Two-Dimensional Viewing

COMPUTER GRAPHICS

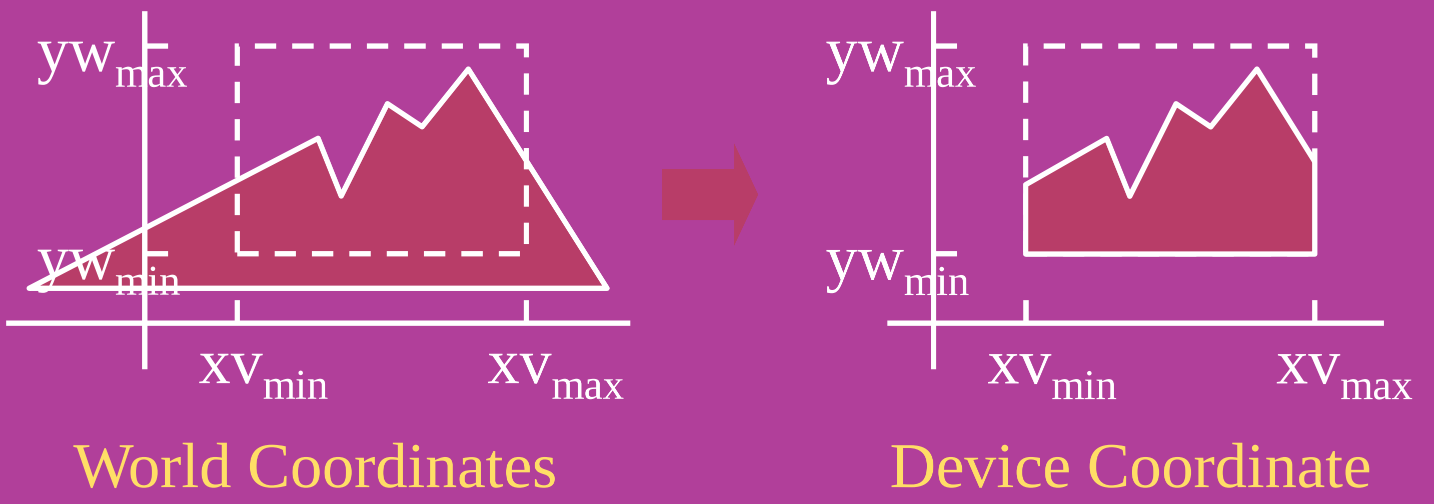
Sudipto Chaki

Lecturer, CSE, BUBT

Contents

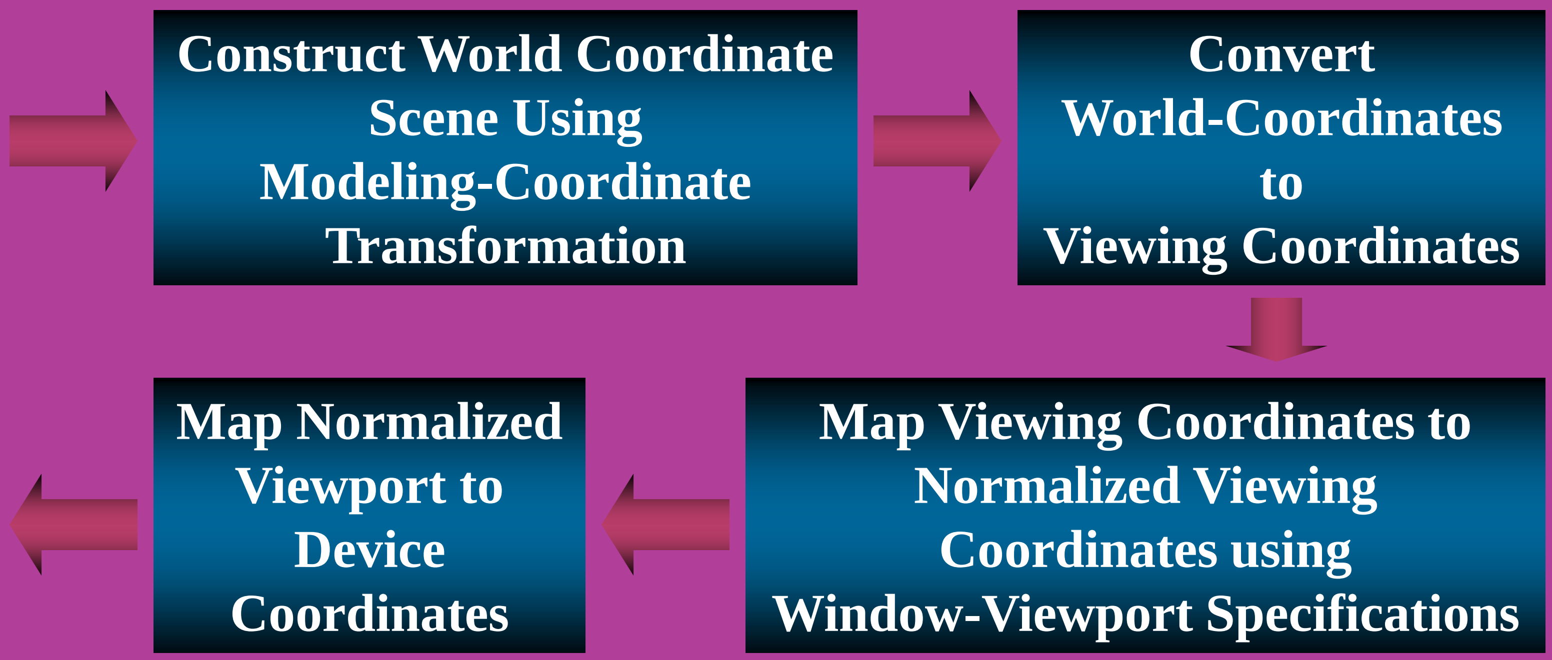
The Viewing Pipeline (1/3)

- What's the viewing pipe line??
  - Viewing transformation in several steps
- A viewing transformation using standard rectangles for the window and viewport



