Unit -3
Transformation, Clipping, Projections and Solid Modeling

Learning Objective

Transformations

2-D Viewing Transformation, Computing location of Viewport, Window-to viewport transformation

Clipping

Line Clipping; Sutherland Cohen clipping algorithms, Sutherland-Hodgeman

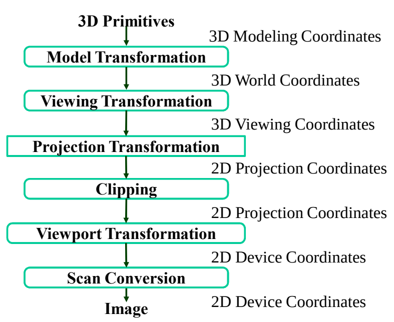
Projection

Parallel and Perspective Projections

Solid Modeling

Sweeping a polygon or a surface patch along a path to form solids, Boundary Representation (B-Rep), octrees, CSG – Constructive Solid Geometry.

Viewing Pipeline



Two-Dimensional Viewing

2-D viewing is the formal mechanism for displaying views of a picture on an output device. Terms related to 2-D viewing are :

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Geometric
  - Modeling
  - Viewing
- Structures/ Segments/ Objects

Two-Dimensional Viewing

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Geometric
  - Modeling
  - Viewing
- Structures/ Segments/ Objects

Co-ordinate Representations

- Modeling coordinates
- World coordinates
- Viewing coordinates
- Normalized viewing coordinates
- Device coordinates

(Xmc,Ymc)→(Xwc,Ywc)→(Xvc,Yvc)→(Xnvc,Ynvc)→(Xdc,Ydc)

Two-Dimensional Viewing

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Modeling
  - Geometric
  - Viewing
- Structures/ Segments/ Objects

Two-Dimensional Viewing

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Modeling
  - Geometric
  - Viewing
- Structures/ Segments/ Objects

Two-Dimensional Viewing

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Modeling
  - Geometric
  - Viewing
- Structures/ Segments/ Objects

Two-Dimensional Viewing

- Co-ordinate representations
- Output primitive
- Attributes
- Transformations
  - Modeling
  - Geometric
  - Viewing
- Structures/ Segments/ Objects

Window-to-Viewport transformation

- **Window**
  - World coordinate area selected for display
  - It defines what is to be displayed
- **Viewport**
  - An area on a display device to which a window is mapped
  - It defines where it is to be displayed

Window-to-Viewport transformation

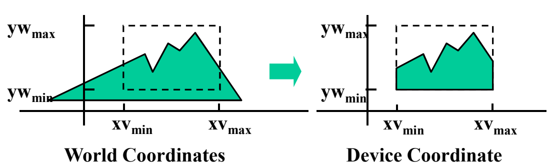
- Transform 2D Geometric Primitives from Scene Coordinate System (World Coordinates) to Image Coordinate System (Device Coordinates)
- When we are interested in displaying certain portions of the drawing, *windowing* technique is used
- *Viewing transformation* is the mapping of a part of a world-coordinate scene to device coordinates.
- 2D (two dimensional) viewing transformation sometimes referred to as the *window-to-viewport transformation* or the *windowing transformation*.

Viewport Transformation Cont..

- Mapping a window onto a viewport involves converting from one coordinate system to another.
- Zooming effects can be achieved by successively mapping different-sized windows on a fixed-size viewport.
- Panning effects are produced by moving a fixed-size window across various sized viewport.

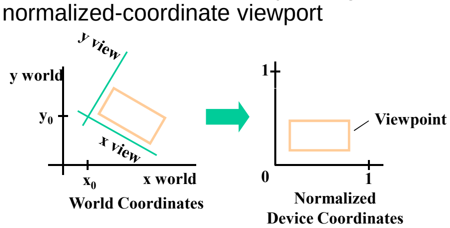
Window-to-Viewport transformation

- The mapping of a part of world co-ordinate scene to device coordinates is referred to as viewing transformation.
- A viewing transformation uses standard rectangles for the window and viewport



Window-to-Viewport transformation

- Setting up a rotated world window in viewing coordinates and the corresponding normalized-coordinate viewport



Viewing Coordinate Reference Frame

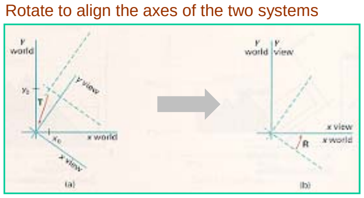
- Used to provide a method for setting up arbitrary orientations for rectangular windows
- Matrix for converting world-coordinate positions to viewing coordinate

$$\mathbf{M}_{WC,VC} = \mathbf{R} \cdot \mathbf{T}$$

R: rotation matrix
T: translation matrix

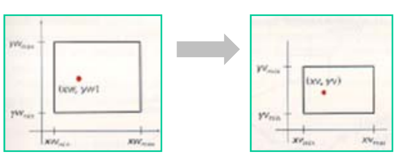
Viewing Coordinate Reference Frame

- The steps in this coordinate transformation
  - A viewing coordinate frame is moved to coincidence with the world frame in two steps
    a) Translate the viewing origin to the world origin, then
    b) Rotate to align the axes of the two systems



Window-to-Viewport transformation cont..

