**Lecture 7 of 41**

**Viewing 4 of 4: Culling and Clipping**

Lab 1b: Flash Intro

William H. Hsu
Department of Computing and Information Sciences, KSU

Public mirror web site: http://www.kddresearch.org/Courses/CIS636
Instructor home page: http://www.cis.ksu.edu/~bhsu

Readings:
Today: Sections 2.3.5, 2.4, 3.1.3, Eberly
Next class: Sections 2.4, 2.5, 3.1.6, Eberly
Brown CS123 slides on Clipping – http://bit.ly/eWU7i1

Lab 1b: Flash Intro

**Lecture Outline**

- Reading for Last Class: Sections 2.5.1, 3.1 Eberly 2
- Reading forToday: §2.3.5, 2.4, 3.1.3, Eberly 2
- Reading for Next Class: §2.4, 2.5 (Especially 2.5.4), 3.1.6, Eberly 2
- Last Time: Scan Conversion (aka Rasterization) of Lines
  - Incremental algorithm
  - Bresenham’s algorithm & midpoint line algorithm
- Preview: Circles and Ellipses (Lecture 8)
- Today: Intro to Clipping and Culling
  - Clipping
    - 2-D derivation: clip edges
      - Algorithms: Cohen-Sutherland, Liang-Barsky/Cyrus-Beck
    - 3-D derivation: clip faces
  - Culling
    - Back face culling
    - Occlusion culling

**Where We Are**

**Clipping**

- Adapted from slides © 1997 – 2010 van Dam et al., Brown University
- http://bit.ly/hiSt0f
- Reused with permission.

**Line Clipping**

- Adapted from slides © 1997 – 2010 van Dam et al., Brown University
- http://bit.ly/hiSt0f
- Reused with permission.

**Parametric Line Formulation For Clipping**

- Parametric form for line segment
  \[ X = x_0 + t(x_1 - x_0), \quad 0 \leq t \leq 1 \]
  \[ Y = y_0 + t(y_1 - y_0) \]
  \[ P(t) = P_0 + t(P_1 - P_0) \]

- "true" i.e., interior intersection, if \( x_0 \) and \( x_1 \) lie in \([0,1]\)
  (hard to compute)
Cohen-Sutherland 2-D Clipping: Outcodes [1]

- Divide plane into 8 regions
- Compute the sign of 4 comparison between a vertex and an edge
- Then use a 2-bit number to indicate each edge

Cohen-Sutherland Algorithm [1]

1. If we can neither trivially accept (T/A/T/A), divide and conquer
2. Subdivide line into two segments and then T/A/T/A one or both segments.

Scan Conversion after Clipping

- Don't round and then scan convert, because the line will lose the wrong edge
- Calculate decision variable based on gradient of left edge

Cohen-Sutherland 2-D Clipping: Outcodes [2]

- Very similar to 2D
- Divide volume into 27 regions (Picture & Rain's cube)
- 6-bit outcodes result in 6 bounds tests

Sutherland-Hodgman Polygon Clipping

- Use Sutherland-Hodgman clipping
- May use Cohen-Sutherland clipping for initial approximation
- Then use Sutherland-Hodgman clipping

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.
Cyrus-Beck / Liang-Barsky
Parametric Line Clipping [1]

- Use parametric line formulation: \( P(t) = P_0 + tP_1 - P_0P_1 \)
- Parametric line intersects infinite line formed by each clip rectangle edge
- Solve for multiple times depending on the number of clip edges crossed
- Decide which of these intersections actually occurs on the rectangle

\[
\begin{align*}
N_{1} \cdot (P_2 - P_1) &= 0 \\
N_{2} \cdot (P_3 - P_1) &= 0 \\
N_{3} \cdot (P_1 - P_4) &= 0 \\
N_{4} \cdot (P_0 - P_1) &= 0
\end{align*}
\]

For any point \( P_k \) on edge \( E_k \),

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.

Cyrus-Beck / Liang-Barsky
Parametric Line Clipping [2]

Given the values \( s \) and \( t \) at the intersection of \( P_0 \) and \( P_1 \) with the edges \( E_j \),

\[
N_j \cdot (P_1 - P_0) = 0
\]

- First, substitute for \( s(t) \):
  \[
  N_j \cdot (P_1 - P_0) = 0
  \]
- Next, group terms and distribute dot product:
  \[
  N_j \cdot (P_1 - P_0) = 0
  \]
- Let \( D \) be the vector from \( P_0 \) to \( P_1 \), and solve for \( t \):
  \[
  N_j \cdot (P_1 - P_0) = 0
  \]

Note that this gives a valid value of \( t \) only if the denominator of the expression is nonzero.

For this to be true, it must be the case that:
- \( \text{If } D \text{ is not parallel to the edge, then this could occur only as a minimum.} \)
- \( \Delta \) is parallel to \( E_j \), and \( D \) is not parallel to \( E_j \).
- The algorithm checks these conditions.