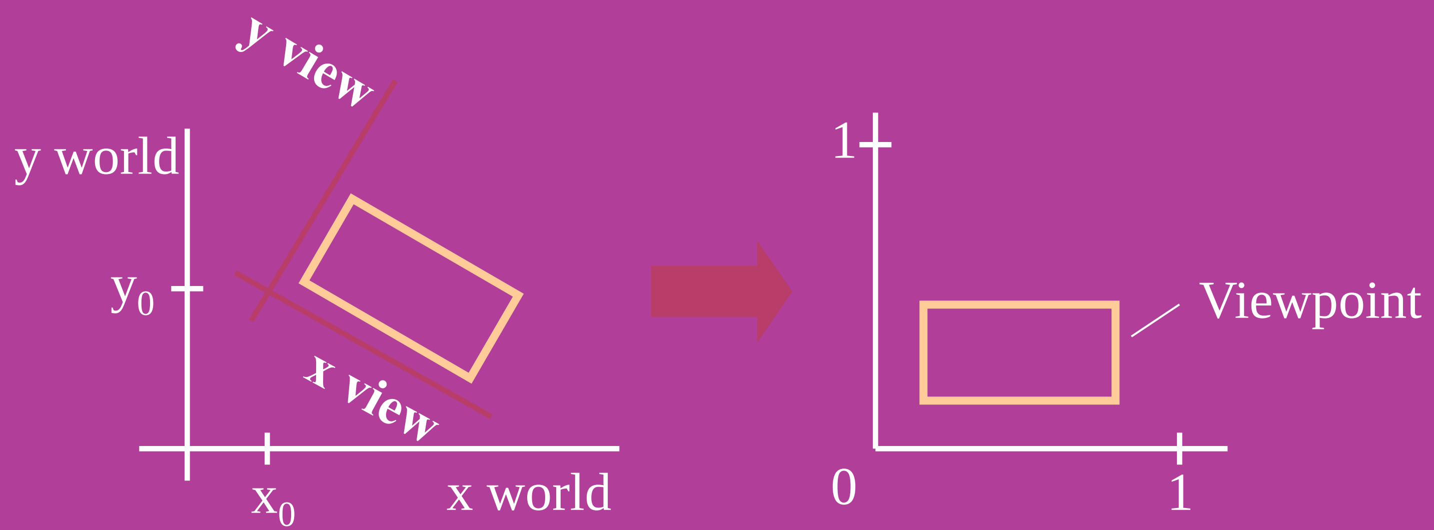
The Viewing Pipeline (2/3)

- The two-dimensional viewing-transformation pipeline



The Viewing Pipeline (3/3)

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



World Coordinates

Normalized Device Coordinates

Viewing Coordinate Reference Frame (1/2)

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

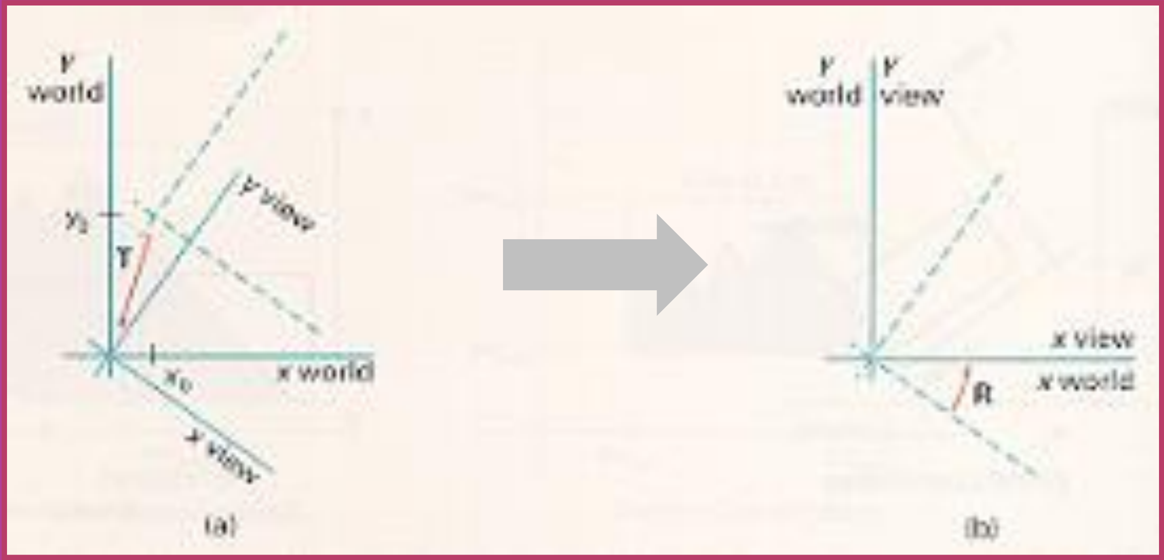
$$\mathbf{M}_{WC,VC} = \mathbf{R} \cdot \mathbf{T}$$

R: rotation matrix

T: translation matrix

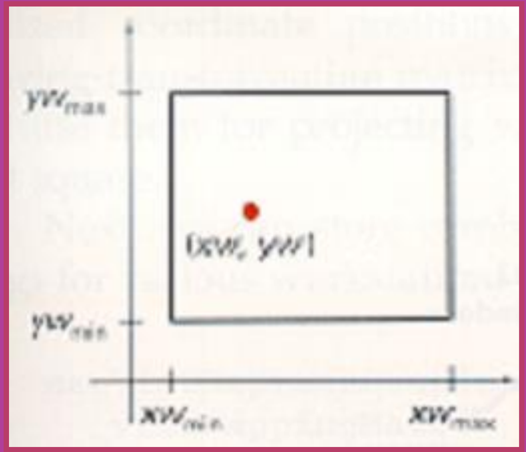
Viewing Coordinate Reference Frame (2/2)

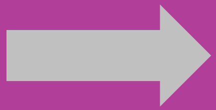
- The steps in this coordinate transformation
  - A viewing coordinate frame is moved into coincidence with the world frame in two steps
    a) Translate the viewing origin to the world origin, then
    b) Rotate to align the axes of the two systems

