Table 5.6. Computation of Parametric intersections (Example 5.14)

Edge	n	Q	w	$n \cdot w$	$n \cdot d$	t_{min}	t_{max}
Left	i	(0, 0)	$-i+j$	-1	10	$\frac{1}{10}$	—
Right	$-i$	(8, 4)	$-9i-3j$	9	-10	—	$\frac{9}{10}$
Bottom	j	(0, 0)	$-i+j$	1	2	$-\frac{1}{2}$	—
Top	$-j$	(8, 4)	$-9i-3j$	3	-2	—	$\frac{3}{2}$

The above example dealt with a conventional regular rectangular clipping window. However, the significance of the Cyrus Beck algorithm is its applicability to windows of arbitrary shapes (convex). In the following example, we consider one such non-rectangular window.

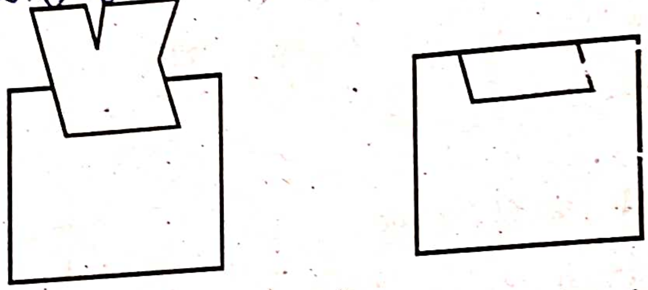
5.7. POLYGON CLIPPING

Line clipping algorithm cannot be used on a polygon because we must generate new edges along the window boundaries as well as clip the original edges. For example,

A polygon is called **convex** if the line joining any two interior points of the polygon lies completely inside the polygon. A non-convex polygon is said to be **concave.**

A polygon with vertices $P_1, P_2, ..., P_N$ is said to be **positively oriented,** if a tour of the vertices in the given order produces a **counterclockwise circuit.**

(anticlockwise)



(a) Before clipping (b) After Clipping

Fig. 5.17.

Let $A(x_1, y_1)$ and $B(x_2, y_2)$ be the endpoints of a directed line segment A point $P(x, y)$ will be to the left of the line segment if the expression

$$C = (x_2 - x_1)(y - y_1) - (y_2 - y_1)$$

$(x - x_2)$ is positive.

We say that the point is to the right of the line segment if this quantity is negative. If a point P is to the right of any one edge of a positively oriented, convex polygon, it is outisde the polygon. If it is to the left of every edge of the polygon, it is inside the polygon.

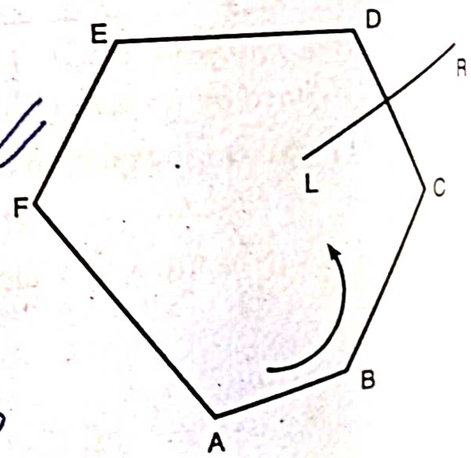


Fig. 5.18.

A polygon boundary processed with a line clipper may be displayed as a series of unconnected line segments. For example display of a polygon processed by a line-clipping algorithm is as follows:

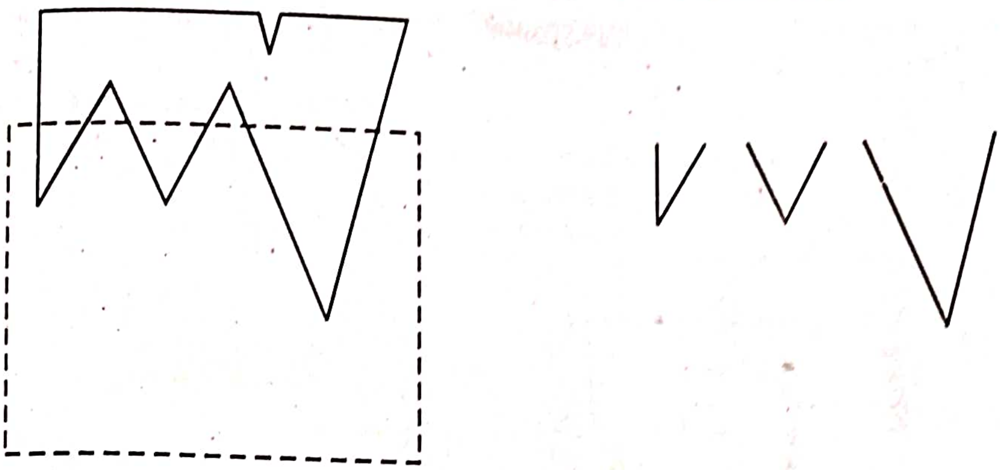


Fig. 5.19.

For polygon clipping, we require an algorithm that will generate one or more closed areas that are then scan converted for the appropriate area fill. Thus, the output of a polygon clipper should be a sequence of vertices that defines the clipped polygon boundaries.

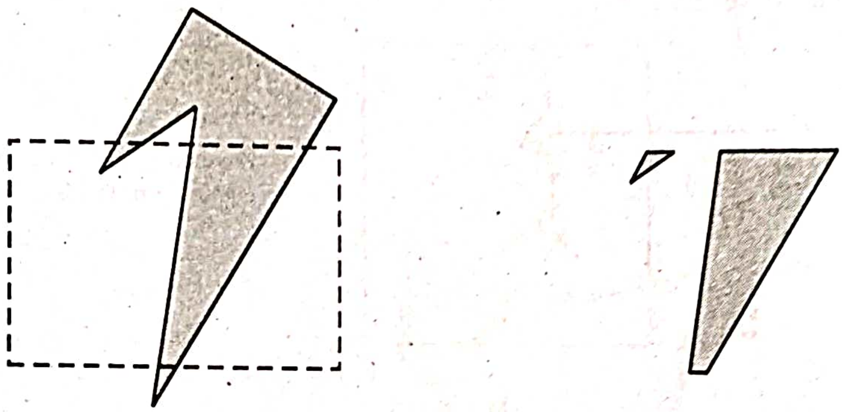


Fig. 5.20. Display of a correctly clipped polygon

In polygon clipping, each edge of the polygon must be tested against each edge of the clip rectangle; new edges must be added and existing edges must be discarded, retained or divided. Multiple polygons may result from clipping a single polygon. We need an organized way to deal with all these cases.

5.7.1. The Sutherland-Hodgman Polygon-Clipping Algorithm

Sutherland and Hodgman's polygon-clipping algorithm uses a *divide-and-conquer strategy*. It solves a series of simple and identicial problems that, when combined solve the overall problem.

Note the difference between this strategy for a polygon and the Cohen-Sutherland algorithm for clipping a line. The polygon clipper clips against four edges in succession, whereas the line clipper tests the outcodes to see which edge is crossed and clips only when necessary. In Sutherland-Hodgeman algorithm a polygon consists of an ordered sequence of vertices. Let $P_1, P_2, ..., P_N$ be the vertex list of the polygon to be clipped. Let edge E be any edge of the positively oriented convex clipping polygon. The edges will be processed by the clipping algorithm in the order of the vertex pairs.

The clipping algorithm will be called once for each vertex of the polygon. For each call, the algorithm will return either no vertex at all, the original vertex without change or one or more vertices.

When we are clipping a polygon with respect to any particular edge of the window at that time, we have to consider following four different cases:

Case-1: If the first vertex is outside the window boundary and the second vertex is inside the window; then the intersection point of polygon with boudnary edge of window and the vertex which is inside the window is stored in a output vertex list *i.e.*, P_1P_2 edge.