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.

Cyrus-Beck / Liang-Barsky
Parametric Line Clipping [3]

- Eliminate the same [1, 2, 3]
- Which remaining \( t \) produce interior intersection?
- Can’t just take the intersection values!

Move from \( P_3 \) to \( P_4 \) for a given edge, just before crossing:
- \( P_i := P_i - P_{i-1} \) (potentially shrinking this segment)
- \( P i(1) := P i(0) \) (potentially lengthening this segment)
- Pick inner PE, PL, pair t_i for \( P_i \) with \( t_i, t_i \), for \( P_0 \) with \( t_i, t_i \), and \( t_0, t_0 \, t_i, t_i \).
- \( 0 < t_i < 1 \) is some p1 point

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.

Cyrus-Beck / Liang-Barsky
Line Clipping Algorithm

Pre-calculate \( N_i \) and \( n_i \), for each edge:
- Select the segment to be clipped
- If \( n_i \), then \( i \) is degenerate or on one or parallel
- Otherwise:
  - \( i \) is not parallel to \( E_j \)
  - \( i \) is parallel to \( E_j \)
  - \( i \) is parallel to \( E_j \)
  - \( n_i \) is parallel to \( E_j \)

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.

Parametric Line Clipping
For Upright Clip Rectangle [1]

- \( D = P_3 - P_2 = (x_3 - x_2, y_3 - y_2) \)
- Leave \( P_2 \) as an arbitrary point on the clip edge

Calculations for Parametric Line Clipping Algorithm

\[
\begin{align*}
\text{Clip Edges, } & \text{ Intersect } \& \text{ PE, PL, } & \text{PE} & \text{PL} & \text{N} \times (\text{PE} - \text{PL}) \\
\text{Left} & \text{Segment} & (L) & (L) & \text{N} \times (\text{PL} - \text{PE}) \\
\text{Right} & \text{Segment} & (R) & (R) & \text{N} \times (\text{PE} - \text{PL}) \\
\text{Top} & \text{Segment} & (T) & (T) & \text{N} \times (\text{PE} - \text{PL}) \\
\text{Bottom} & \text{Segment} & (B) & (B) & \text{N} \times (\text{PL} - \text{PE}) \\
\end{align*}
\]

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.

Parametric Line Clipping
For Upright Clip Rectangle [2]

- Examine \( t \)
- \( t \) numerator is just the directed distance to an edge; sign corresponds to \( OC \)
- \( t \) denominator is the length of the projection of the line, \( dx \) or \( dy \)
- \( t \) sign determines PE or PL for a given edge
- \( t \) is constant of proportionality: "how far ever" from \( P_0 \) to \( P_1 \) intersection is relative to \( dx \) or \( dy \)

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
http://bit.ly/hiSt0f
Reused with permission.
Culling: A Form of Visible Surface Determination

- Given a set of 3-D objects and a view specification (camera), determine which lines or surfaces of the object are visible.
- Why might objects not be visible?
  - occlusion vs. clipping
  - clipping in one object at a time, while occlusion is global.
- Also called Hidden Surface Removal (HSR).
- We begin with some history of previously used VSD algorithms.

Visibility Culling: View Frustum, Back Face, Occlusion

- Clipping techniques:
  - Cohen-Sutherland: outcodes (quick rejection), test intersections
  - Liang-Barsky / Cyrus-Beck: solve for t, find innermost PE/PL

Occlusion Culling

- Adobe Flash
  - Basic 2-D (up to Flash v9)
  - 3-D: Flash 10+
  - Simple Flash animation exercise

Lab 1B

- Animation Ideas
  - Animate: to "bring to life"
  - From still frames to animations
  - Incremental change and smoothness

- Using Culling
  - Back faces illustrated
  - What to do besides cull

- Simple Flash Animation Exercise
  - Watch Semocula.com tutorial(s) as needed (http://bit.ly/hhlgtk)
  - Turn in
    - ActionScript source code
    - Screenshot(s) as instructed in Lab 1 handout

Summary

- Last Time: Scan Conversion (aka Rasterization)
  - Lines: incremental algorithm vs. (Bresenham’s) midpoint algorithm
  - Decision variables and forward differences
  - Circles and Ellipses (preview)

- See Also: CG Basics 3 - 4

- CG Basics 3: Projections and 3-D Viewing (in detail)
- CG Basics 4: Fixed-Function Graphics Pipeline

- Today: Clipping and Culling
  - What parts of scene to clip: edges vs. polygons of model
  - What parts of viewport to clip against: clip faces vs. clip edges

- Clipping techniques
  - Cohen-Sutherland: outcodes (quick rejection), test intersections
  - Liang-Barsky / Cyrus-Beck: solve for t, find innermost PE/PL

- Visibility culling: view frustum, back face, occlusion

Next: More Scan Conversion (Polygons, Scan Line Interpolation)