Clipping & Visibility

Luiz Velho
IMPA

Main Viewing Operations

Project / Raster

- **Clipping**
- **Rasterisation**
- **Visibility**

Ray Casting

Clipping

Clipping

Overview

- Usage
  - Visualization
  - Others
- Geometry
  - Line
  - Polygon
  - Patches

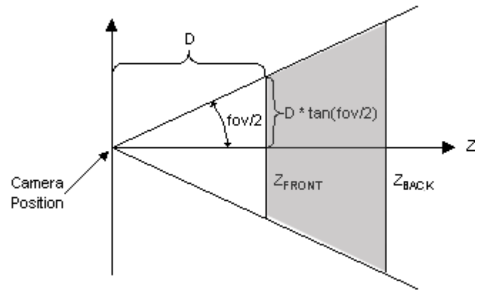
Analysis of the Problem

- Space Partition
  - Local
  - Global
- Separability

Clipping for Visualization

- Clipping Volume (Pyramid)
  - Normalized
  - Convex

  - Reasons
    - Avoid Projection Errors
    - Efficiency
  - Strategies
    - Eliminate Simple Cases
    - Canonical Situation
  - Coordinate System
    - Euclidean Space
    - Projective Space



Clipping Types

- **Culling**

- **Analytic** (Exact)

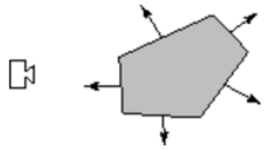
- **Subdivision** (Approximate)

Culling

Techniques

Backface Elimination

- Discard Polygons Facing Away from Camera
- Dot Product with Viewing Direction



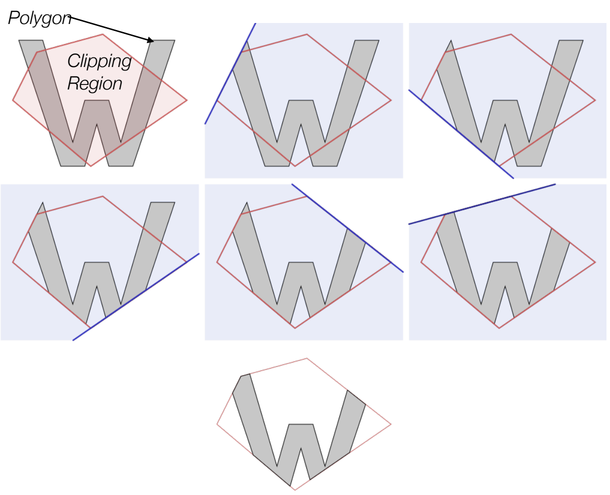
Elimination of Trivial Cases

- Entire Polygon (or Bounding Box) Outside Frustrum

Analytic Clipping

- Plane Schedule
  - Entire Polygon
  - Each Plane
- Polygon Types
  - Concave / Convex
  - Planar / Non Planar
- Clipping Region
  - Convex
- Pipeline
- Storage