- Window-to-viewport mapping
  - A point at position (xw, yw) in a designated window is mapped to viewport coordinates (xv, yv) so that relative positions in the two areas are the same



Window-to-Viewport transformation cont..

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}}$$

Window-to-Viewport transformation cont..

From these two equations we derived

$xv = xv_{min} + (xw - xw_{min})sx$

$yv = yv_{min} + (yw - yw_{min})sy$

where the scaling factors are

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}} \qquad sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

Window-to-Viewport transformation cont..

Thus, the conversion of window area into viewport area is performed with the following sequence of transformations:

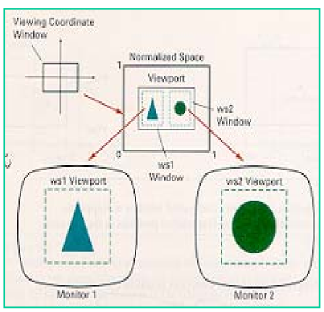
1. Perform a scaling that scales the window area to the size of the viewport.
2. Translate the scaled window area to the position of the viewport.
3. Relative proportions are maintained if the scaling factors are same (sx=sy)

Window-to-Viewport transformation cont..

- The way of character string mapping
  - Maintaining character size
    - ✓Using standard character fonts
  - Changing character size
    - ✓Using characters formed with line segments
- Workstation transformation
  - Opening any number of output devices in a particular application
  - Performing another window-to-viewport transformation for each open output device

Window-to-Viewport transformation cont..

- Mapping selected parts of a scene in normalized coordinates to different video monitors with Workstation transformation

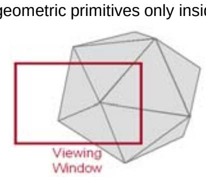


Clipping

•Any procedure that identifies those portions of a picture that are either inside or outside of a specified region of space is called **clipping**

•The region against which an object is clipped is called a **clip window** or **viewing window**

•Avoid Drawing Parts of Primitives Outside Window
- Window defines part of scene being viewed
- Must draw geometric primitives only inside window



Clipping

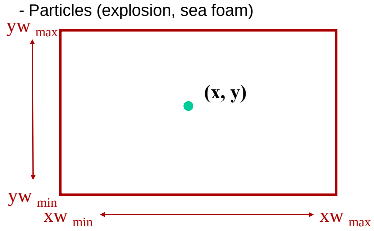
- Clipping algorithms can be considered for following primitive types
  - Point
  - Line
  - Area (polygons)

Point Clipping

- Assuming that the clip window is a rectangle in standard position
- Saving a point P=(x, y) for display

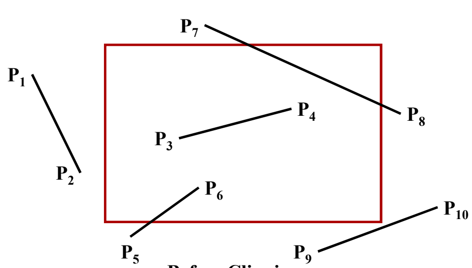
Inside =
$xw_{min} <= x <= xw_{max}$
$yw_{min} <= y <= yw_{max}$

- Appling Fields
  - Particles (explosion, sea foam)



Line Clipping

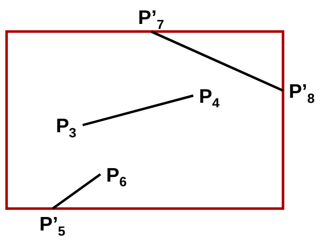
- Find the Part of a Line Inside the Clip Window



Before Clipping

Line Clipping

- Find the Part of a Line Inside the Clip Window



After Clipping

Cohen-Sutherland Line Clipping

Line Clipping

- Split plane into 9 regions
- Assign each a 4-bit tag(left, right, below, above)
- Assign each endpoint a tag

1001	1000	1010
0001	0000	0010
0101	0100	0110

clip rectangle

Cohen-Sutherland Line Clipping

Sutherland Cohen clipping algorithms

Checking for trivial acceptance or rejection using outcodes

1. Each endpoint of a line segment is assigned an outcode;
2. If both 4-bit codes are zero, the line can be accepted;
3. A logical **and** is performed on both outcodes;
4. If the result is nonzero, the line can be rejected.
5. Else If not accepted or rejected, divide the line segment into two at a clip edge
6. Iteratively clipped by test acceptance or rejection, and divided into two segments until completely inside or rejection.

