Polygon Clipping Algorithm(30.3.2020)

- Sutherland-Hodgeman Polygon Clipping
- Weiler-Atherton Polygon Clipping

Sutherland-Hodgeman Polygon Clipping:

It is performed by processing the boundary of polygon against each window corner or edge. First of all entire polygon is clipped against one edge, then resulting polygon is considered, then the polygon is considered against the second edge, so on for all four edges

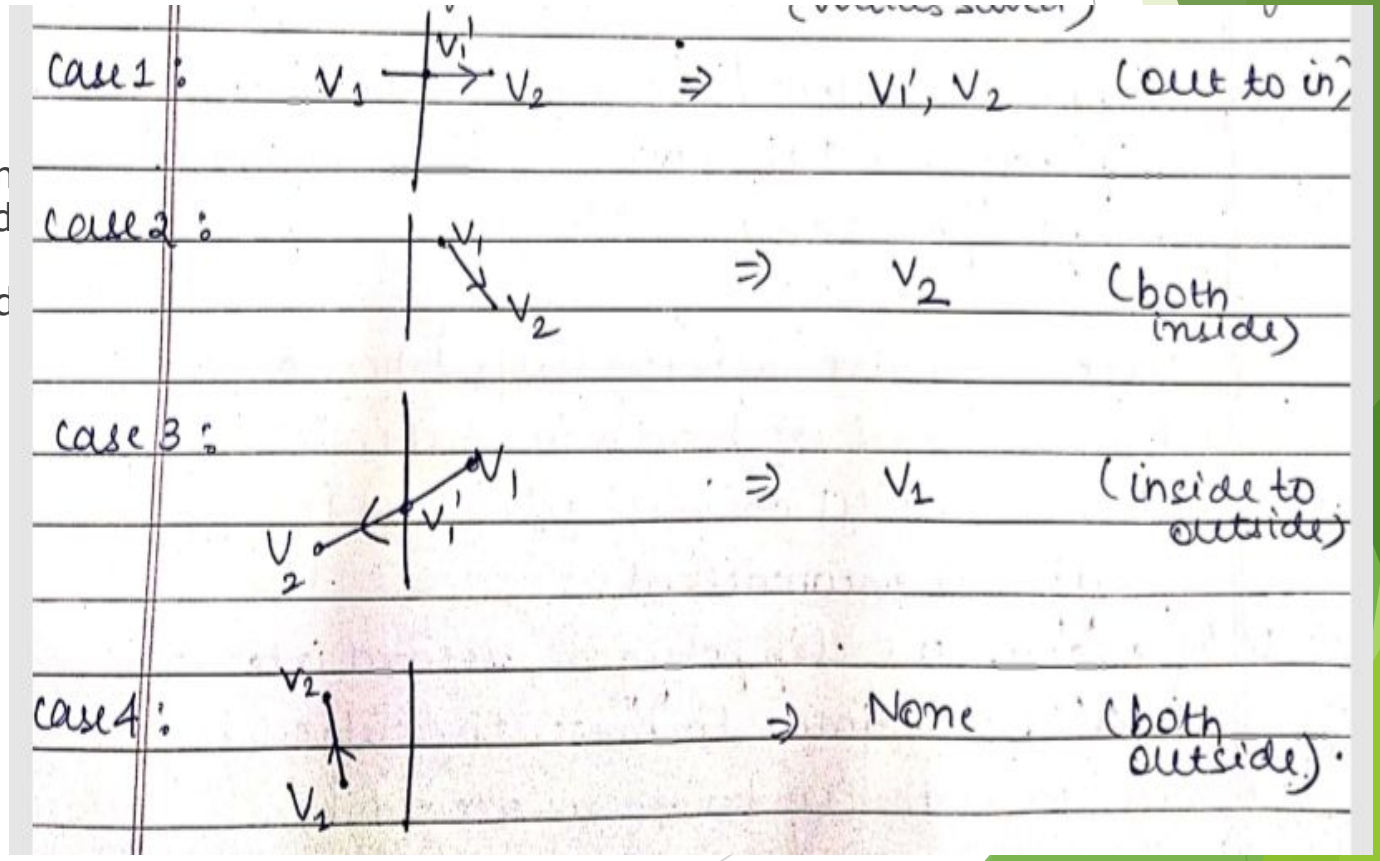
Four possible situations while processing

