Visibility

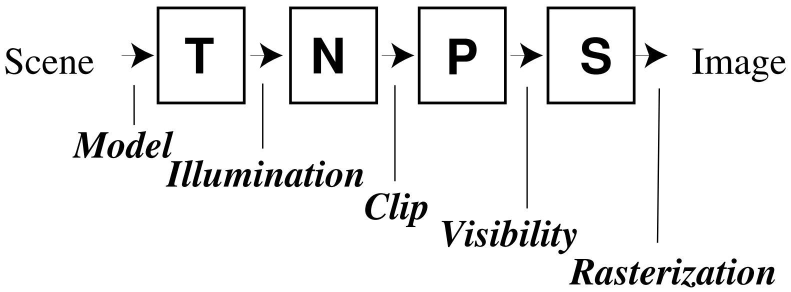
- Rasterization
- Visible Surface Calculation

Analysis

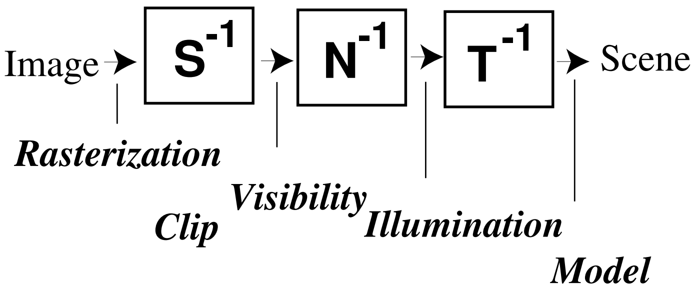
- Main Problem: Ordering
  - Rasterization: XY
  - Visibility: Z
- Characterisitics
  - Partial (1st)
  - Random / Nearly Sorted
- Operations
  - Sorting
  - Searching
  - Culling
  - Merging
- Properties
  - Exponential Complexity
  - Data Structures

Visibility

- Approaches
  - Object Centered



  - Image Centered



- Order of Operations

Coordinate Systems

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

Object Centered Methods

- Algorithm

  for each object

  enumerate all pixels

  solve conflicts

- Sequence of Operations
  - Transform
  - Clip
  - Rasterize
  - Visible Surface Calc

Image Centered Methods

- Algorithm

  for each pixel

  find all objects

  solve conflicts

- Sequence of Operations
  - Rasterization + Cliping
  - Transform
  - Visible Surface Calc

Geometry

- Object Types
  - Polygons
  - Parametric Patches
  - Algebraic Surfaces
  - Procedural
- Cases
  - Special
  - General

Scene Properties

– Number of Objects
– Object Sizes
– Depth Complexity
– Isotropy
– Inter-Penetration
– Translucency
– Clustering

Strategy

- Divide and Conquer
- Coherence
  - Frame
  - Object
  - Face
  - Edge
  - Image
  - Depth
- Complexity vs. Scale

Classification

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

Ray Casting

for each pixel

  intersect ray with objects

  find closest intersection

  paint pixel

- (Optimization)
- (Shading Rays)
- (CSG)
- (Stochastic Sampling)

Scanline Z-Buffer

```
Sort Polygon Edges

for each scanline
    init Z-buffer
    for each active polygon
       determine span
       for each pixel in span
         if z(polygon) ¡ z(buffer)
         paint pixel
```

- (Search over $\{z_i(x,y)\}$ smallest z_i)
- (A-Buffer)

Z Sort

Sort Polygons by Z

while poly list not empty

    get P and Q

    if P not obscure Q
        continue

    else if Q is marqued
        Split and Reinsert

    else if Q not obscure P
        Mark Q

        Exchange Q

Paint In order

- Test
(Bbox / Half Space / Screen Overlap)
- Split
(plane / half)

Space Subdivision

- Build BSP-Tree
- Traverse BSP-Tree

Screen Subdivision

```
Warnock(list, rect)
    for each poly in list
        if poly in rect
            classify poly
        else
            discard poly
    if simple
        paint rect
    else
        divide rect in 4
        Warnock(list,Q1)
        Warnock(list,Q2)
        Warnock(list,Q3)
        Warnock(list,Q4)
```

Recursive Clipping

```
RClip(list)
    if list empty
        return
    Sort in Z
    Clip list with first = inside, outside
    while inside not empty
        get next
        if first in front next
            remove next
        else
            exchange first with next
    Paint first
    RClip(outside)
```