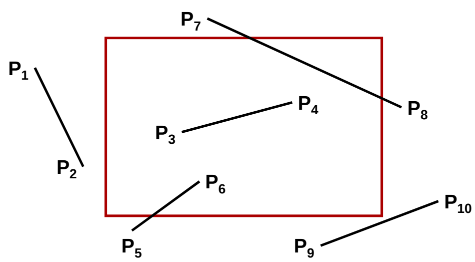
Cohen-Sutherland Line Clipping

Which Line is accepted or rejected or clipped

- G = 1001, H = 1000
- I = 0001, J = 1000
- A = 0000, B = 0000
- C = 0000, B = 0010
- E = 0100, F = 0100

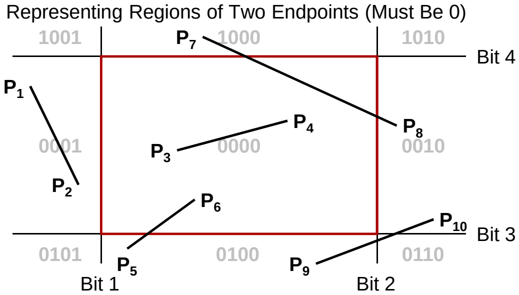
Cohen-Sutherland Line Clipping

- Use Simple Tests to Classify Easy Cases First



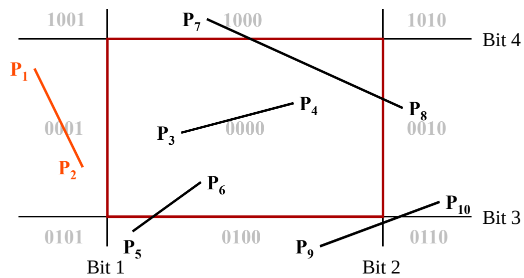
Cohen-Sutherland Line Clipping

- Classify Some Lines Quickly by AND of Bit Codes Representing Regions of Two Endpoints (Must Be 0)



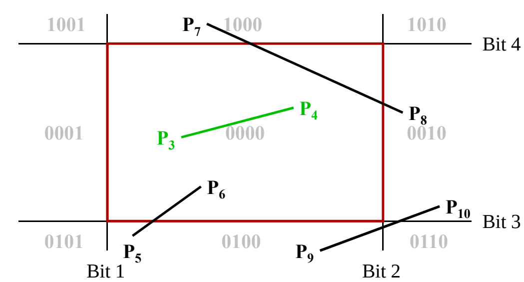
Cohen-Sutherland Line Clipping

- Classify Some Lines Quickly by AND of Bit Codes Representing Regions of Two Endpoints (Must Be 0)



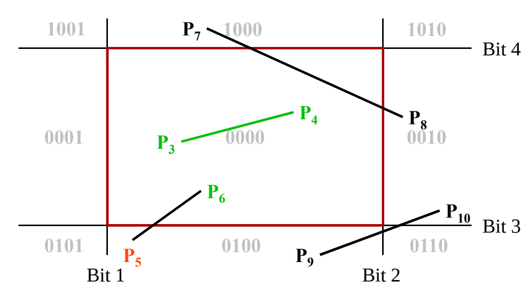
Cohen-Sutherland Line Clipping

- Classify Some Lines Quickly by AND of Bit Codes Representing Regions of Two Endpoints (Must Be 0)



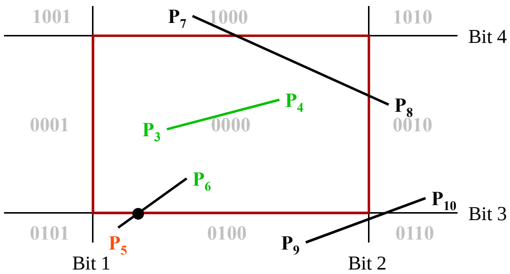
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



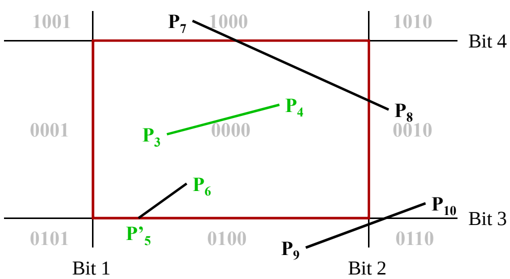
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



