Lecture 2:

Digital Drawing

Computer Graphics and Imaging
UC Berkeley CS184/284
Today: Rasterizing Triangles into Pixels
Drawing Machines
Pablo Picasso: drawing machine?

https://www.youtube.com/watch?v=JU9oaD0e7uU
CNC Sharpie Drawing Machine

Aaron Panone with Matt W. Moore

Laser Cutters
Oscilloscope
Cathode Ray Tube
Oscilloscope Art

Jerobeam Fenderson
https://www.youtube.com/watch?v=rtR63-ecUNo
Television - Raster Display CRT

Cathode Ray Tube

Raster Scan Pattern of Interlaced Display

Raster Scan
(modulate intensity)
Frame Buffer: Memory for a Raster Display

Image = 2D array of colors
A Sampling of Different Raster Displays
Flat Panel Displays

Low-Res LCD Display
LCD (Liquid Crystal Display) Pixel

Principle: block or transmit light by twisting polarization

Illumination from backlight (e.g. fluorescent or LED)

Intermediate intensity levels by partial twist
LED Array Display

Light emitting diode array
DMD Projection Display

DIGITAL MICRO MIRROR DEVICE (DMD)
(SLM - Spatial Light Modulator)

MICRO MIRRORS CLOSE UP
DMD Projection Display

Array of micro-mirror pixels

DMD = Digital Micromirror Device
Electrophoretic (Electronic Ink) Display

Greenland or right-whale, he is the best existing authority. But Scoresby knew nothing and says nothing of the great sperm whale, compared with which the Greenland whale is almost unworthy mentioning. And here be it said, that the Greenland whale is an usurper upon the throne of the seas. He is not even by any means the largest of the whales. Yet, owing to the long priority of his claims, and the profound ignorance which, till some seventy years back, invested the then fabulous or utterly unknown sperm-whale, and which ignorance to this present day still reigns in all but some few scientific retreats and whale-ports; this usurpation has been every way complete. Reference to nearly all the leviathanic allusions in the great ports of past days, will satisfy you that the Greenland whale, without one rival, was to them the monarch of the seas. But the time has at last come for a new proclamation. This is Charing Cross; hear ye! good people all,—the Greenland whale is deposed,—the great sperm whale now reigneth!

There are only two books in being which at all pretend to put the living sperm whale before you, and at the same time, in the remotest degree succeed in the attempt. Those books are Beale's and Bennett's, both in their time surpases to English South Sea whale-ships, and both exact and reliable men. The original matter touching the sperm whale to be found in their volumes is necessarily small; but so far as it goes, it is of excellent quality, though
Drawing to Raster Displays
Polygon Meshes
Triangle Meshes
Triangle Meshes
Graphics Pipeline = Abstract Drawing Machine

- Vertices
- Transformed vertices
- Fragments
- Shaded fragments
- Pixels
- OpenGL commands
- Per-vertex ops
- Rasterizer
- Texturing
- Per-fragment ops
- Frame buffer ops
- Triangles, lines, points, images
- Pixels in the framebuffer

CS184/284A

Jonathan Ragan-Kelley & Ren Ng
Triangles - Fundamental Area Primitive

Why triangles?

• Most basic polygon
• Break up other polygons
• Optimize one implementation
• Triangles have unique properties
• Guaranteed to be planar
• Well-defined interior
• Well-defined method for interpolating values at vertices over triangle (barycentric interpolation)
Drawing a Triangle To The Framebuffer ("Rasterization")
What Pixel Values Approximate a Triangle?

Input: position of triangle vertices projected on screen

Output: set of pixel values approximating triangle

(2.2, 1.3) (4.4, 11.0) (15.3, 8.6)
Today, Let’s Start With A Simple Approach: Sampling
Sampling a Function

Evaluating a function at a point is sampling.

We can discretize a function by periodic sampling.

```
for( int x = 0; x < xmax; x++ )
  output[x] = f(x);
```

Sampling is a core idea in graphics. We’ll sample time (1D), area (2D), angle (2D), volume (3D) ...