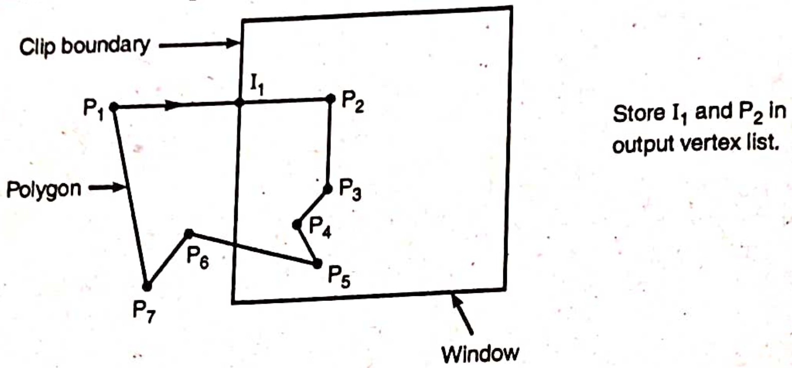


Fig. 5.21.

Case-2: If both *i.e.*, first and second vertex are inside the window boundary then we have to store the second vertex only in output vertex list, *i.e.*, P_2P_3 edge.

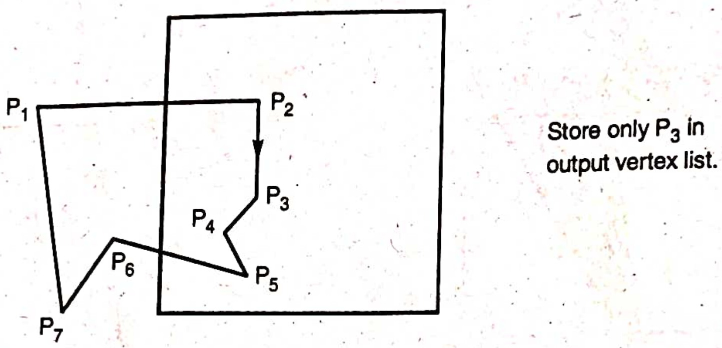


Fig. 5.22.

Case-3: If the first vertex is inside the window and second vertex is outside the window boundary then we have to store only intersection point in output vertex list *i.e.*, P_5P_6 edge.

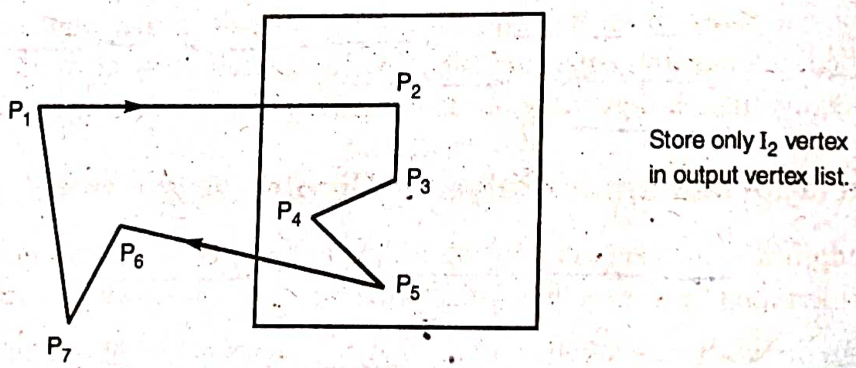


Fig. 5.23.

Case-4: If both first and second vertex of a polygon are lieing outside the window boundary then no vertex is stored in output vertex list *i.e.* P_6P_7 edge nothing is stored in output vertex list.

Once all vertices have been considered for one clip window boundary, the output list of vertices is clipped against the next window boundary.

The block diagram for this algorithm is as follows :

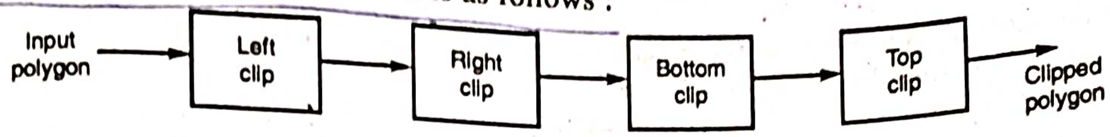


Fig. 5.24.