Cohen-Sutherland Line Clipping

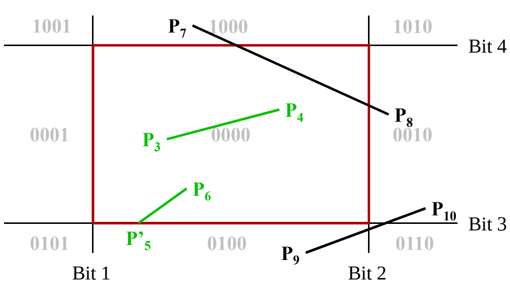
- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

Cohen-Sutherland Line Clipping

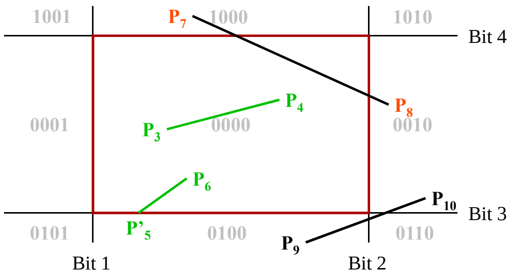
- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

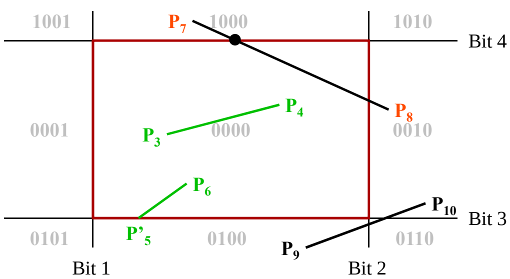
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



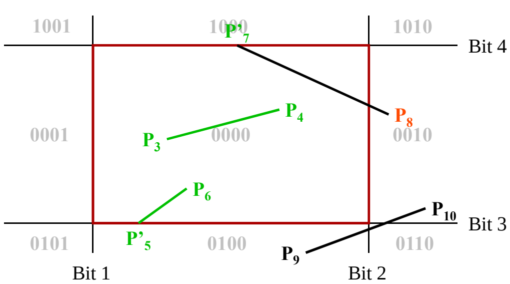
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



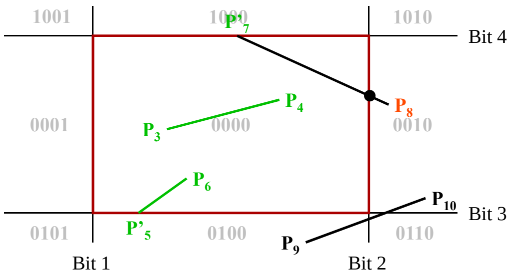
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



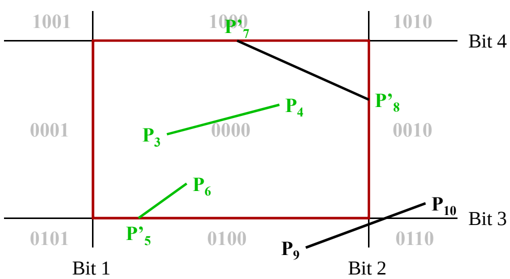
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



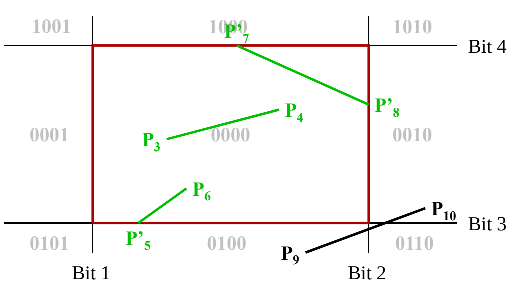
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



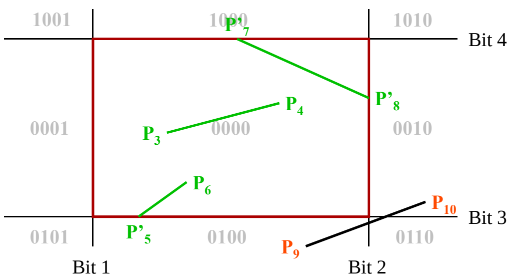
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



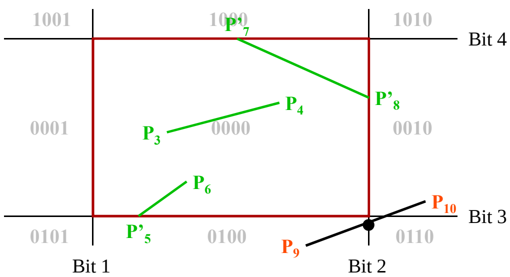
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