We’ll sample N-dimensional functions, even infinite dimensional functions.
Let’s Try Rasterization As 2D Sampling
Sample If Each Pixel Center Is Inside Triangle
Sample If Each Pixel Center Is Inside Triangle
Define Binary Function: \textit{inside}(\texttt{tri}, x, y)

\[
\text{inside}(t,x,y) = \begin{cases} 
1 & (x,y) \text{ in triangle } t \\
0 & \text{otherwise} 
\end{cases}
\]
Rasterization = Sampling A 2D Indicator Function

\[
\text{for( int } x = 0; x < \text{xmax}; x++ ) \\
\quad \text{for( int } y = 0; y < \text{ymax}; y++ ) \\
\quad \quad \text{Image}[x][y] = f(x + 0.5, y + 0.5);
\]

Rasterize triangle \text{tri} by sampling the function \[ f(x,y) = \text{inside(tri,x,y)} \]
Implementation Detail: Sample Locations

Sample location for pixel \((x,y)\)

\((x+1/2, y+1/2)\)
Evaluating inside \((\text{tri},x,y)\)
Triangle = Intersection of Three Half Planes
Each Line Defines Two Half-Planes

Implicit line equation
• \( L(x,y) = Ax + By + C \)

• On line: \( L(x,y) = 0 \)
• Above line: \( L(x,y) > 0 \)
• Below line: \( L(x,y) < 0 \)
Line Equation Derivation

Line Tangent Vector

\[ T = P_1 - P_0 = (x_1 - x_0, y_1 - y_0) \]
Line Equation Derivation

General Perpendicular Vector in 2D

\[ \text{Perp}(x, y) = (-y, x) \]
Line Equation Derivation

$$N = \text{Perp}(T) = (- (y_1 - y_0), x_1 - x_0)$$
Line Equation Derivation

Let $P = (x, y)$ be a point on the line, and $P_0 = (x_0, y_0)$ be another point on the line. Then the vector $V = P - P_0 = (x - x_0, y - y_0)$. This vector $V$ is orthogonal to the normal vector $N$ of the line.
Line Equation

\[ L(x, y) = V \cdot N = -(x - x_0)(y_1 - y_0) + (y - y_0)(x_1 - x_0) \]
Line Equation Tests

\[ L(x, y) = V \cdot N > 0 \]
Line Equation Tests

\[ L(x, y) = V \cdot N = 0 \]
Line Equation Tests

\[ L(x, y) = V \cdot N < 0 \]
Point-in-Triangle Test: Three Line Tests

\( P_i = (X_i, Y_i) \)

\[ dX_i = X_{i+1} - X_i \]
\[ dY_i = Y_{i+1} - Y_i \]

\[ L_i(x, y) = -(x - X_i) \, dY_i + (y - Y_i) \, dX_i \]
\[ = A_i \, x + B_i \, y + C_i \]

\[ L_i(x, y) = 0 : \text{point on edge} \]
\[ < 0 : \text{outside edge} \]
\[ > 0 : \text{inside edge} \]

Compute line equations from pairs of vertices

Jonathan Ragan-Kelley & Ren Ng
Point-in-Triangle Test: Three Line Tests

\[ P_i = (X_i, Y_i) \]

\[ dX_i = X_{i+1} - X_i \]
\[ dY_i = Y_{i+1} - Y_i \]

\[ L_i(x, y) = -(x - X_i) \, dY_i + (y - Y_i) \, dX_i \]
\[ = A_i \, x + B_i \, y + C_i \]

\[ L_i(x, y) = 0 \text{ : point on edge} \]
\[ < 0 \text{ : outside edge} \]
\[ > 0 \text{ : inside edge} \]

\[ L_0(x, y) > 0 \]
Point-in-Triangle Test: Three Line Tests

\[ P_i = (X_i, Y_i) \]

\[ dX_i = X_{i+1} - X_i \]
\[ dY_i = Y_{i+1} - Y_i \]

\[ L_i(x, y) = -(x - X_i) dY_i + (y - Y_i) dX_i \]
\[ = A_i x + B_i y + C_i \]

\[ L_i(x, y) = 0 \text{ : point on edge} \]
\[ < 0 \text{ : outside edge} \]
\[ > 0 \text{ : inside edge} \]

\[ L_1(x, y) > 0 \]

Jonathan Ragan-Kelley & Ren Ng
Point-in-Triangle Test: Three Line Tests

\[ P_i = (X_i, Y_i) \]

\[ dX_i = X_{i+1} - X_i \]

\[ dY_i = Y_{i+1} - Y_i \]

\[ L_i(x, y) = -(x - X_i) dY_i + (y - Y_i) dX_i \]

\[ = A_i x + B_i y + C_i \]

\[ L_i(x, y) = 0 : \text{point on edge} \]
\[ < 0 : \text{outside edge} \]
\[ > 0 : \text{inside edge} \]

\[ L_2(x, y) > 0 \]
Point-in-Triangle Test: Three Line Tests

Sample point $s = (sx, sy)$ is inside the triangle if it is inside all three lines.