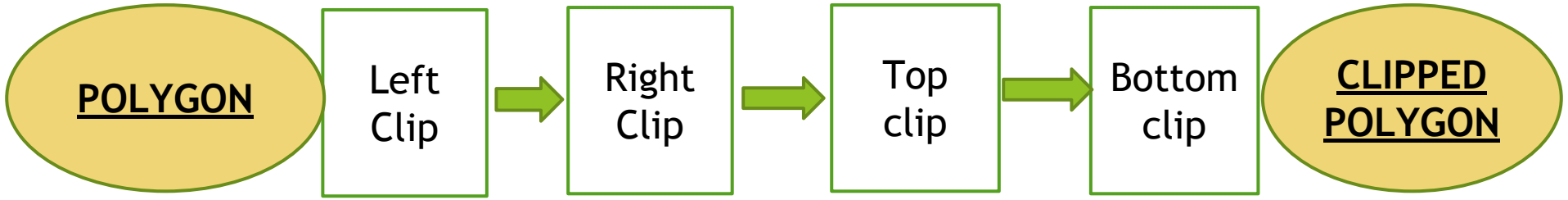
1) If the first vertex is an outside the window, th second vertex is inside the window. Then second vertex is added to the output list. The point of intersection of window boundary and polygon sid (edge) is also added to the output line.

2) If both vertexes are inside window boundary. Then only second vertex is added to the output list.

3) If the first vertex is inside the window and second is an outside window. The edge which intersects with window is added to output list.

4) If both vertices are the outside window, then nothing is added to output list.





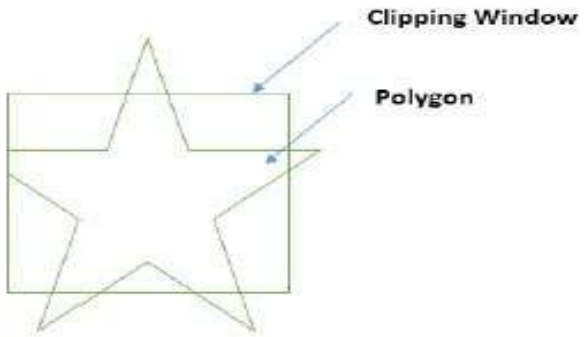


Figure: Clipping Left Edge

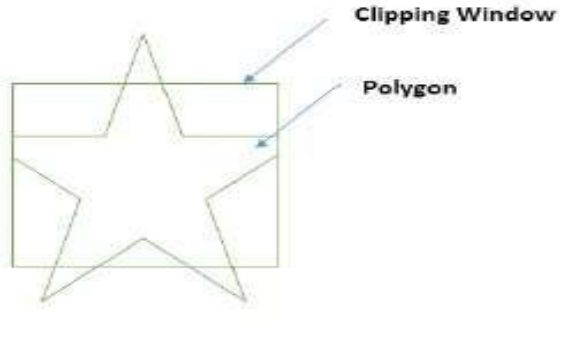


Figure: Clipping Right Edge

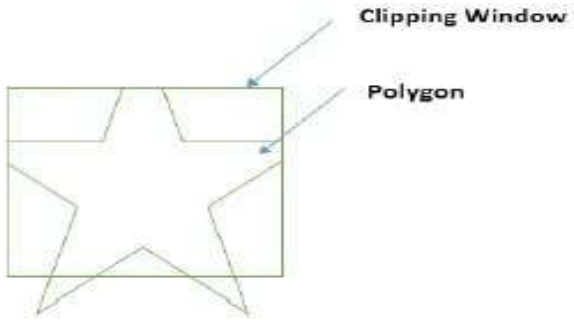


Figure: Clipping Top Edge

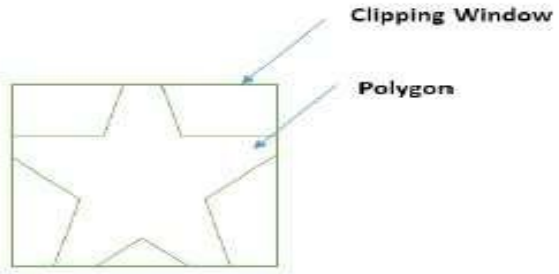


Figure: Clipping Bottom Edge

Disadvantage of Sutherland Hodgmen Algorithm:

1. Convex polygon are currently clipped with this algorithm, but concave polygon cannot be clipped
2. This occur when a clipped polygon has two or more separate sections but there is only one output list.
3. The last vertex is always joined to first vertex.
4. This method requires a considerable amount of memory. The first of all polygons are stored in original form. Then clipping against left edge done and output is stored. Then clipping against right edge done, then top edge. Finally, the bottom edge is clipped. Results of all these operations are stored in memory. So wastage of memory for storing intermediate polygons