For example

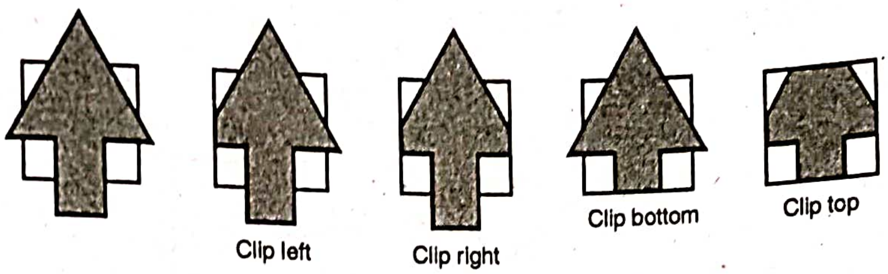


Fig. 5.25.

Flow-chart of Sutherland-Hodgman Algorithm

This algorithm inputs the vertices of a polygon one at a time. For each input vertex, either zero, one or two output vertices are generated depending on the relationship of the input vertices to the clipping edge *E*.

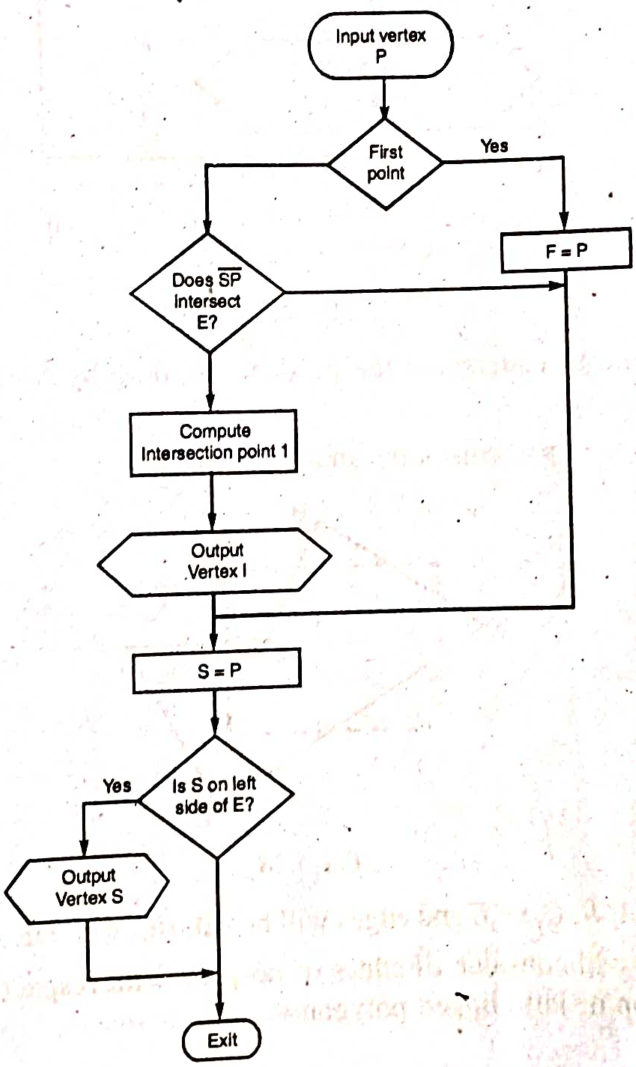


Fig. 5.26.

We denote P as the input vertex, S the previous input vertex and F the first arriving input vertex. The vertex or vertices to be output are determined according to the flow-chart below:

If the polygon has n edges then the edge $\overline{P_n P_1}$ is closing the polygon. In order to avoid the need to duplicate the input of P_1 as the final input vertex the closing logic shown in the flow chart below is called after processing the final input vertex P_n.