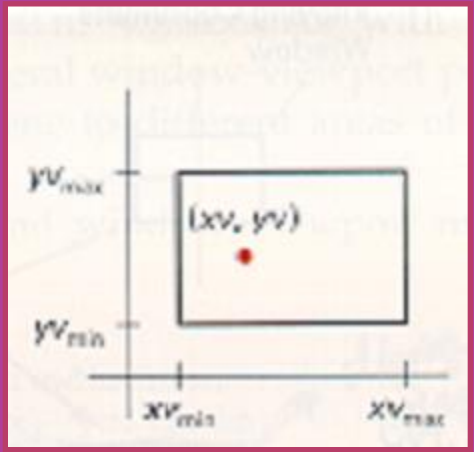


Window-To-Viewport Coordinate Transformation (1/5)

- 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 Coordinate Transformation (2/5)

- To maintain the same relative placement

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

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

- Solving these expressions for the viewport position (xv, yv)

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

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

Window-To-Viewport Coordinate Transformation (3/5)

- The scaling factors

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$

$$sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

- Conversion sequence of transformation

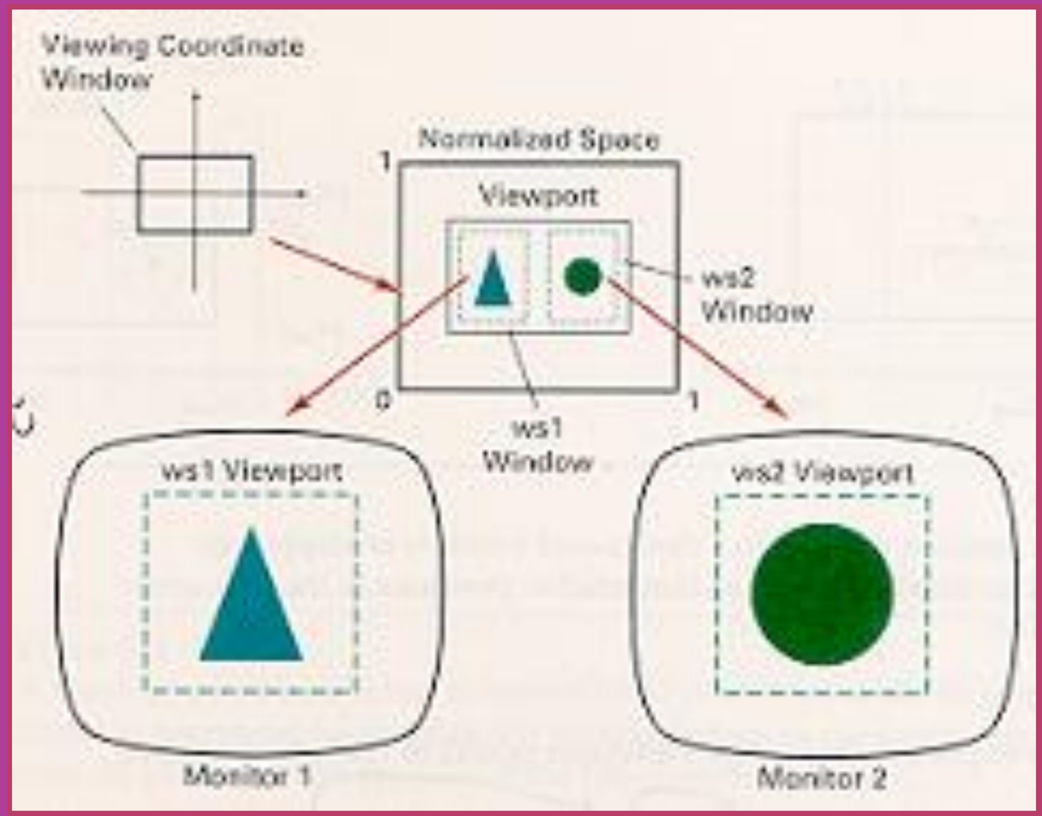
1. Perform a scaling transformation using a fixed-point position of (xw_{min}, yw_{min}) that scales the window area to the size of the viewport
2. Translate the scaled window area to the position of the viewport

Window-To-Viewport Coordinate Transformation (4/5)

- 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 Coordinate Transformation (5/5)

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



Two-Dimensional Viewing Functions (1/2)

- Definition about a viewing reference system
  - evaluateViewOrientationMatrix (x0, y0, xV, yV, error, viewMatrix)
- Setting up the elements of a window-to-viewport mapping matrix
  - setviewRepresentation (ws, viewIndex, viewMatrix, viewMap-pingMatrix, xclipmin, xclipmin, xclipmin, xclipmin, clipxy)
- Storing combinations of viewing and window-viewport mappings for various workstations in a viewing table
  - evaluateViewMappingMatrix (xwmin, xwmax, ywmin, ywmax, xvmin, xvmax, yvmin, yvmax, error, viewMappingMatrix)

Two-Dimensional Viewing Functions (2/2)

- Selection of a paticular set of options from the viewing table
  - setViewIndex (viewIndex)
- Selection of a workstation window-viewport pair
  - setWorkstationWindow (ws, xwsWindmin, xwsWindmax, ywsWindmin, ywsWindmax)
  - setWorkstationViewport (ws, xwsVPortmin, xwsVPortmax, ywsVPortmin, ywsVPortmax)

Clipping Operations

- Clipping
  - Any procedure that identifies those portions of a picture that are either inside or outside of a specified region of space
- Applied in World Coordinates
- Adapting Primitive Types
  - Point
  - Line
  - Area (or Polygons)
  - Curve, Text (omit!!)