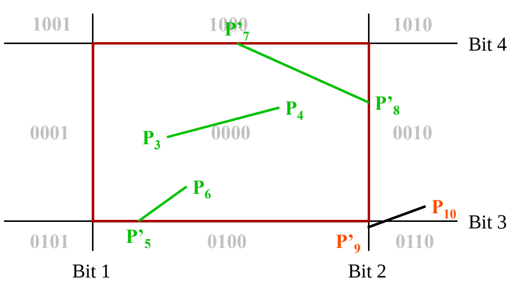
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



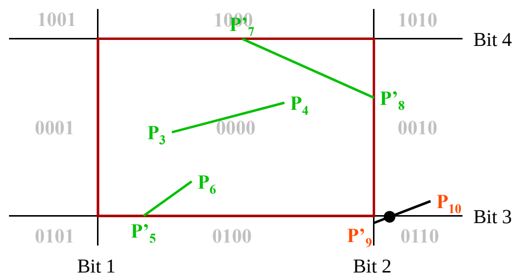
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



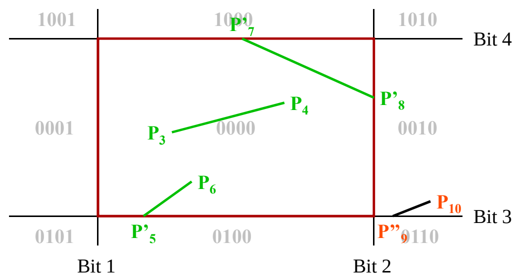
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



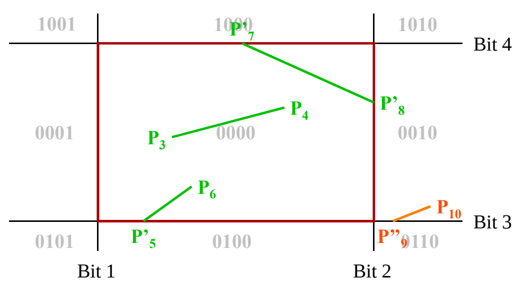
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



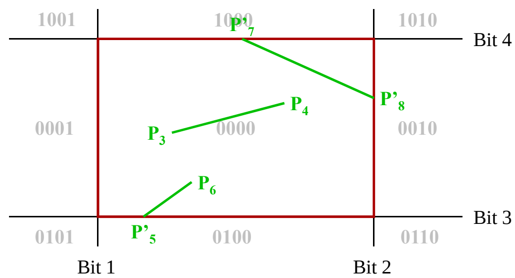
Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



Cohen-Sutherland Line Clipping

- Compute Intersections with Window Boundary for Lines That Can't be Classified Quickly



Cohen-Sutherland Line Clipping

Intersection points with a clipping boundary :

$$x = x_1 + \frac{(y - y_1)}{m}$$

$$y = y_1 + m(x - x_1)$$

Where x = either xw_{min} or xw_{max}

y = either yw_{min} or yw_{max}

$m = (y_2 - y_1) / (x_2 - x_1)$

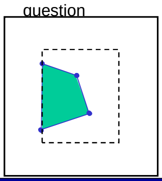
Polygon Clipping

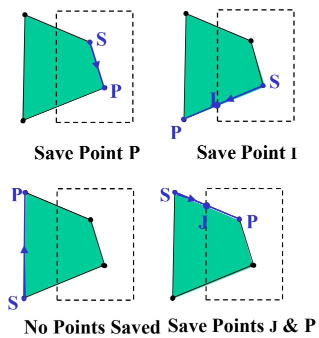
Sutherland-Hodgman Clipping Algorithm

- To clip an area against an individual boundary:
  - Consider each vertex in turn against the boundary
  - Vertices inside the boundary are saved for clipping against the next boundary
  - Vertices outside the boundary are clipped
  - If we proceed from a point inside the boundary to one outside, the intersection of the line with the boundary is saved
  - If we cross from the outside to the inside intersection point and the vertex are saved

Polygon Clipping

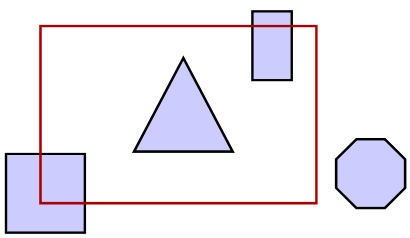
- Each example shows the point being processed (P) and the previous point (S)
- Saved points define area clipped to the boundary in question





Polygon Clipping

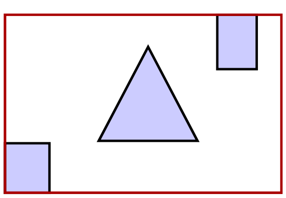
- Find the Part of a Polygon Inside the Clip Window?