There are two sub-problems that need to be discussed before implementing the algorithm:-

- **To decide if a point is inside or outside the clipper polygon**

If the vertices of the clipper polygon are given in clockwise order then all the points lying on the right side of the clipper edges are inside that polygon. This can be calculated using:

Given that the line starts from (x_1, y_1) and ends at (x_2, y_2)

$$P = (x_2 - x_1)(y - y_1) - (y_2 - y_1)(x - x_1)$$

if $P<0$, the point is on the right side of the line

$P=0$, the point is on the line

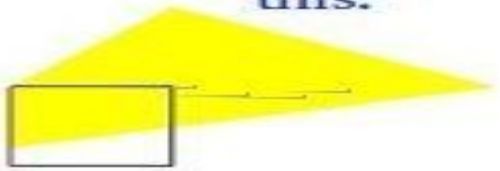
$P>0$, the point is on the left side of the line

- To find the point of intersection of an edge with the clip boundary

We use general equations of line to compute intersection point

Vertex List : (100, 150) (200, 250)
(300, 200)

Left edge of clipping area is extended infinitely and polygon points are clipped using this.



Vertex List : (150, 200) (200, 250)
(300, 200) (150, 162)

Top edge is extended infinitely

Vertex List : (300, 200) (150, 162)
(150, 200)

Right edge is extended infinitely

Vertex List : (200, 174) (150, 162)
(150, 200) (200, 200)

Bottom edge is extended infinitely

Vertex List : (150, 162) (150, 200)
(200, 200) (200, 174)

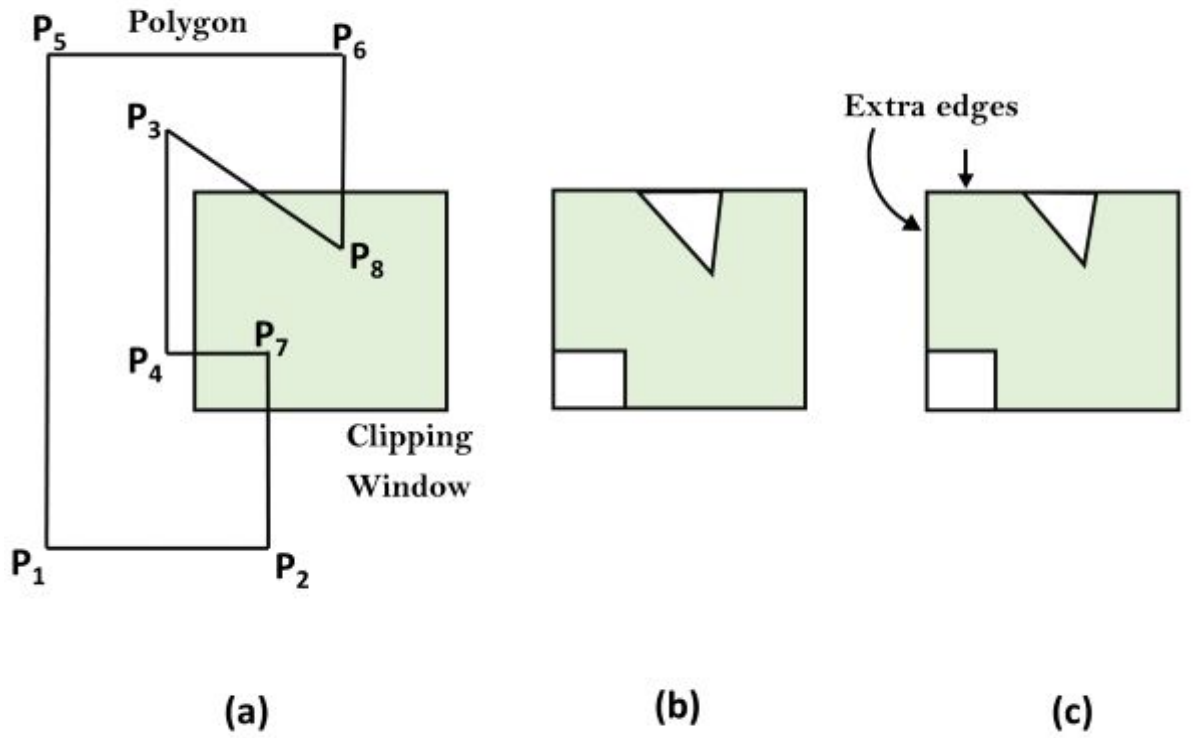
Weiler-Atherton Polygon Clipping:

- When the clipped polygons have two or more separate sections, then it is the concave polygon handled by this algorithm. The vertex-processing procedures for window boundaries are modified so that concave polygon is displayed.

