Triangles
barycentric coordinates

\[ p = a + \beta(b - a) + \gamma(c - a) \]
barycentric coordinates

\[ p = a + \beta(b - a) + \gamma(c - a) \]

\[ p = (1 - \beta - \gamma)a + \beta b + \gamma c \]

\[ \alpha \equiv 1 - \beta - \gamma \]

\[ p(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c \]

\[ \alpha + \beta + \gamma = 1 \]
barycentric coordinates

If $p$ inside the triangle,

$$p(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c$$

$$0 < \alpha < 1$$
$$0 < \beta < 1$$
$$0 < \gamma < 1.$$
barycentric coordinates

If \( p \) on an edge, e.g.,

\[
p(\alpha, \beta, \gamma) = \alpha a + \beta b + \gamma c
\]

\[
\beta = 0
\]

\[
\alpha + \gamma = 1
\]
barycentric coordinates

\[ \mathbf{p} = \alpha \mathbf{a} + \beta \mathbf{b} + \gamma \mathbf{c} \]

What are \((\alpha, \beta, \gamma)\) ?

<whiteboard>
Triangle rasterization
Which pixels should be used to approximate a triangle?
Which pixels should be used to approximate a triangle?

Use Midpoint Algorithm for edges and fill in?
Which pixels should be used to approximate a triangle?

Use an approach based on barycentric coordinates
We can interpolate attributes using barycentric coordinates

\[ c = \alpha c_0 + \beta c_1 + \gamma c_2 \]

Gouraud shading

(Gouraud, 1971)

http://jtibble.dyndns.org/graphics/eecs487/eecs487.html
Triangle rasterization algorithm

for all $x$ do
  for all $y$ do
    compute $(\alpha, \beta, \gamma)$ for $(x,y)$
    if $(\alpha \in [0, 1] \text{ and } \beta \in [0, 1] \text{ and } \gamma \in [0, 1])$ then
      $c = \alpha c_0 + \beta c_1 + \gamma c_2$
      drawpixel$(x,y)$ with color $c$
Triangle rasterization algorithm

for all x do
    for all y do
        compute \((\alpha, \beta, \gamma)\) for \((x,y)\)
        if \((\alpha \in [0, 1] \text{ and } \beta \in [0, 1] \text{ and } \gamma \in [0, 1])\) then
            \[c = \alpha c_0 + \beta c_1 + \gamma c_2\]
            drawpixel\((x,y)\) with color \(c\)
Triangle rasterization algorithm

use a bounding rectangle

for x in \([x_{\text{min}}, x_{\text{max}}]\)
  for y in \([y_{\text{min}}, y_{\text{max}}]\)
    compute \((\alpha, \beta, \gamma)\) for \((x,y)\)
    if \((\alpha \in [0, 1] \text{ and } \beta \in [0, 1] \text{ and } \gamma \in [0, 1]\)) then
      \[c = \alpha c_0 + \beta c_1 + \gamma c_2\]
      drawpixel\((x,y)\) with color \(c\)
Triangle rasterization algorithm

for x in [x_min, x_max]
  for y in [y_min, y_max]
    \[\alpha = \frac{f_{bc}(x, y)}{f_{bc}(x_a, y_a)}\]
    \[\beta = \frac{f_{ca}(x, y)}{f_{ca}(x_b, y_b)}\]
    \[\gamma = \frac{f_{ab}(x, y)}{f_{ab}(x_c, y_c)}\]
    if (\(\alpha \in [0, 1]\) and \(\beta \in [0, 1]\) and \(\gamma \in [0, 1]\)) then
      \[c = \alpha c_0 + \beta c_1 + \gamma c_2\]
      drawpixel(x,y) with color c
Triangle rasterization algorithm

Optimizations?

for x in [x_min, x_max]
    for y in [y_min, y_max]
        \[ \alpha = \frac{f_{bc}(x, y)}{f_{bc}(x_a, y_a)} \]
        \[ \beta = \frac{f_{ca}(x, y)}{f_{ca}(x_b, y_b)} \]
        \[ \gamma = \frac{f_{ab}(x, y)}{f_{ab}(x_c, y_c)} \]
        if \((\alpha \in [0, 1] \text{ and } \beta \in [0, 1] \text{ and } \gamma \in [0, 1])\) then
            \[ c = \alpha c_0 + \beta c_1 + \gamma c_2 \]
            drawpixel(x,y) with color c
Triangle rasterization algorithm

Optimizations?