Point Clipping

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

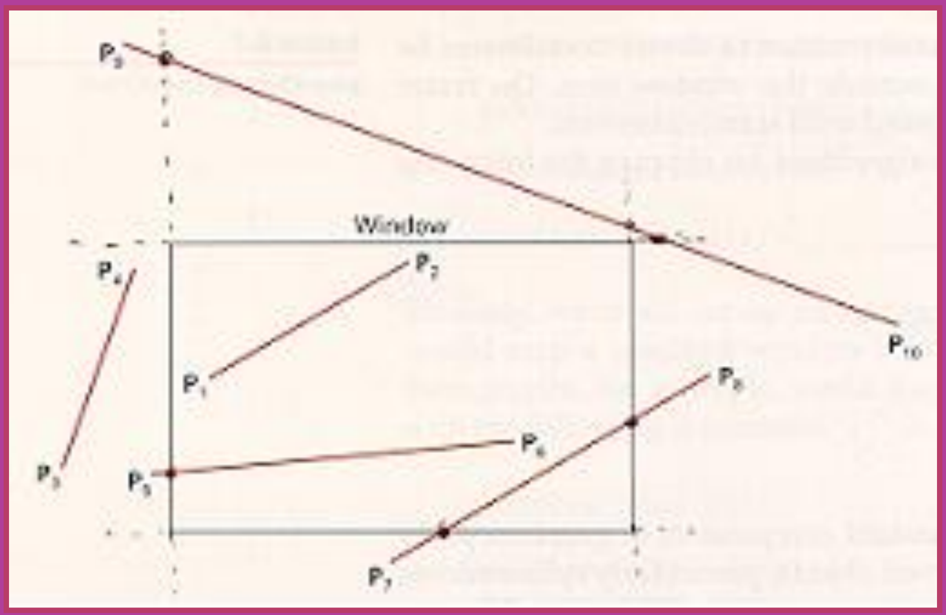
$$xw_{\min} \le x \le xw_{\max}$$

$$yw_{\min} \le y \le yw_{\max}$$

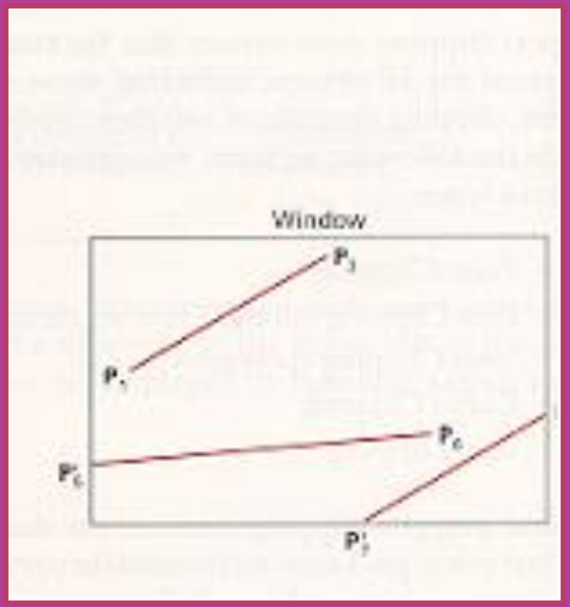
- Appling Fields
  - Particles (explosion, sea foam)

Line Clipping (1/3)

- Line clipping against a rectangular clip window



a) Before Clipping



b) After Clipping

Line Clipping (2/3)

- Parametric representation of Line segment with endpoints (x1, y1) and (x2, y2)

$$x = x_1 + u(x_2 - x_1)$$

$$y = y_1 + u(y_2 - y_1), \quad 0 \le u \le 1$$

- Exterior of the window
  – Intersection with outside the range u
- Interior of the window
  – Intersection with inside the range u

Line Clipping (3/3)

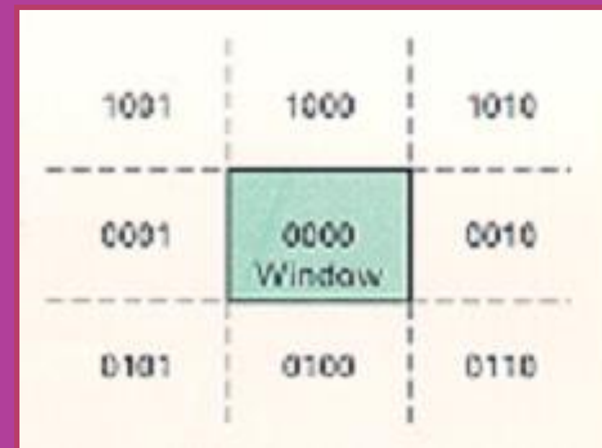
- Cohen-Sutherland Line Clipping
- Liang-Barsky Line Clipping
- NLN(Nicholl-Lee-Nicholl) Line Clipping
- Line Clipping Using Nonrectangular Clip Windows
- Splitting Concave Polygons

Cohen-Sutherland Line Clipping (1/3)

- Region Code Creation
  - Region Code

1	2	3	4

    - Bit 1: above window
    - Bit 2: below window
    - Bit 3: right of window
    - Bit 4: left of window



  - Calculate differences between endpoint coordinates and clipping boundaries
  - Use the resultant sign bit of each difference calculation to set the corresponding value in the region code

Cohen-Sutherland Line Clipping (2/3)

- Line visible if both the region codes 0000
- Not visible if bitwise logical AND of the codes is not 0000
- Clipping candidate if bitwise logical AND of region codes is 0000

Cohen-Sutherland Line Clipping (3/3)

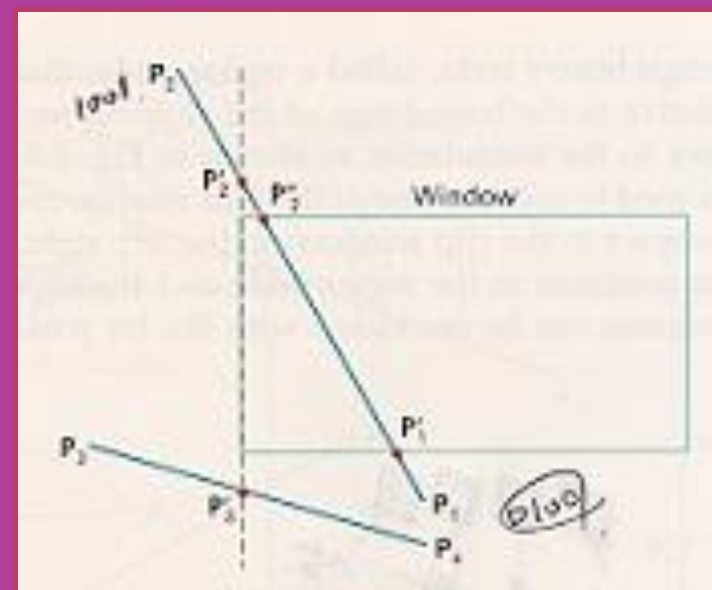
- Calculate Intersection Point
  - Using the slope-intercept form
  - Vertical Boundary