Hardware Implementation



Sutherland-Hodgman Algorithm

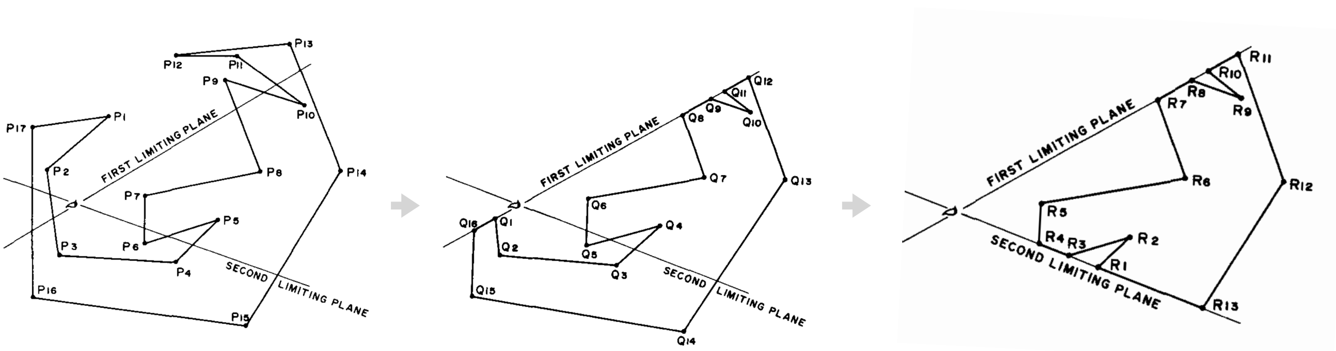
```
#define PLANE_CROSS(D0, D1) ((D0) * (D1) < 0.0)

#define ON_POSITIVE_SIDE(D1) ((D1) >= 0.0)

int pclip(int plane, Hpoly *s, Hpoly *d)
{
  int i, k0=0, k1;
  double d0=0, d1;

  for (d->n = k1 = i = 0; i <= s->n ; i++,  k1 = (i == s->n)? 0 : i) {
    d1 = plane_point_dist(plane, s->v[k1]);
    if (i != 0) {
      if (PLANE_CROSS(d0, d1))
        d->v[d->n++] = v4_add(s->v[k1], v4_scale(d1/(d1-d0), v4_sub(s->v[k0], s->v[k1])));
      if (ON_POSITIVE_SIDE(d1))
        d->v[d->n++] = s->v[k1];
    }
    d0 = d1;
    k0 = k1;
  }
  return (plane++ == LAST_P)? d->n : pclip(plane, d, s, dd);
}
```

Sutherland-Hodgman Clipping



Rasterisation

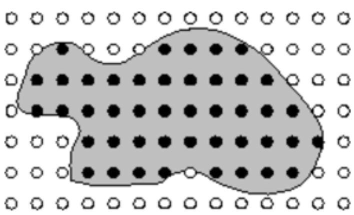
Rasterisation

Description of the Problem

Scan Conversion

Enumerate all image elements corresponding to a graphical object, and sample its attributes

- Enumeration
- Sampling



Visibility

Visibility & Visualization

Analysis

- Main Problem: Ordering
  - Rasterization: *XY*
  - Visibility: *Z*
- Characterisitics
  - Partial (need only first)
  - Random / Nearly Sorted
- Operations
  - Sorting
  - Searching
  - Culling
  - Merging

Classification

- YXZ
  - Z-Buffer
  - Scanline
- (XY)Z
  - Ray Casting
  - Screen Subdivision
- Z(XY)
  - Z-Sort
  - Recursive Clipping

Visibility & Geometry

Scene Geometry

- Object Types
  - Polygons
  - Parametric Patches
  - Implicit Algebraic Surfaces
  - Procedural
- Cases
  - (triangles)
  - General

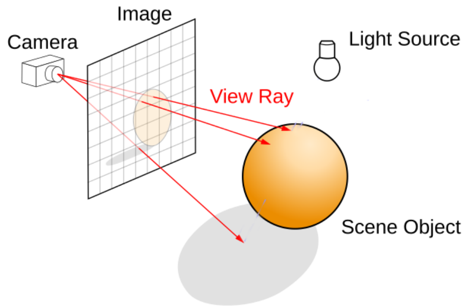
Coordinate Systems

- Image Space
  - Ordering at Each Ray
  - (Visibility Last)
- Object Space
  - Pre-Compute Order of Pieces
  - (Visibility First)