- Let the clipping window be initially called clip polygon and the polygon to be clipped the subject polygon. We start with an arbitrary vertex of the subject polygon and trace around its border in the clockwise direction until an intersection with the clip polygon is encountered:

CASE 1

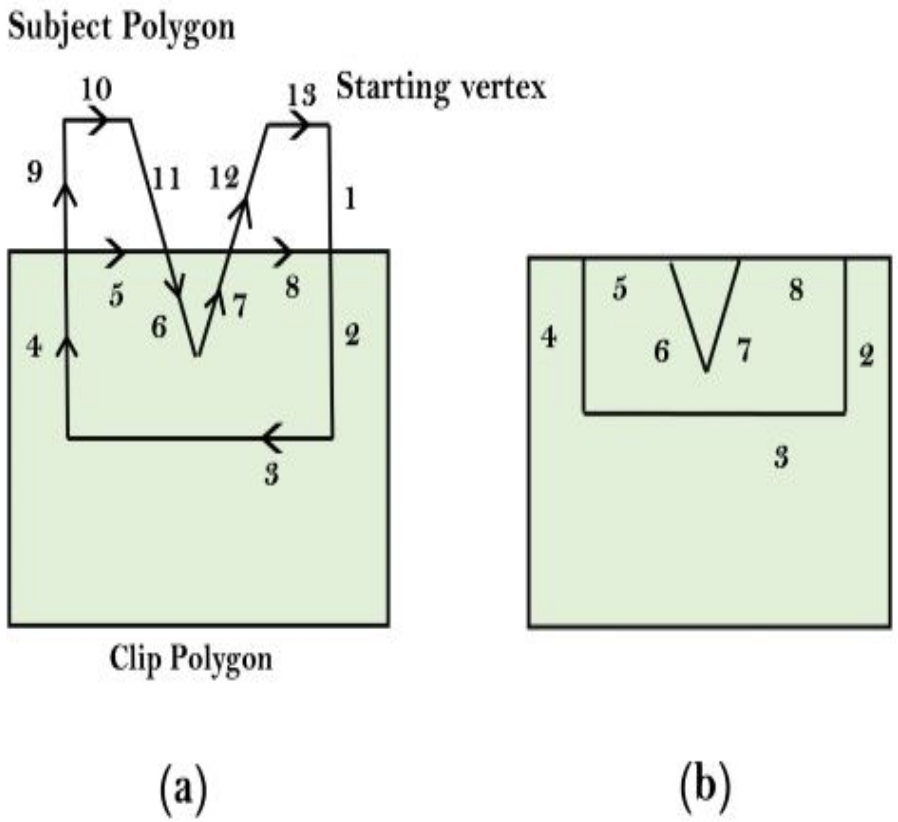
1. If the edge enters the clip polygon, record the intersection point and continue to trace the subject polygon.



CASE 2

If the edge leaves the clip polygon, record the intersection point and make a right turn to follow the clip polygon in the same manner (i.e., treat the clip polygon as subject polygon and the subject polygon as clip polygon and proceed as before).

Whenever our path of traversal forms a sub-polygon we output the sub-polygon as part of the overall result. We then continue to trace the rest of the original subject polygon from a recorded intersection point that marks the beginning of a not-yet traced edge or portion of an edge. The algorithm terminates when the entire border of the original subject polygon has been traced exactly once.



- For example, the number in fig (a) indicates the order in which the edges and portion of edges are traced. We begin at the starting vertex and continue along the same edge (from 1 to 2) of the subject polygon as it enters the clip polygon. As we move along the edge that is leaving the clip polygon, we make a right turn (from 4 to 5) onto the clip polygon, which is now considered the subject polygon. Following the same logic leads to the next right turn (from 5 to 6) onto the current clip polygon, this is the original subject polygon. With the next step done (from 7 to 8) in the same way, we have a sub-polygon for output in fig (b). We then resume our traversal of the original subject polygon from the recorded intersection point where we first changed our course. Going from 9 to 10 to 11 produces no output. After skipping the already traversed 6 and 7, we continue with 12 and 13 and come to an end. The fig (b) is the result.