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

  - Horizontal Boundary

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

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



Liang-Barsky Line Clipping (1/4)

- Developed that are based on analysis of the parametric equation of line segment

$$x = x_1 + u\Delta x \qquad \Delta x = x_2 - x_1 \qquad 0 \le u \le 1$$

$$y = y_1 + u\Delta y, \quad \Delta y = y_2 - y_1$$

- From pre-condition in the parametric form

$$xw_{\min} \le x_1 + u\Delta x \le xw_{\max}$$

$$yw_{\min} \le y_1 + u\Delta y \le yw_{\max}$$

Liang-Barsky Line Clipping (2/4)

- Inequalities can be expressed as

  $up_k \leq q_k ,$ k = 1, 2, 3, 4

- Definition of parameter p, q

$$p_1 = -\Delta x, \quad q_1 = x_1 - xw_{\min}$$

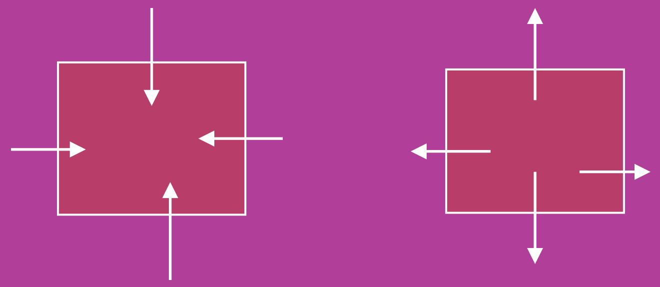
$$p_2 = \Delta x, \quad q_2 = xw_{\max} - x_1$$

$$p_3 = -\Delta y, \quad q_3 = y_1 - yw_{\min}$$

$$p_4 = \Delta y \quad q_4 = yw_{\max} - y_1$$

Liang-Barsky Line Clipping (3/4)

- $p_k = 0$
  - $q_k = 0$: boundary
  - $q_k < 0$: rejection
  - $q_k > 0$: **test!!**



- $p_k < 0$
  - Line proceeds from outside to inside
- $p_k > 0$
  - Line proceeds from inside to outside

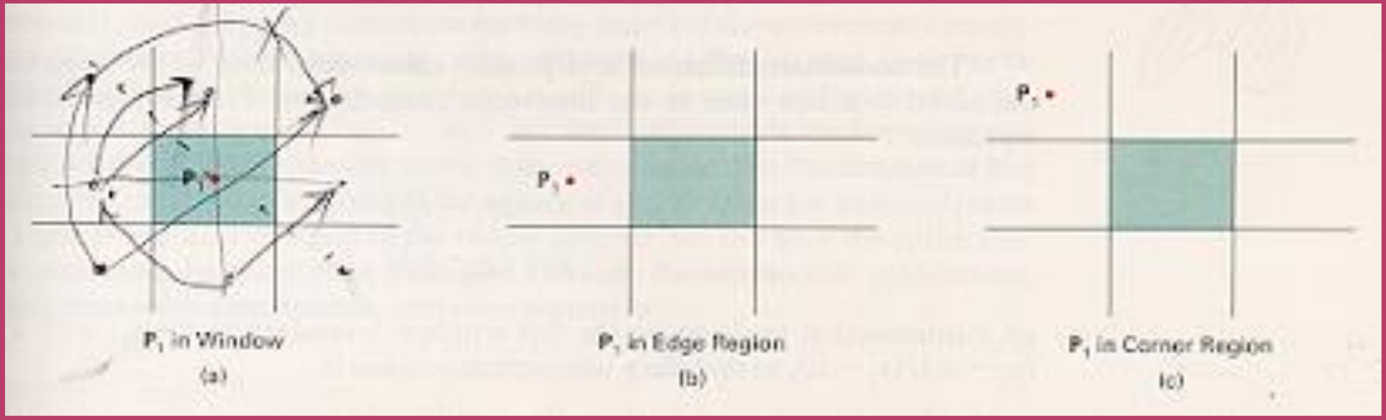
Liang-Barsky Line Clipping (4/4)

- $p_k < 0$
  - $u_1 = \min(u)$
- $p_k > 0$
  - $u_2 = \max(u)$
- $u_1 > u_2$
  - Rejection
- $u_1 < u_2$ or $u_1 = u_2$
  - Draw

$$u = \frac{q_k}{p_k}$$

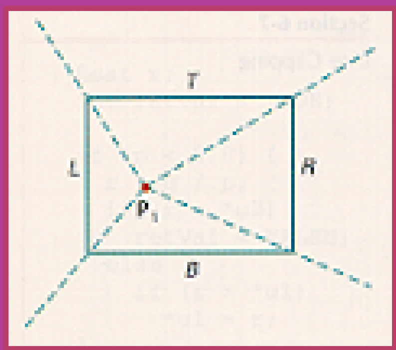
NLN Line Clipping(1/4)

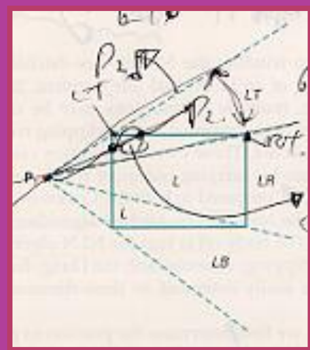
- Three possible position for a line end point P_1
  - Equal position with rotation or translation