Visibility Code

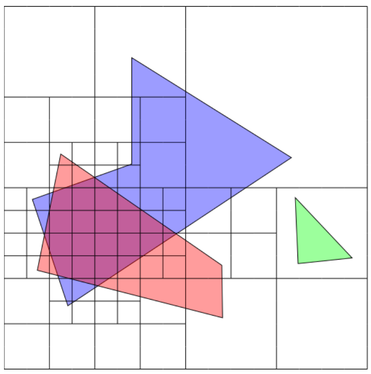
Ray Tracing

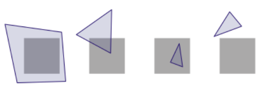
```
for(each pixel (sample) on the viewing area)
{
    for(each primitive in the world model)
    {
        if(ray-pixel intersection)
        {
            select the frontmost intersection;
           calculate color;
        }
    }
}
```



Screen Subdivision Visible Surface

- Divide & Conquer Algorithm
- Recursive
  - If (*simple*)
    draw
    else
    subdivide & recurse
- Subdivision Criteria
  - Complex Visibility Configuration
- When is simple ?
  - *No more than one polygon in view*





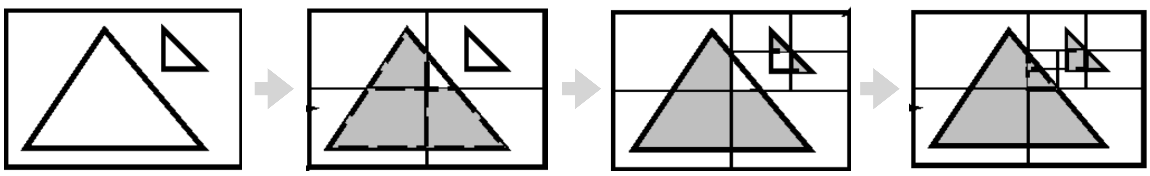
Warnock Algorithm

- Quad-Tree Recursion

```
Warnock(PolyList PL, ViewPort VP)
If ( PL simple in VP) then
    Draw PL in VP
else
    Split VP vertically and horizontally into VP1,VP2,VP3,VP4
    Warnock(PL in VP1, VP1)
    Warnock(PL in VP2, VP2)
    Warnock(PL in VP3, VP3)
    Warnock(PL in VP4, VP4)
end
```

Warnock Algorithm

- In Action

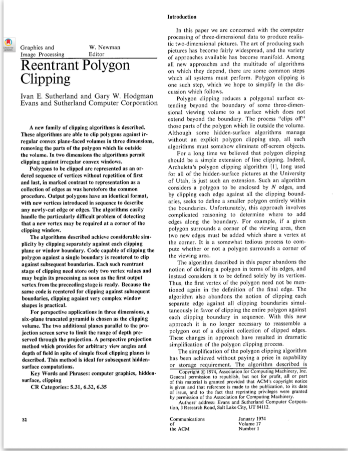


- Runtime: $O(p \times n)$
  - p: number of pixels
  - n: number of polygons

Clipping & Visibility

A Bit of History

- From Polygon Clipping to Visible Surface Computation



Graphics and Image Processing — W. Newman, Editor

Reentrant Polygon Clipping

Ivan E. Sutherland and Gary W. Hodgman
Evans and Sutherland Computer Corporation

A new family of clipping algorithms is described. These algorithms are able to clip polygons against irregular convex plane-faced volumes in three dimensions, removing the parts of the polygon which lie outside the volume. In two dimensions the algorithms permit clipping against irregular convex windows.

Polygons to be clipped are represented as an ordered sequence of vertices without repetition of first and last, in marked contrast to representation as a collection of edges as was heretofore the common procedure. Output polygons have an identical format, with new vertices introduced in sequence to describe any newly-cut edge or edges. The algorithms easily handle the particularly difficult problem of detecting that a new vertex may be required at a corner of the clipping window.

The algorithms described achieve considerable simplicity by clipping separately against each clipping plane or window boundary. Code capable of clipping the polygon against a single boundary is reentered to clip against subsequent boundaries. Each such reentrant stage of clipping need store only two vertex values and may begin its processing as soon as the first output vertex from the preceeding stage is ready. Because the same code is reentered for clipping against subsequent boundaries, clipping against very complex window shapes is practical.