$$inside(sx, sy) = L_0(sx, sy) > 0 \& \& L_1(sx, sy) > 0 \& \& L_2(sx, sy) > 0;$$

Note: actual implementation of $inside(sx, sy)$ involves $\leq$ checks based on edge rules.
Edge Cases (Literally)

Is this sample point covered by triangle 1, triangle 2, or both?
OpenGL/Direct3D Edge Rules

When sample point falls on an edge, the sample is classified as within triangle if the edge is a “top edge” or “left edge”.

Top edge: horizontal edge that is above all other edges

Left edge: an edge that is not exactly horizontal and is on the left side of the triangle. (triangle can have one or two left edges)

Source: Direct3D Programming Guide, Microsoft
Incremental Triangle Traversal (Faster?)
Modern Approach: Tiled Triangle Traversal

Traverse triangle in blocks

Test all samples in block in parallel

Advantages:
- Simplicity of wide parallel execution overcomes cost of extra point-in-triangle tests (most triangles cover many samples, especially when super-sampling)
- Can skip sample testing work: entire block not in triangle (“early out”), entire block entirely within triangle (“early in”)

All modern GPUs have special-purpose hardware for efficient point-in-triangle tests
3 minute break
Signal Reconstruction on Real Displays
Real LCD/OLED Screen Pixels (Closeup)

Notice R,G,B pixel geometry! But in this class, we will assume a colored square full-color pixel.
Aside: What About Other Display Methods?

Color print: observe half-tone pattern
Assume Display Pixels Emit Square of Light

Each image sample sent to the display is converted into a little square of light of the appropriate color:
(a pixel = picture element)

* LCD pixels do not actually emit light in a square of uniform color, but this approximation suffices for our current discussion
So, If We Send The Display This Sampled Signal
The Display Physically Emits This Signal
Compare: The Continuous Triangle Function
What’s Wrong With This Picture?

Jaggies!
Jaggies (Staircase Pattern)

Is this the best we can do?
Discussion: What Value Should a Pixel Have?

Potential topics for your pair discussion:

• Ideas for “higher quality” pixel formula?
• What are all the relevant factors?
• What’s right/wrong about point sampling?
• Why do jaggies look “wrong”?
What does it mean for a pixel to be covered by a triangle?
Question: which triangles “cover” this pixel?
One option: compute fraction of pixel area covered by triangle, then color pixel according to this fraction.

Intuition: if triangle covers 10% of pixel, then pixel should be 10% red.
Analytical coverage schemes get tricky when considering occlusion of one triangle by another.

Pixel covered by triangle 1, other half covered by triangle 2.

Interpenetration of triangles: even trickier.

Two regions of triangle 1 contribute to pixel. One of these regions is not even convex.
Sampling: taking measurements a signal
Below: five measurements ("samples") of $f(x)$
How can we represent a sampled signal more accurately?

Sample signal more densely (increase sampling rate)
Point sampling: one sample per pixel
Supersampling: step 1
Take NxN samples in each pixel

(but... how do we use these samples to drive a display, since there are four times more samples than display pixels!)
Supersampling: step 2
Average the NxN samples “inside” each pixel

Averaging down
Supersampling: step 2

Average the N x N samples “inside” each pixel

Averaging down
Supersampling: step 2
Average the N x N samples “inside” each pixel
Supersampling: result

This is the corresponding signal emitted by the display
Point sampling

One sample per pixel
4x4 supersampling + downsampling

Pixel value is average of 4x4 samples per pixel
Next time: Antialiasing
Understanding what just happened in a more principled way
Things to Remember

Drawing machines

• Many possibilities
• Why framebuffers and raster displays?
• Why triangles?

We posed rasterization as a 2D sampling process

• Test a binary function inside \( \text{inside}(\text{triangle}, x, y) \)
• Evaluate triangle coverage by 3 point-in-edge tests
• Finite sampling rate causes “jaggies” artifact
  (next time we will analyze in more detail)
Acknowledgments

Thanks to Kayvon Fatahalian, Pat Hanrahan, Mark Pauly and Steve Marschner for slide resources.