Hidden Surface Elimination

Introduction

Need is to eliminate ambiguity
Hidden Surface Elimination

Introduction

Wire frame

Hidden Line Elimination

Hidden Surface Elimination
Hidden Surface Elimination

Introduction

Approaches

• Image Space
  Through pixel

• Object Space
  Through primitive
Hidden Surface Elimination

Image Space Approach

for (each pixel in the image)
{
    determine the object closest to the viewer that is intercepted by the projector through the pixel;

    draw the pixel in the appropriate color;
}

Computational effort: \( np \)

\( n \) : number of objects

\( p \) : number of pixels
Hidden Surface Elimination

Object Space Approach

for (each object in the world)
{
    determine those parts of the object whose view is unobstructed by other parts of it or any other object;

    draw those parts in the appropriate color;

}

Computational effort: \( n^2 \)

\( n \): number of objects
Floating Horizon Algorithm

Surface Function

\[ F(x, y, z) = 0 \]
Floating Horizon Algorithm

With \( z = \text{constant} \) plane closest to the viewpoint, the curve in each plane is generated (for each \( x \) coordinate in image space the appropriate \( y \) value is found).

Projection on \( z = 0 \) plane

**Algorithm:**
If at any given value of \( x \) the \( y \) value of the curve in the current plane is larger than the \( y \) value for any previous curve at that \( x \) value, then the curve is visible, otherwise it is hidden.
Hidden Surface Elimination

Floating Horizon Algorithm

Algorithm:
If at any given value of $x$ the $y$ value of the curve in the current plane is larger than the $y$ value or smaller than the minimum $y$ value for any previous curve at that $x$ value, then the curve is visible, otherwise it is hidden.

Projection on $z=0$ plane
Back Face Culling

Preprocessing to eliminate faces which are not visible

If a surface’s normal is pointing away from the eye (viewer), then this is a back face

$$\text{If } n_p \cdot V < 0 \text{ then backface}$$
Hidden Surface Elimination

Back Face Culling

Conservative algorithm
Hidden Surface Elimination

Back Face Culling

Conservative algorithm
Hidden Surface Elimination

Z-Buffer Algorithm

Proposed by Catmull 1974
Simple to implement
Z-buffer is like a frame buffer, contains depth

\[(x, y)\]
\[Z_b(x, y)\]
\[C(x, y)\]
Initialize all $d[i,j]=1.0$ (max depth), $c[i,j]=$background color.
for (each polygon)
    for (each pixel in polygon’s projection)
    {
        Find depth-z of polygon at $(x,y)$ corresponding to pixel $(i,j)$;
        If $z < d[i,j]$
        $d[i,j] = z$;
        $p[i,j] = color$;
        end
    }
Z-Buffer Algorithm

Computationally

\[ Ax + By + Cz + D = 0 \]
\[ \text{At } x \]
\[ z = \frac{-(Ax + By + D)}{C}, C \neq 0 \]
\[ \text{At } x + \Delta x \]
\[ z_1 = \frac{-(A(x + \Delta x) + By + D)}{C} \]
\[ z_1 - z = \frac{-A}{C} \Delta x \]
\[ z_1 = z - \frac{A}{C} \Delta x = z - \frac{A}{C} \quad (\because \Delta x = 1) \]
Hidden Surface Elimination

Z-Buffer Algorithm

Example

Z-buffer

Screen
Hidden Surface Elimination

Z-Buffer Algorithm

Example

Z-buffer
Hidden Surface Elimination

Z-Buffer Algorithm

Example

\[ [0,6,7] \]

\[ [0,1,2] \]

\[ [5,1,7] \]
Hidden Surface Elimination

Z-Buffer Algorithm

Example

Z-buffer

Screen
Hidden Surface Elimination

Z-Buffer Algorithm

- Simple method
- Complexity
  - Time: \( nxm \) buffer \( k \) polygons
    \( O(nmk) \)
  - Space: \( b \) depth precision
    \( O(nmb) \)
Hidden Surface Elimination

Painter’s Algorithm

Depth Sort, List Priority
Polygons are painted to the screen in the order of their distance from the viewer (More distant objects are painted first)

Screen display
Hidden Surface Elimination

Painter’s Algorithm

• Sort polygons in order of increasing depth
• Draw polygons in sequence, starting from the polygon (surface) of greatest depth
• Careful processing of depth
• Efficiency depends on sorting algorithm
Hidden Surface Elimination

