute alone)

(2 minutes with your neighbors)

(5 minutes with the whole group)
Answer:

If \( \text{edgeSide}(p,v0,v1) \geq 0 \) and \( \text{edgeSide}(p,v1,v2) \geq 0 \) and \( \text{edgeSide}(p,v2,v0) \geq 0 \)

Then \( p \) is inside

Only works for correctly oriented triangles!
The edge side function

\[ \text{edgeSide}(p, v_0, v_1) = (p.x - v_0.x)(v_1.y - v_0.y) - (p.y - v_0.y)(v_1.x - v_0.x) \]

2D cross product magnitude

\[ \text{edgeSide}(p, v_0, v_1) = \|(p - v_0) \times (v_1 - v_0)\| = \|(p - v_0)\| \|v_1 - v_0\| \sin(\theta) \]

Sign come from the sin function
The edge side function

\[
\text{edgeSide}(p, v0, v1) = (p.x - v0.x)(v1.y - v0.y) - (p.y - v0.y)(v1.x - v0.x)
\]

Signed Area

\[
\text{edgeSide}(p, v0, v1) = \text{Area}(\text{parall\'e}lograme) = 2 \times \text{Area}(p, v0, v1)
\]
Where are we?

We can check if a pixel is in a triangle.
Problem: checking every pixels

Red = pixel tested outside of the triangle

Green = pixel tested inside of the triangle
Problem: checking every pixels

Small triangle:
Red = pixel tested outside of the triangle
Green = pixel tested inside of the triangle
To much inside/outside check!
Question

How can we reduce the amount of inside/outside check?

(1 minute alone)
(2 minutes with your neighbors)
(5 minutes with the whole group)
Answer: Axis Aligned Bounding Box

Only check pixel that are inside the AABBox of the triangle.
Answer: Axis Aligned Bounding Box

Only check pixel that are inside the AABBox of the triangle

Reduces the number of test on pixel outside of the triangle
An AABBox is defined by two points in space.

A : the bottom left corner of the box

\[
A = \begin{bmatrix}
\min(v0.x, v1.x, v2.x) \\
\min(v0.y, v1.y, v2.y)
\end{bmatrix}
\]

B : the top right corner of the box

\[
B = \begin{bmatrix}
\max(v0.x, v1.x, v2.x) \\
\max(v0.y, v1.y, v2.y)
\end{bmatrix}
\]
Drawing a Triangle

For each pixel $p$ in the AABBox:

If $p$ is inside the triangle:

Emit a fragment*
Fragments

Fragment: Candidate Pixel
Contains:
- Position of the pixel
- Depth of the fragment
- Interpolated vertices data*

*Future Lecture
Finding Depth:

How can we find p.z?
Barycentric coordinates

For every point $p$ in the triangle

$$p = \lambda_0 v_0 + \lambda_1 v_1 + \lambda_2 v_2$$

with

$$\lambda_0 + \lambda_1 + \lambda_2 = 1$$
Barycentric coordinates

\[ \lambda_0 = \frac{\text{Area}(p, v1, v2)}{\text{Area}(v0, v1, v2)} \]
The edge side function

$$edgeSide(p, v0, v1) = (p.x - v0.x)(v1.y - v0.y) - (p.y - v0.y)(v1.x - v0.x)$$

Signed Area

$$edgeSide(p, v0, v1) = Area(parallelograme) = 2 \times Area(p, v0, v1)$$
Barycentric coordinates

\[ \lambda_0 = \frac{\text{Area}(p, v1, v2)}{\text{Area}(v0, v1, v2)} = \frac{0.5 \times \text{edgeSide}(p, v1, v2)}{0.5 \times \text{edgeSide}(v0, v1, v2)} \]
Barycentric coordinates

\[ \lambda_0 = \frac{\text{Area}(p, v1, v2)}{\text{Area}(v0, v1, v2)} = \frac{0.5 \times \text{edgeSide}(p, v1, v2)}{0.5 \times \text{edgeSide}(v0, v1, v2)} \]
Finding \( p.z \)

\[
p.xy = \lambda_0 v0.xy + \lambda_1 v1.xy + \lambda_2 v2.xy
\]

In projected space \( Z \) is not linear

\[
p.z = \lambda_0 v0.z \mid \lambda_1 v1.z \mid \lambda_2 v2.z
\]
Finding $p.z$

$$p.xy = \lambda_0 v0.xy + \lambda_1 v1.xy + \lambda_2 v2.xy$$

$$\frac{1}{p.z} = \lambda_0 \frac{1}{v0.z} + \lambda_1 \frac{1}{v1.z} + \lambda_2 \frac{1}{v2.z_{31}}$$
Fragments

Fragment : Candidate Pixel
Contains :
- Position of the pixel ✓
- Depth of the fragment ✓
- Interpolated vertices data* ✓

*Future Lecture
Vertices & Camera
Indices
Light sources

Vertex Shader → Rasterizer → Fragment Shader → Depth Buffer → Image
Choose which Fragment get to become a pixel using a Depth test
Question:

How can we choose which fragment to draw?

(1 minute alone)
(2 minutes with your neighbors)
(5 minutes with the whole group)
Our use case:

Let’s draw the blue triangle first
The depth buffer

<table>
<thead>
<tr>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Array with the same size as the image

Initialized with 1.0
The depth buffer

1. Draw the first triangle
2. Emit fragment f
3. If \( f.z < \text{depthBuffer}[f.xy] \)
   - Draw f
4. Set \( \text{depthBuffer}[f.xy] = f.z \)
The depth buffer

Draw the first triangle

Emit fragment f

If \( f.z < \text{depthBuffer}[f.xy] \)

Draw f

\[ \text{depthBuffer}[f.xy] = f.z \]

\[ \text{triangle1.z} = 0.5 \]
The depth buffer

Our image
Our use case:

Now the red triangle
The depth buffer

Draw the second triangle

Emit fragment f

If \( f.z < \text{depthBuffer}[f.xy] \)

Draw f

\( \text{depthBuffer}[f.xy] = f.z \)
The depth buffer

Draw the second triangle

Emit fragment f

If f.z < depthBuffer[f.xy]
  Draw f
  depthBuffer[f.xy] = f.z

triangle2.z = 0.8
The depth buffer

Our image
Depth buffer

The depth Buffer allow us to draw triangle in a whichever order we want regardless of the configuration