Computer Graphics

Z Sweedyk
Lecture 4
9/11/00

Outline

- Polygons
- Filled polygons

Polygon: \( p_1, p_2, p_3, p_4, p_5 \)

Order Matters

Polygon: \( p_1, p_3, p_5, p_2, p_4 \)

Polygon: Scan Conversion

\[
\text{Polygon}(p_1, \ldots, p_n) \\
\text{For } i = 1 \text{ to } n-1 \\
\text{DrawLine}(p_i, p_{i+1}) \\
\text{DrawLine}(p_{n}, p_1)
\]
Filled Polygon
Which pixels should be on?

Here we get the same size!

Tie Breaker 1: Entering Owns

Tie Breaker 2: Up wins
Exercise

Scan Line Algorithm

1. Compute intersections
2. Order by x-coordinate
3. Use odd-even test to turn on pixels

Odd-Even Test

Odd = in
Even = out

Odd-Even Test

May not be what you’re looking for …
but it’s easy to implement

Scan Line Algorithm

How to implement efficiently?

1. Compute intersections
2. Order by x-coordinate
3. Use odd-even test to turn on pixels

Key ideas

• Let S be the set of line segments that intersect scan line i. The set of lines that intersect scan line i+1 is:

  $S + \text{new-done}$

• Suppose that line $L = (m,b)$ intersects scan line i at $(x,i)$. If $L$ intersects scan line i+1 it does so at:

  $(x + 1/m, i+1)$
Data Structures

- Edge Table
- Active Edge Table

Example: Edge Table

<table>
<thead>
<tr>
<th>Yval</th>
<th>Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>L_2, L_3</td>
</tr>
<tr>
<td>3</td>
<td>L_4</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>L_0, L_3</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

Example: Active Edge Table

List of the Line segments intersecting current scan line

Example: Active Edge Table

Record with info about L_2

Line Segments “beginning” at scan line 3

Example: Active Edge Table

Record with info about L_0
Scan Line Algorithm

Build ET
Yval=1
Initialize AET=ϕ
Repeat until ET and AET are empty:
    Yval ++
    Update info on line segments
    Add ET[Yval] to AET
    Remove lines from AET that are "done"
Sort lines in AET by x-intercept at y=Yval
Choose pixels based on odd-even test

Line Record

Field 1: ymax

<table>
<thead>
<tr>
<th>line</th>
<th>ymax</th>
</tr>
</thead>
<tbody>
<tr>
<td>L_0</td>
<td>7</td>
</tr>
<tr>
<td>L_1</td>
<td>3</td>
</tr>
<tr>
<td>L_2</td>
<td>7</td>
</tr>
<tr>
<td>L_3</td>
<td>6</td>
</tr>
<tr>
<td>L_4</td>
<td>6</td>
</tr>
</tbody>
</table>

Scan Line Algorithm

Build ET
Yval=1
Initialize AET=ϕ
Repeat until ET and AET are empty:
    Yval ++
    Update info on line segments
    Add ET[Yval] to AET
    Remove lines from AET when Yval=ymax
Sort lines in AET by x-intercept at y=Yval
Choose pixels based on odd-even test

Line Record

Field 2. Xval

<table>
<thead>
<tr>
<th>line</th>
<th>Xval</th>
</tr>
</thead>
<tbody>
<tr>
<td>L_0</td>
<td>4/3</td>
</tr>
<tr>
<td>L_1</td>
<td>7/2</td>
</tr>
</tbody>
</table>

Scan Line Algorithm

Build ET
Yval=1
Initialize AET=ϕ
Repeat until ET and AET are empty:
    Yval ++
    Update info on line segments
    Add ET[Yval] to AET
    Remove lines from AET when Yval=ymax
Sort lines in AET by x-intercept at y=Yval
Choose pixels based on odd-even test

Line Record: Current Yval=2
Scan Line Algorithm

Build ET
Yval=1
Initialize AET=∅
Repeat until ET and AET are empty:
Yval ++

Update info on line segments
Add ET[Yval] to AET
Remove lines from AET when Yval = ymax

Sort lines in AET by x-intercept at y=Yval
Choose pixels based on odd-even test

Line Record: Current Yval=2

Field 3. 1/m

Line Record: Current Yval=2

Field 3. 1/m

Scan Line Algorithm

Build ET
Yval=1
Initialize AET=∅
Repeat until ET and AET are empty:
Yval ++

Increment xval by 1/m for each line in AET
Add ET[Yval] to AET
Remove lines from AET when Yval = ymax
Sort lines in AET by x-intercept at y=Yval
Choose pixels based on odd-even test

Initialize Line Records

<table>
<thead>
<tr>
<th>line</th>
<th>xval</th>
<th>1/m</th>
</tr>
</thead>
<tbody>
<tr>
<td>L_0</td>
<td>7/3</td>
<td>1/3</td>
</tr>
<tr>
<td>L_1</td>
<td>3/2</td>
<td>5/2</td>
</tr>
<tr>
<td>L_2</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>L_3</td>
<td>6</td>
<td>3/2</td>
</tr>
<tr>
<td>L_4</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>
Initialize Edge Table

<table>
<thead>
<tr>
<th>Yval</th>
<th>Lines</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>(L_2\rightarrow L_1)</td>
</tr>
<tr>
<td>3</td>
<td>(L_4)</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>(L_0\rightarrow L_3)</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

Initialize Active Edge Table

\[\varphi\]

Yval=0

\[\varphi\]

Yval=1

(sorted) AET

<table>
<thead>
<tr>
<th>Line</th>
<th>(y_{max})</th>
<th>(x)</th>
<th>(1/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(L_0)</td>
<td>7</td>
<td>1</td>
<td>1/3</td>
</tr>
<tr>
<td>(L_4)</td>
<td>3</td>
<td>1</td>
<td>5/2</td>
</tr>
</tbody>
</table>

Turn on \((i,1)\) where: \(1 \leq i < 7/2\)

Yval=2

(sorted) AET

<table>
<thead>
<tr>
<th>Line</th>
<th>(y_{max})</th>
<th>(x)</th>
<th>(1/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(L_0)</td>
<td>7</td>
<td>4/3</td>
<td>1/3</td>
</tr>
<tr>
<td>(L_3)</td>
<td>3</td>
<td>7/2</td>
<td>5/2</td>
</tr>
</tbody>
</table>

Turn on \((i,2)\) where: \(4/3 \leq i < 7/2\)

Yval=3

(sorted) AET

<table>
<thead>
<tr>
<th>Line</th>
<th>(y_{max})</th>
<th>(x)</th>
<th>(1/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(L_0)</td>
<td>7</td>
<td>5/3</td>
<td>1/3</td>
</tr>
<tr>
<td>(L_6)</td>
<td>6</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>

Turn on \((i,3)\) where: \(5/3 \leq i < 6\)
Claims

- AET always contains an even number of lines
- The algorithm implements the correct tie-breaking rules