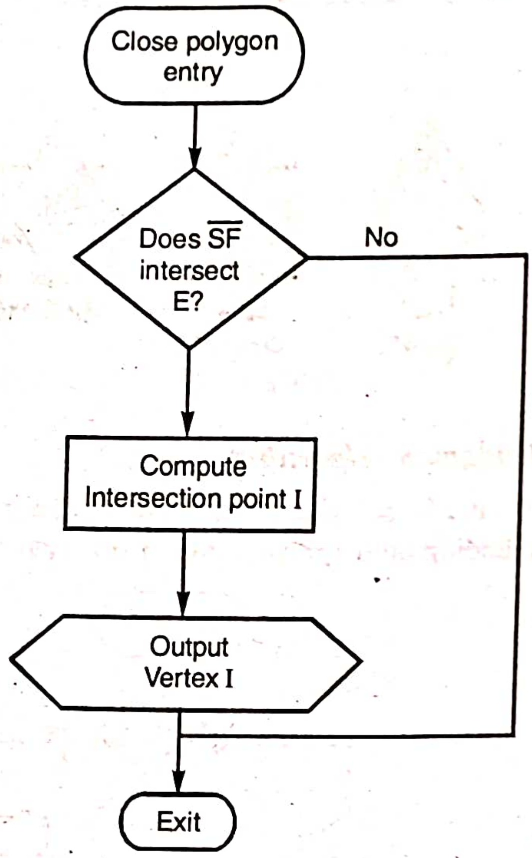


Fig. 5.27.

Let us take an example to understand the polygon clipping by Sutherland Hodgman algorithm. Suppose we are having polygon.

ABCD which we want to clip against a rectangular window.

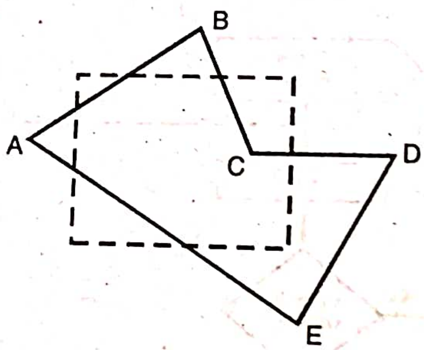
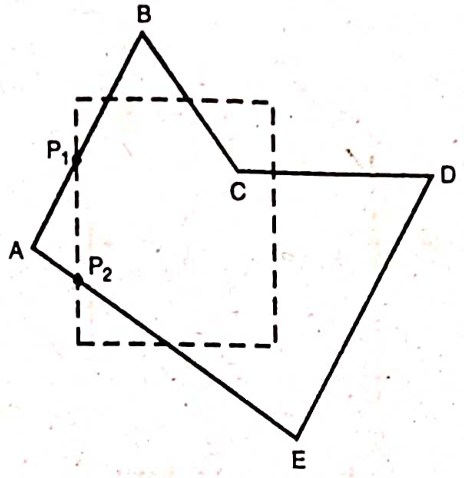


Fig. 5.28

Here vertex list will be A, B, C, D, E and edges will be AB, BC, CD, DE and EA.

Step 1. Clip Left: We will consider all edges of polygon with respect to left boundary of window. Diagrammatic representation of left clipped polygon is



Left clip

$AB \rightarrow P_1B$
$BC \rightarrow C$
$CD \rightarrow D$
$DE \rightarrow E$
$EA \rightarrow P_2$

Fig. 5.29.

So after left clipping our output vertex list will become

$$\{P_1, B, C, D, E, P_2\} \text{ i.e.,}$$

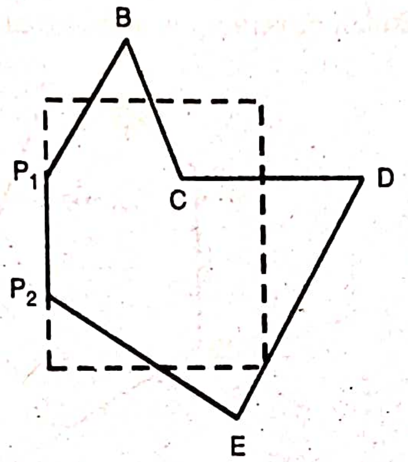
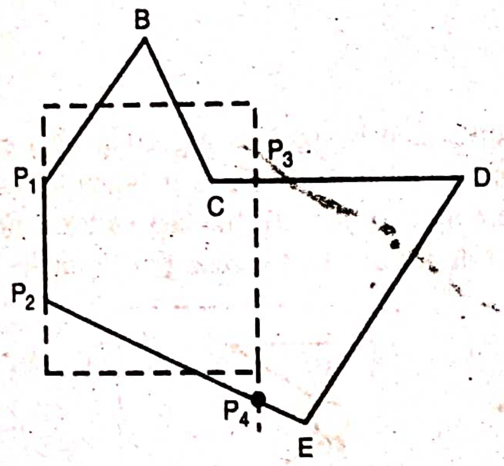


Fig. 5.30

Step 2. Clip Right: Now modified list of vertices is passed to clip right procedure.