for \( x \) in \([x_{\text{min}}, x_{\text{max}}]\)
for \( y \) in \([y_{\text{min}}, y_{\text{max}}]\)
\[
\alpha = \frac{f_{bc}(x, y)}{f_{bc}(x_a, y_a)}
\]
\[
\beta = \frac{f_{ca}(x, y)}{f_{ca}(x_b, y_b)}
\]
\[
\gamma = \frac{f_{ab}(x, y)}{f_{ab}(x_c, y_c)}
\]
if \((\alpha \geq 0 \text{ and } \beta \geq 0 \text{ and } \gamma \geq 0)\) then
\[
c = \alpha c_0 + \beta c_1 + \gamma c_2
\]
drawpixel(x,y) with color \( c \)

make computation of bary. coords. incremental
color can also be computed incrementally
don’t need to check upper bound
Triangle rasterization issues
Who should fill in shared edge?
Who should fill in shared edge?
for x in \([x_{\text{min}}, x_{\text{max}}]\)
  for y in \([y_{\text{min}}, y_{\text{max}}]\)

\[
\begin{align*}
\alpha &= f_{bc}(x, y)/f_{bc}(x_a, y_a) \\
\beta &= f_{ac}(x, y)/f_{ac}(x_b, y_b) \\
\gamma &= f_{ab}(x, y)/f_{ab}(x_c, y_c)
\end{align*}
\]

if \((\alpha \geq 0 \text{ and } \beta \geq 0 \text{ and } \gamma \geq 0)\) then
  if \((\alpha > 0 \text{ or } f_{bc}(a) f_{bc}(r) > 0)\) and
    \((\beta > 0 \text{ or } f_{ca}(b) f_{ca}(r) > 0)\) and
    \((\gamma > 0 \text{ or } f_{ab}(c) f_{ab}(r) > 0)\)
  then
    \[
    \mathbf{c} = \alpha \mathbf{c}_0 + \beta \mathbf{c}_1 + \gamma \mathbf{c}_2
    \]
    drawpixel(x, y) with color \(\mathbf{c}\)
Graphics Pipeline (cont.)
Graphics Pipeline
Transform
“Modelview” Transformation

Object coordinates

Model

World coordinates

View

Eye coordinates
Project
Clip

Geometric Pipeline

Transform → Project → Clip

Pixel Pipeline

OpenGL application program → Pixel operations → Rasterizer → Frame buffer
Clip against view volume
Clipping against a plane

What’s the equation for the plane through $\mathbf{q}$ with normal $\mathbf{N}$?

$$f(p) = \mathbf{N} \cdot (p - q) = 0$$
Intersection of line and plane

How can we distinguish between these cases?
Intersection of line and plane

\[ f(a)f(b) \geq 0 \]

\[ f(a)f(b) < 0 \]
Intersection of line and plane

How can we find the intersection point?

<whiteboard>
Clip against view volume

\[ s = \frac{\mathbf{N} \cdot (\mathbf{q} - \mathbf{c})}{\mathbf{N} \cdot (\mathbf{b} - \mathbf{c})} \]

\[ t = \frac{\mathbf{N} \cdot (\mathbf{q} - \mathbf{a})}{\mathbf{N} \cdot (\mathbf{b} - \mathbf{a})} \]

need to generate new triangles
Hidden Surface Removal
Occlusion

“painter’s algorithm”
draw primitives in back-to-front order

[Wikipedia Commons]
Occlusion

“painter’s algorithm”
draw primitives in back-to-front order

problem:
triangle intersection
Occlusion

“painter’s algorithm”
draw primitives in back-to-front order

problem:
occlusion cycle
Use a **z-buffer** for hidden surface removal

test depth on a pixel by pixel basis

red drawn last
Use a *z-buffer* for hidden surface removal

at each pixel, record distance to the closest object that has been drawn in a *depth* buffer

without z-buffer  
with z-buffer
Use a \textit{z-buffer} for hidden surface removal

Figure 1. Block diagram of OpenGL.

without z-buffer \hspace{1cm} with z-buffer
Use a z-buffer for hidden surface removal

http://www.beyond3d.com/content/articles/41/
Backface culling: another way to eliminate hidden geometry
Hidden Surface Removal in OpenGL

```c
glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);

 glEnable(GL_DEPTH_TEST);

 glEnable(GL_CULL_FACE);
```

For a perspective transformation, there is more precision in the depth buffer for z-values closer to the near plane.