Before Clipping

Polygon Clipping

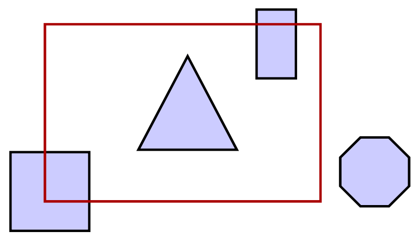
- Find the Part of a Polygon Inside the Clip Window?



After Clipping

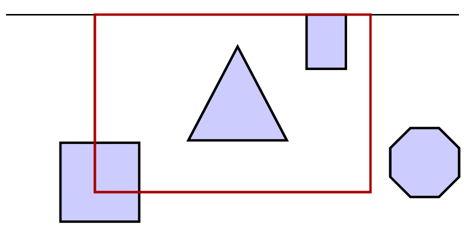
Sutherland-Hodgeman Polygon Clipping

- Clip to Each Window Boundary One at a Time



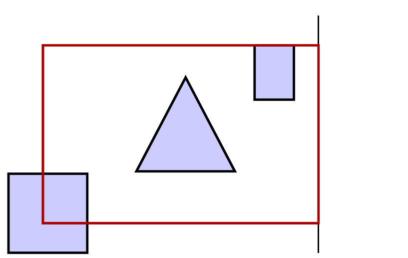
Sutherland-Hodgeman Polygon Clipping

- Clip to Each Window Boundary One at a Time



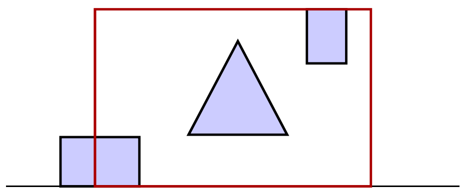
Sutherland-Hodgeman Polygon Clipping

- Clip to Each Window Boundary One at a Time



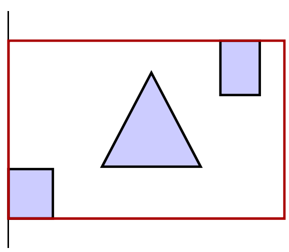
Sutherland-Hodgeman Polygon Clipping

- Clip to Each Window Boundary One at a Time



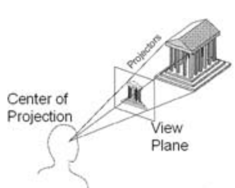
Sutherland-Hodgeman Polygon Clipping

- Clip to Each Window Boundary One at a Time

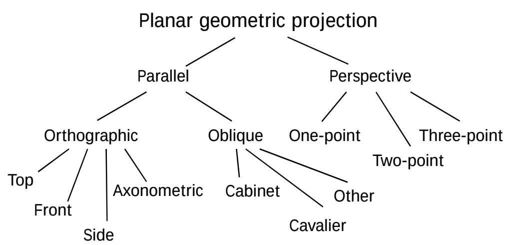


Projections

- General definition
  - Transform points in n-space to m-space(m<n)
- In computer graphics
  - Map viewing coordinates to 2D screen coordinates

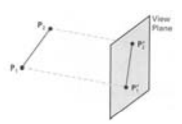


Taxonomy of Projections

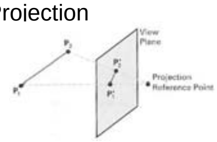


Projections

- Parallel Projection



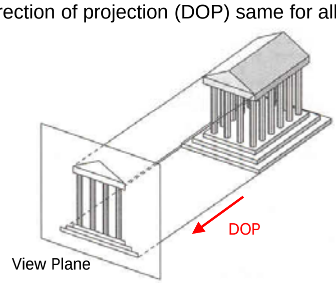
- Perspective Projection



© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

Parallel Projection

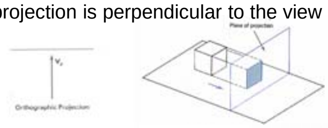
- Center of projection is at infinity
  - Direction of projection (DOP) same for all points



© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

Orthographic and Oblique

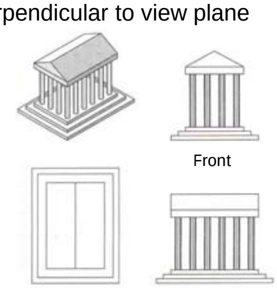
- Orthographic parallel projection
  - the projection is perpendicular to the view plane