Painter’s Algorithm

Z=20

Z=15

Z=10
Hidden Surface Elimination

Painter’s Algorithm

Draw first P then Q and then R

Draw first P then Q
Hidden Surface Elimination

Painter’s Algorithm

Cyclic overlapping
May require splitting
Binary Space Partitioning (BSP)

Space partitioning using planes

BSP Tree
Binary Space Partitioning (BSP)

For hidden surface elimination: sets a display order
Binary Space Partitioning (BSP)

For hidden surface elimination: sets a display order

Display order (back to front): BPF

BSP Tree
Binary Space Partitioning

Example
Binary Space Partitioning

Example
Hidden Surface Elimination

Binary Space Partitioning

Example
Binary Space Partitioning

Example

Display order: 5a 2 1 3 5b 4
Hidden Surface Elimination

Binary Space Partitioning

Example
Binary Space Partitioning

Example

Display order: 5 2 1 3 4
Binary Space Partitioning

Issues

How to select the root polygon?

Criteria: Number of split (fragmentation)

a. Arbitrary
b. Heuristic based with 4-5 polygons and consider which gives the least number of split

Static vs Dynamic scene
Hidden Surface Elimination

Area Subdivision

Quad tree
Hidden Surface Elimination

Area Subdivision Algorithm

Warnock’s Algorithm

Surrounding  Intersecting  Contained  Disjoint

Polygon

Area of interest
Consider an area of the projected image.

If it is easy to decide which polygons are visible in the area, display.
Else the area is subdivided into smaller areas and the decision is made recursively.

Divide and Conquer.
No Subdivision is required if

1. All the polygons are disjoint: **background color in the area.**

2. Only one intersecting or only one contained polygon: **The area is filled first by background color, then the polygon part contained in the area.**

3. Only one surrounding polygon (no contained and intersecting polygons): **The area is filled with the color of the surrounding polygon.**

4. More than one polygon is intersecting, contained in, or surrounding the area, with surrounding polygon in front: **Fill the area with the color of the surrounding polygon.**
Hidden Surface Elimination

Warnock’s Algorithm

Area
Hidden Surface Elimination

Warnock’s Algorithm
Hidden Surface Elimination

Warnock’s Algorithm
Hidden Surface Elimination

Warnock’s Algorithm
Hidden Surface Elimination

Warnock’s Algorithm

Maximum subdivision: *pixel*
Hidden Surface Elimination

Weiler Atherton Algorithm

Subdivision along polygon boundaries

Clipping!
Weiler Atherton Algorithm

- Initial z-sort
- Consider front most polygon, clip all polygons to generate Fragments inside polygon and outside polygon
- All inside polygons if behind delete from the list
- If there is an inside polygon in front (offending), clip the inside list of polygons against this polygon
- Process the outside polygon(s)
Hidden Surface Elimination

Weiler Atherton Algorithm

A as clip polygon
Inside list: A, \( B_{\text{in}A} \)
Outside list: \( B_{\text{out}A} \)

Display A
Process \( B_{\text{out}A} \)
Hidden Surface Elimination

Weiler Atherton Algorithm
Weiler Atherton Algorithm

A as clip polygon

- Inside list: A, $B_{in}A$
- Outside list: $B_{out}A$

$B_{in}A$ as clip polygon

- Inside list: $B_{in}A$, $A_{in}B$
- Outside list: $B_{out}A$, $A_{out}B$

Display $B_{in}A$
Hidden Surface Elimination

Weiler Atherton Algorithm

More polygons with more fragments
Hidden Surface Elimination

Scan Line Algorithm

Based on

Scan-line coherence across multiple scan-lines or span-coherence!