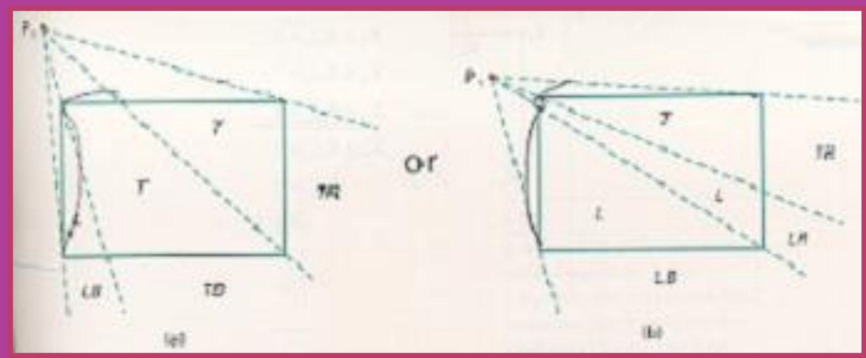


NLN Line Clipping (2/4)

- The four clipping region when P_1 is inside the clip window
- The four clipping region when P_1 is directly left of the clip window
- The two possible sets of clipping regions when P_1 is above and to the left of the clip window







NLN Line Clipping (3/4)

- Region Determination
  - P_1 is left of the clipping rectangle, then P_2 is in the region LT if

$$\text{Slope } P_1P_{TR} < \text{slope } P_1P_2 < \text{slope } P_1P_{TL}$$

or

$$\frac{y_T - y_1}{x_R - x_1} < \frac{y_2 - y_1}{x_2 - x_1} < \frac{y_T - y_1}{x_L - x_1}$$

  - Clipping the entire line

$$(y_T - y_1)(x_2 - x_1) < (x_L - x_1)(y_2 - y_1)$$

NLN Line Clipping (4/4)

- Intersection Position Calculation

$$x = x_1 + (x_2 - x_1)u$$

$$y = y_1 + (y_2 - y_1)u$$

  - Left Boundary

$$x = x_L$$

$$u = (x_L - x_1)/(x_2 - x_1)$$

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_L - x_1)$$

  - Top Boundary

$$y = y_T$$

$$u = (y_T - y_1)/(y_2 - y_1)$$

$$x = x_1 + \frac{x_2 - x_1}{y_2 - y_1}(y_T - y_1)$$

Line Clipping Using Nonrectangular Clip Windows

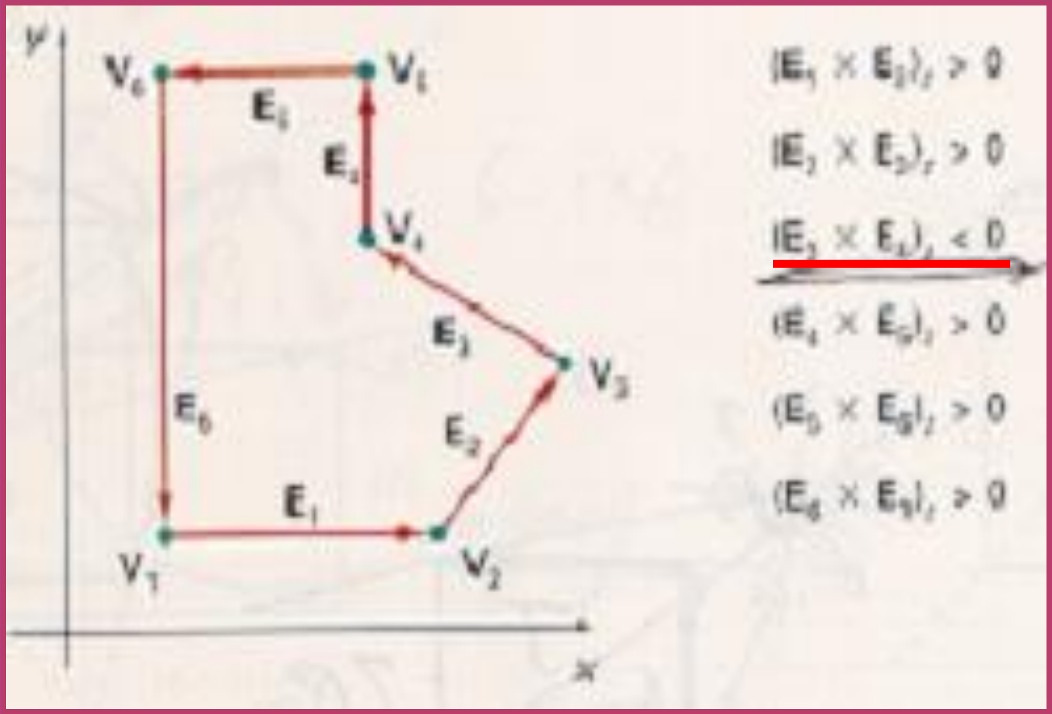
- Line clipping against arbitrarily shaped polygon
- Modifying the algorithm to include the parametric equations for the boundaries of the clip region
- Concave Polygon Clipping Region
  - Splitting concave polygon into a set of convex polygons
- Circle, Curved-Boundary Clipping Region
  - Less Commonly Used
  - Very Slow Algorithm

Polygon Clipping

- Convex polygon and concave polygon: In a convex polygo n, a line joining any two interior points of the polygon lies completely inside the polygon
- Positive orientation (counter clockwise orientation) and ne gative orientation (clockwise orientation)
- Let $A(x_1, y_2)$ and $B(x_2, y_2)$ be the end points of a directed li ne segments. A point $P(x, y)$ will be to the left of the line s egment if the expression $C = (x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1)$ is positive.