- Oblique parallel projection
  - The projectors are inclined with respect to the view plane

Orthographic Projections

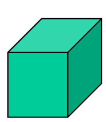
- DOP perpendicular to view plane



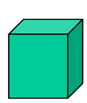
© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

Oblique Projections

- DOP not perpendicular to view plane



Cavalier
(DOP at 45°)

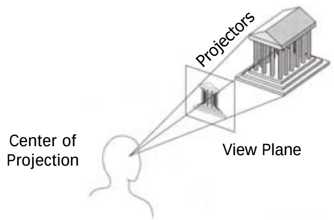


Cabinet
(DOP at 63.4°)

© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Vaishali Joshi, Assistant Professor U3. ‹#›

Perspective Projection

- Map points onto “view plane” along “projectors” emanating from “center of projection”(cop)



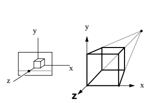
Perspective Projection

- Used for:
  - Human visual system…
- Pros:
  - gives a realistic view and feeling for 3D form of object
- Cons:
  - does not preserve shape of object or scale
- Different from a parallel projection because
  - parallel lines not parallel to the projection plane converge
  - size of object is diminished with distance
  - foreshortening is not uniform

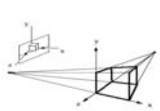
Perspective Projection

What is vanishing point?

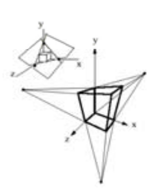
- A vanishing point is where parallel lines that move off into the distance seem to converge.
- In computer graphics, lines extending from edges converge to common vanishing point(s)



One Point Perspective
(*z*-axis vanishing point)



Two Point Perspective
(*z*, and *x*-axis vanishing points)



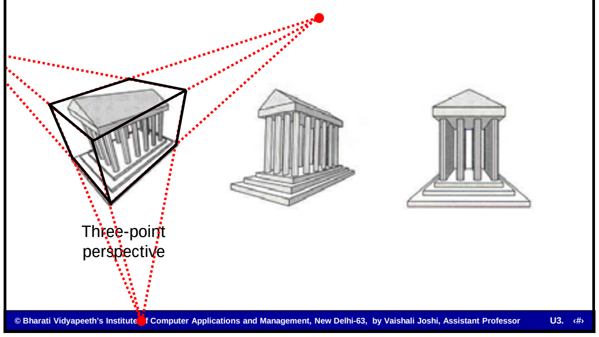
Three Point Perspective
(*z*, *x*, and *y*-axis vanishing points)

Perspective Projection

- How many vanishing point?

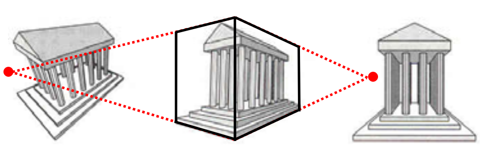
Perspective Projection

- How many vanishing point?



Perspective Projection

- How many vanishing point?

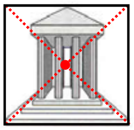


Perspective Projection

- How many vanishing point?

Three-point perspective

Two-point perspective



One-point perspective

Perspective vs. Parallel

- Perspective projection
  + Size varies inversely with distance – looks realistic
  – Distance and angles are not(in general) preserved
  – Parallel line do not (in general) remain parallel
- Parallel projection
  + Good for exact measurements
  + Parallel lines remain parallel
  – Angles are not (in general) preserved
  – Less realistic looking

Solid Modeling

- It is a mathematical technique for representing solid objects in terms of basic output primitives and splines.
- Representation of solids can be done in following ways :
  - Sweep representations
  - Boundary representations (B-Reps)
  - Space-partitioning representations (Octrees, BSP trees)
  - **C**onstructive **S**olid **G**eometry (CSG)

Solid Modeling

Sweep Representations

- Sweep representations are useful for constructing 3-D objects that posses translational or rotational symmetries.
- A large class of shapes can be formed by *sweeping* a 2D shape through space.
- Sweeps can be :
  - Translational
  - Rotational

Solid Modeling Cont..

Boundary Representation (B-Rep)

- A solid is modeled as a set of surfaces forming its boundary.
- These surfaces seperate object interior from environment.
- Example : Polygon facets

Topology and Geometry

1. Topology – how the surfaces are connected together.

2. Geometry – where the surfaces actually are in space.

3. Topology and Geometry are strongly linked.

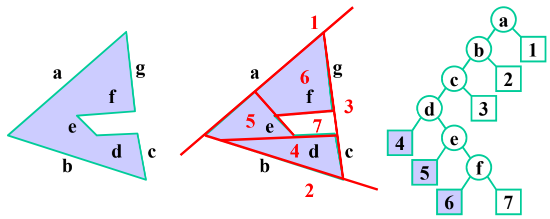
Solid Modeling Cont..