– scan-conversion algorithm and a little more data structure
Hidden Surface Elimination

Scan Line Algorithm

Spans
Scan Line Algorithm

Overview

- Each scan line is subdivided into several "spans"
- Determine which polygon the current span belongs to
- Shade the span using the current polygon’s color
- Exploit "span coherence" :
- For each span, a visibility test may need to be done
Hidden Surface Elimination

Scan Line Algorithm

inside span (2 polygon)

outside span

inside span (1 polygon)
Hidden Surface Elimination

Scan Line Algorithm

Inside/Outside Spans

- A scan line is subdivided into a sequence of spans
- Each span can be "inside" or "outside" polygon areas
  - "outside": no pixels need to be drawn (background color)
  - "inside": can be inside one or multiple polygons
- If a span is inside one polygon, the pixels in the span will be drawn with the color of that polygon
- If a span is inside more than one polygon, then compare the z values of those polygons at the scan line edge intersection point to determine the color of the pixel
Hidden Surface Elimination

Scan Line Algorithm

Inside/Outside Spans

- When a scan line intersects an edge of a polygon
  - for the 1\textsuperscript{st} time, the span becomes "inside" of the polygon from that intersection point on
  - for a 2\textsuperscript{nd} time, the span becomes "outside" of the polygon from that point on
- A flag "in/out" for each polygon is used to keep track of the current state
- Initially, the in/out flag is set to be "outside" (value = 0 for example). Invert the tag for “inside”. 
Hidden Surface Elimination

Scan Line Algorithm

- Each polygon will have its own in/out flag
- There can be more than one polygon having the in/out flags to be "in" at a given instance
- Keep track of polygons the scan line is currently in
- If there are more than one polygon "in", perform z value comparison to determine the color of the scan line span
Hidden Surface Elimination

Scan Line Algorithm

Data Structure

Edge Table (ET)
Polygon Table (PT)

<table>
<thead>
<tr>
<th>ET</th>
<th>x</th>
<th>y_{max}</th>
<th>Δx</th>
<th>poly-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>PT</td>
<td>poly-ID</td>
<td>A,B,C,D</td>
<td>color</td>
<td>in/out flag</td>
</tr>
</tbody>
</table>

xy

max

Δx

poly-ID

poly-ID

A,B,C,D

color

in/out flag
Hidden Surface Elimination

Scan Line Algorithm

In addition,
Use active In-Polygon List (IPL)
Active Edge Table (AET)
Hidden Surface Elimination

Scan Line Algorithm

Example
# Hidden Surface Elimination

## Scan Line Algorithm

### Example

<table>
<thead>
<tr>
<th>Y</th>
<th>AET</th>
<th>IPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>$x_0$, ba, bc, $x_N$</td>
<td>BG, BG+S, BG</td>
</tr>
<tr>
<td>II</td>
<td>$x_0$, ba, bc, 32, 13, $x_N$</td>
<td>BG, BG+S, BG, BG+T, BG</td>
</tr>
<tr>
<td>III</td>
<td>$x_0$, ba, 32, ca, 13, $x_N$</td>
<td>BG, BG+S, BG+S+T, BG+T, BG</td>
</tr>
<tr>
<td>IV</td>
<td>$x_0$, ba, ac, 12, 13, $x_N$</td>
<td>BG, BG+S, BG, BG+T, BG</td>
</tr>
</tbody>
</table>
Hidden Surface Elimination

Scan Line Algorithm

Example

<table>
<thead>
<tr>
<th>Y</th>
<th>AET</th>
<th>IPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>$x_0, ba, bc, x_N$</td>
<td>BG, BG+S, BG</td>
</tr>
</tbody>
</table>
Hidden Surface Elimination

Scan Line Algorithm

Example

<table>
<thead>
<tr>
<th>Y</th>
<th>AET</th>
<th>IPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>II</td>
<td>$x_0, ba, bc, 32, 13, x_N$</td>
<td>BG, BG+S, BG, BG+T, BG+T+IPL</td>
</tr>
</tbody>
</table>
Hidden Surface Elimination

Scan Line Algorithm

Example

\[ x_0, \text{ba}, 32, \text{ca}, 13, x_N \]

BG, BG+S, BG+S+T, BG+T, BG
Hidden Surface Elimination

Scan Line Algorithm

Example

<table>
<thead>
<tr>
<th>Y</th>
<th>AET</th>
<th>IPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>IV</td>
<td>( x_0, \ ba, \ ac, \ 12, \ 13, \ x_N )</td>
<td>( \text{BG, BG+S, BG, BG+T, BG} )</td>
</tr>
</tbody>
</table>
Hidden Surface Elimination

Scan Line Algorithm

Non-penetrating
Hidden Surface Elimination

Scan Line Algorithm

Penetrating

I $x_0$, ba, 23, ec, ad, 13, $x_N$ BG, BG+S, BG+S+T,

______________BG+S+T+, BG+T, BG