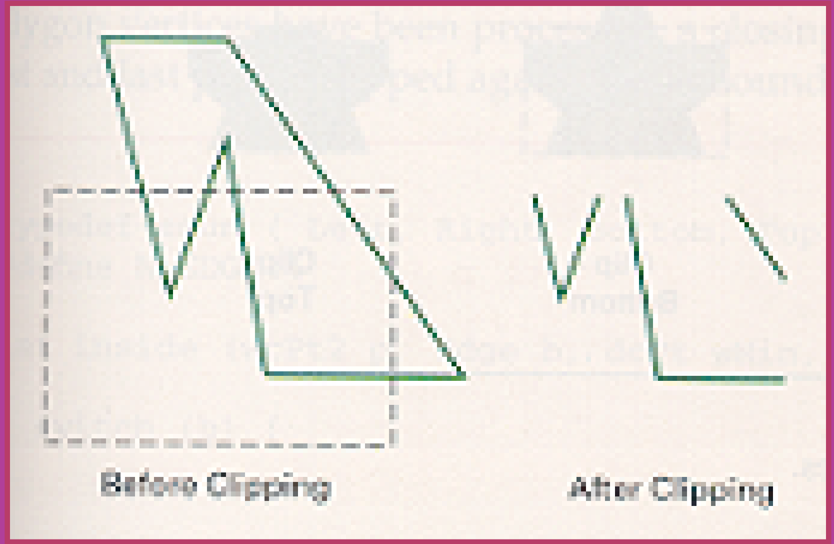
Splitting Concave Polygons (1/2)

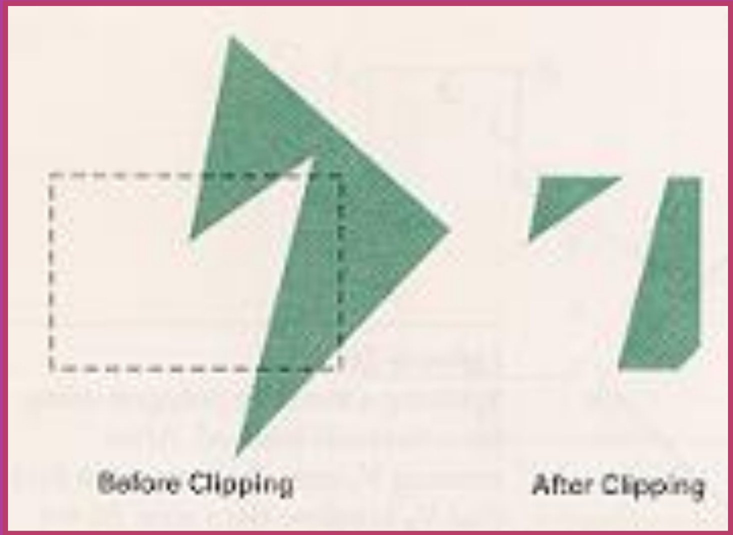
- Vector Method
  - Identifying a concave polygon by calculating cross products of successive pairs of edge vectors



Polygon Clipping (1/2)

- Display of polygon processed by a line-clipping algorithm





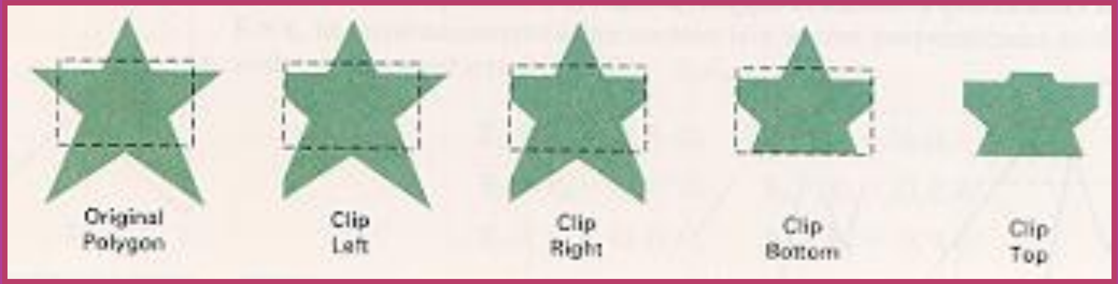
- Display of a correctly clipped polygon

Polygon Clipping (2/2)

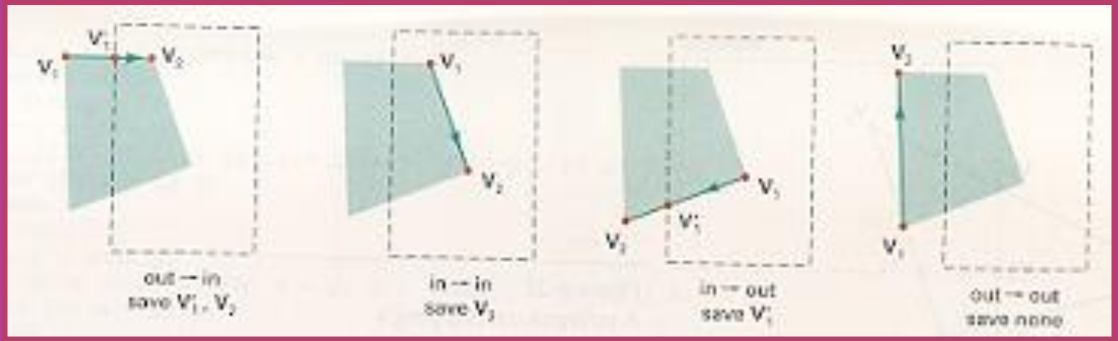
- Sutherland-Hodgeman Polygon Clipping
- Weiler-Atherton Polygon Clipping
- Other Polygon-Clipping Algorithm

Sutherland-Hodgeman Polygon Clipping (1/2)

- Clipping a polygon against successive window boundary

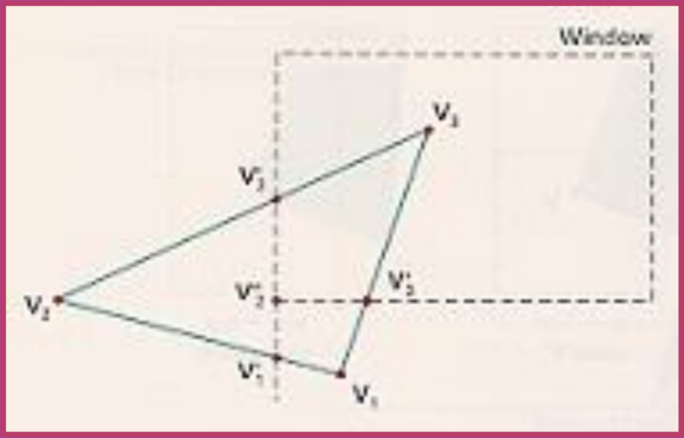


- Successive processing of pairs of polygon vertices against the left window boundary