- The most commonly used boundary representation for a 3-D graphics object is a set of surface polygons that enclose the object interior.
- All surfaces are described with linear equations.
- There are different ways for describing a Polygon Surface :
  - Polygon tables (vertex, edge and polygon-surface table)
  - Plane Equation
  - Polygon meshes (triangle strip, quadrilateral mesh)

Solid Modeling Cont..

Binary Space Partitions (BSPs)

- Recursive Partition of Space by Planes
  - Mark leaf cells as inside or outside object



Object **Binary Spatial Partition** **BSP Tree**

Solid Modeling Cont..

Quadtree and Octrees:

- A *quadtree* is a tree in which each node has at most 4 children.
- An *octree* is a tree in which each node has at most 8 children.

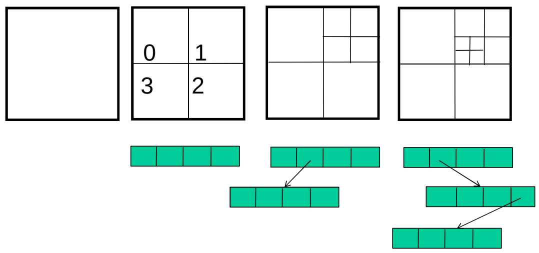
Solid Modeling Cont..

In practice, however, we use "quadtree" and "octree" to mean something more specific:

- Each node of the tree corresponds to a square (quadtree) or cubical (octree) region.
- If a node has children, think of its region being chopped into 4 (quadtree) or 8 (octree) equal subregions. Child nodes correspond to these smaller subregions of their parent's region.
- Subdivide as little or as much as is necessary.
- Each internal node has exactly 4 (quadtree) or 8 (octree) children.

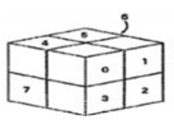
Solid Modeling Cont..

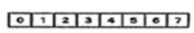
Quadtree Representation



Solid Modeling Cont..

Octrees Representations





Region of a 3D space

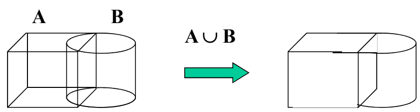
Data element in the Representative Octree Node

Solid Modeling Cont..

- Constructive Solid Geometry (CSG)
- Represent solid object as hierarchy of boolean operations
  - Union
  - Difference
  - Intersection

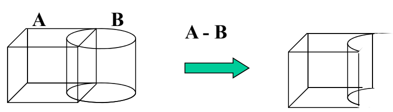
Solid Modeling Cont..

- Union
  - The sum of all points in each of two defined sets. (logical "OR")
  - Also referred to as Add, Combine, Join, Merge



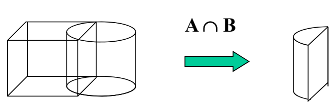
Solid Modeling Cont..

- Difference
  - – logical "NOT" Representation
  - – Also referred to as subtraction, remove, cut



Solid Modeling Cont..

- intersection
  - – Those points common to each of two defined sets (logical "AND")
  - – Set must share common volume
  - – Also referred to as common, conjoin



Conclusion

In this unit we understand the various features like Vanishing point projection, Viewing and window to viewport transformation and various concepts of clipping and Solid modeling as well.

Review Questions cont..

Short answer type Questions

1. Define Vanishing Point.
2. What do you mean by aspect ratio? Give its significance.
3. Define a viewport.
4. Explain with the help of an example Window-to-viewport transformation.
5. What do you mean by Quad tree? Explain with an example.
6. Define Clipping with reference to Computer Graphics.

Review Questions cont..

Long answer type Questions

1. What do you mean by Solid modeling? Explain various aspects of Solid modeling with example.
2. Explain Sutherland-Hodgman Clipping Algorithm? With the help of an example.
3. Explain Sutherland Cohen clipping algorithms? With the help of an example.
4. Differentiate between parallel and perspective projection.
5. Differentiate between Quad tree and Octree methods?

Suggested Reading/References

[1]. Donnald Hearn and M. Pauline Baker, “Computer Graphics”, PHI.

[2]. Foley James D, “Computer Graphics”, AW 2nd Ed.

[3]. Rogers, “Procedural Element of Computer Graphics”, McGraw Hill.

[4]. Newman and Sproul, “Principal of to Interactive Computer Graphics”, McGraw Hill.

[5]. Graphics lab @ Korea University.