For perspective applications in three dimensions, a six-plane truncated pyramid is chosen as the clipping volume. The two additional planes parallel to the projection screen serve to limit the range of depth preserved through the projection. A perspective projection method which provides for arbitrary view angles and depth of field in spite of simple fixed clipping planes is described. This method is ideal for subsequent hidden-surface computations.

Key Words and Phrases: computer graphics, hidden-surface, clipping
CR Categories: 5.31, 6.32, 6.35

Introduction

In this paper we are concerned with the computer processing of three-dimensional data to produce realistic two-dimensional pictures. The art of producing such pictures has become fairly widespread, and the variety of approaches available has become manifold. Among all new approaches and the multitude of algorithms on which they depend, there are some common steps which all systems must perform. Polygon clipping is one such step, which we hope to simplify in the discussion which follows.

Polygon clipping reduces a polygonal surface extending beyond the boundary of some three-dimensional viewing volume to a surface which does not extend beyond the boundary. The process "clips off" those parts of the polygon which lie outside the volume. Although some hidden-surface algorithms manage without an explicit polygon clipping step, all such algorithms must somehow eliminate off-screen objects.

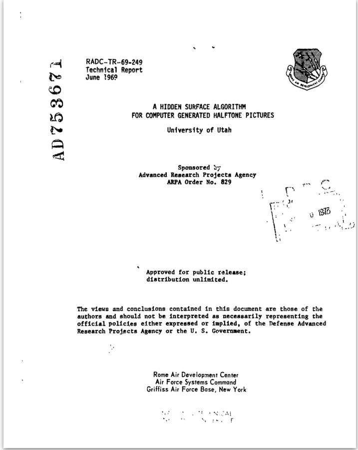
For a long time we believed that polygon clipping should be a simple extension of line clipping. Indeed, Archuleta's polygon clipping algorithm [1], long used for all of the hidden-surface pictures at the University of Utah, is just such an extension. Such an algorithm considers a polygon to be enclosed by N edges, and by clipping each edge against all the clipping boundaries, seeks to define a smaller polygon entirely within the boundaries. Unfortunately, this approach involves complicated reasoning to determine where to add edges along the boundary. For example, if a given polygon surrounds a corner of the viewing area, then two new edges must be added which share a vertex at the corner. It is a somewhat tedious process to compute whether or not a polygon surrounds a corner of the viewing area.

The algorithm described in this paper abandons the notion of defining a polygon in terms of its edges, and instead considers it to be defined solely by its vertices. Thus, the first vertex of the polygon need not be mentioned again in the definition of the final edge. The algorithm also abandons the notion of clipping each separate edge against all clipping boundaries simultaneously in favor of clipping the entire polygon against each clipping boundary in sequence. With this new approach it is no longer necessary to reassemble a polygon out of a disjoint collection of clipped edges. These changes in approach have resulted in dramatic simplification of the polygon clipping process.

The simplification of the polygon clipping algorithm has been achieved without paying a price in capability or storage requirement. The algorithm described is

Authors' address: Evans and Sutherland Computer Corporation, 3 Research Road, Salt Lake City, UT 84112.

Reentrant Polygon Clipping



AD753671

RADC-TR-69-249
Technical Report
June 1969

A HIDDEN SURFACE ALGORITHM
FOR COMPUTER GENERATED HALFTONE PICTURES

University of Utah

Sponsored by
Advanced Research Projects Agency
ARPA Order No. 829

Approved for public release;
distribution unlimited.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies either expressed or implied, of the Defense Advanced Research Projects Agency or the U. S. Government.

Rome Air Development Center
Air Force Systems Command
Griffiss Air Force Base, New York

Screen Subdivision Visibility

Algorithmic Principles

- Sutherland-Hodgman
  - Structure
- Warnock
  - Divide & Conquer
- Both
  - Recursion