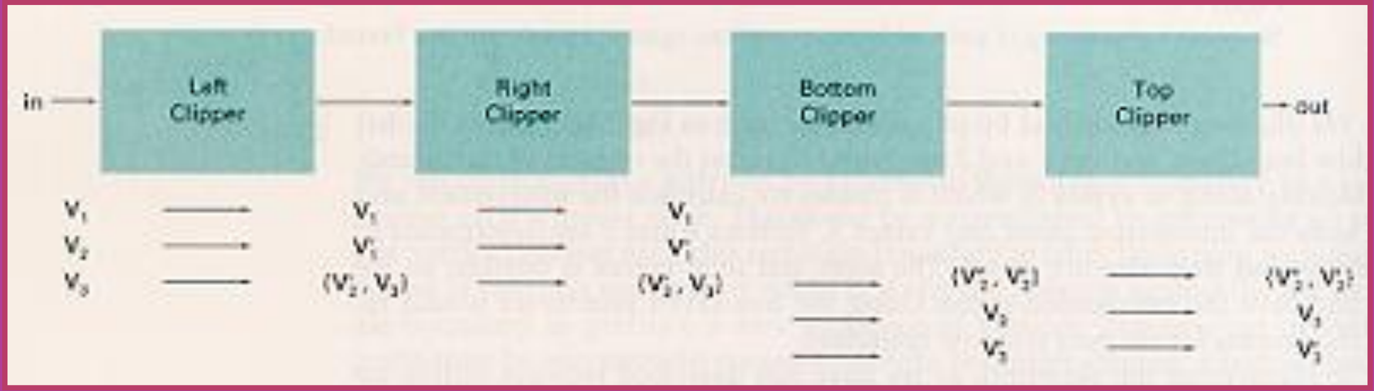


Sutherland-Hodgeman Polygon Clipping (2/2)

- A polygon overlapping a rectangular clip window

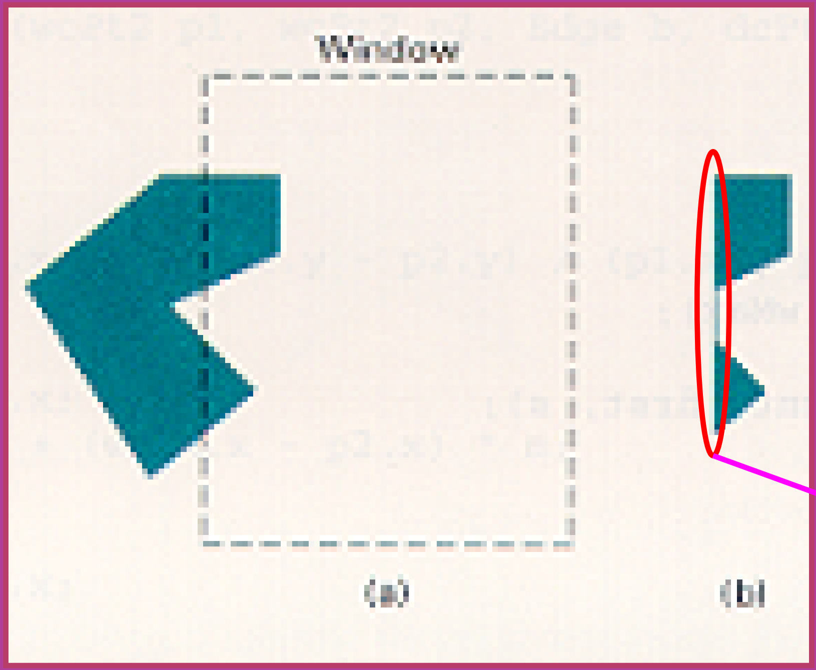


- Processing the vertices of the polygon



Weiler-Atherton Polygon Clipping (1/2)

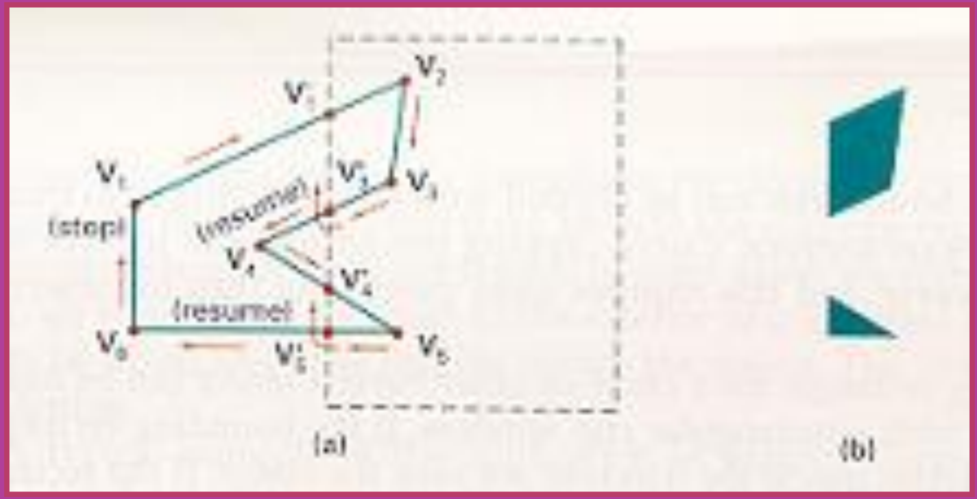
- Problem of Sutherland-Hodgeman clipping
  - Displaying extraneous line



??

Weiler-Atherton Polygon Clipping (2/2)

- Rules
  - For an outside-to-inside pair of vertices, follow the polygon boundary
  - For an inside-to-outside pair of vertices, follow the window boundary in clockwise direction
- Correct Result



Other Polygon-Clipping Algorithm

- Extending parametric line-clipping method
  - Well suited for convex polygon-clipping
  - Using region testing procedures
- Clipping a polygon by determining the intersection of two polygon areas