Right clip

$P_1B \rightarrow B$
$BC \rightarrow C$
$CD \rightarrow P_3$
$DE \rightarrow$ No vertex
$EP_2 \rightarrow P_4, P_2$
$P_2P_1 \rightarrow P_1$

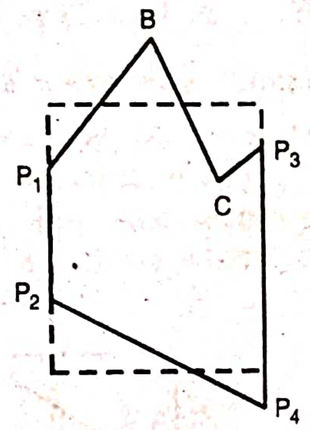


Fig. 5.31 *Fig. 5.32*

After right clipping set of vertices will be $\{B, C, P_3, P_4, P_2, P_1\}$

Step 3. Clip Bottom: Now modified list of vertices is passed to this procedure.

Bottom clip

$P_2P_1 \to P_1$
$P_1B \to B$
$BC \to C$
$CP_3 \to P_3$
$P_3P_4 \to I_1$
$P_4P_2 \to I_1, P_2$

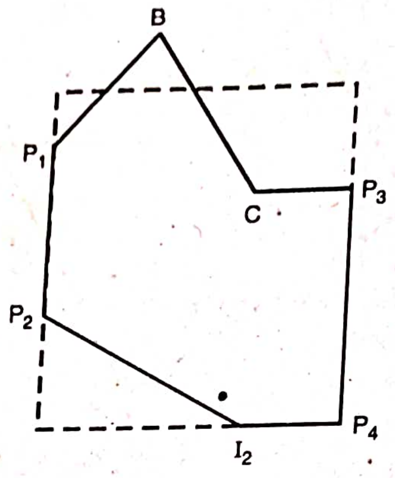


Fig. 5.33

After bottom clipping set of vertices will be $\{P_1, B, C, P_3, I_1, I_2, P_2\}$

Step 4. Clip Top: The modified list of vertices is passed to this procedure.

Top clip

$P_1B \to I_3$
$BC \to I_4, C$
$CP_3 \to P_3$
$P_3I_1 \to I_1$
$I_1I_2 \to I_2$
$I_2P_2 \to P_2$
$P_2P_1 \to P_1$

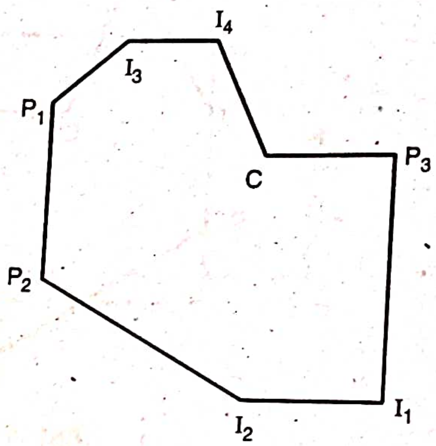


Fig. 5.34

This is the final clipped polygon.

Limitations with Sutherland-Hodgeman Algorithm

All convex polygons are correctly clipped by the Sutherland-Hodgeman algorithm, but concave polygons may be displayed with extraneous lines. This occurs when the clipped polygon should have two or more separate sections. But since there is only one output vertex list, the last vertex *i* the list is always joined to the first *i.e.*, we are forming an edge between last and first vertex. For example:

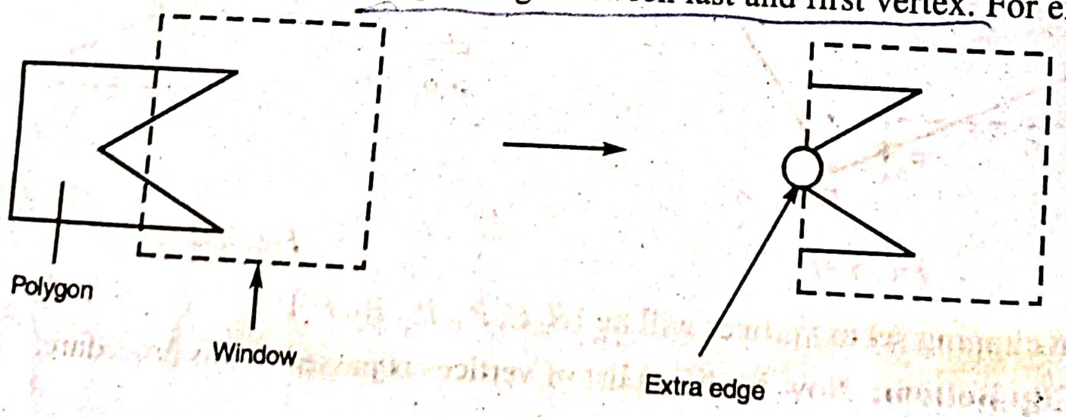


Fig. 5.35

To overcome this problem one way is to split the concave polygon into two or more convex polygons and process each convex polygon separately. Another possibility is to modify the Sutherland-Hodgeman approach to check the final vertex list for multiple vertex points along any clip window boundary and correctly join pairs of vertices. Finally, we would use a more general polygon clipping algorithm such as. Weiler-Atherton polygon clipping algorithm.

SOLVED PROBLEMS

1. **Let R be the rectangular window whose lower left hand corner is at $-L(-3, 1)$ and upper right corner is at R (2, 6). using The cohen sutherland algorithm clip the line segments AB and CD where A (–4, 2), B (–1, 7), C (–1, 5) and D (3, B).**

Ans.

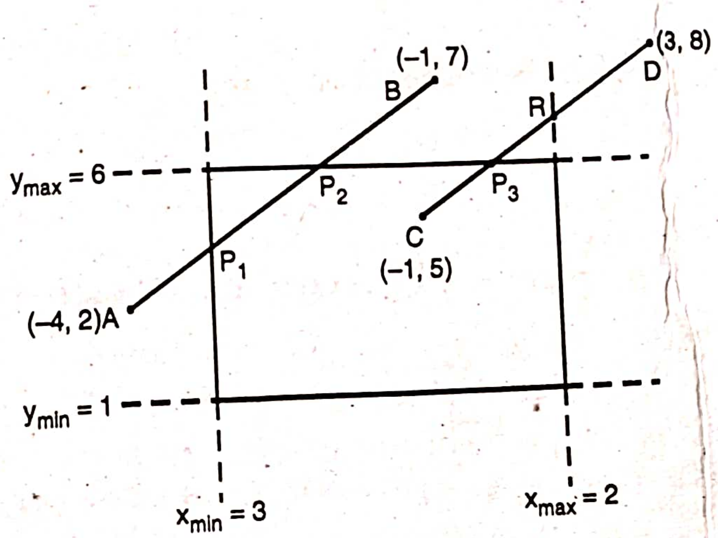


Fig. 5.36

for line AB, the region code are

A : 0001

B : 1000

Now

	0001	A
logicalAND	1000	B
	0000	(zero).

as logical AND of region code of A and B equal to zero *i.e.* line is partially visible.

Now slope of line AB is

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{7-2}{-1-(-4)} = \frac{5}{-1+4} = \frac{5}{3}$$

Point of intersection (P_1) can be calculated as

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

$$\frac{5}{3} = \frac{y-2}{-3-(-4)} \text{ as } x_1 = -4, y_1 = 2 \text{ and } x = -4 \; (x_{min})$$

$$\frac{5}{3} = \frac{y-2}{-3+4}$$

$$\frac{5}{3} = \frac{y-2}{1}$$

$$3y - 6 = 5$$

$$3y = 11 \quad \therefore y = \frac{11}{3}$$

so, Interraction point $\quad P_1 = \left(-4, \frac{11}{3}\right)$

Now for Intersection point (P_2) can be calculated as

$$m = \frac{y - y_2}{x - x_2}$$

$$\frac{5}{3} = \frac{6-7}{x-(-1)} \text{ as } (x_2 = -1, y_2 = 7 \text{ and } (y = y_{max} = 6)$$

$$\frac{5}{3} = \frac{-1}{x+1}$$

So, intersection point P_2 is

$$P_2 = \left(-\frac{8}{5}, 6\right)$$

So line P_1P_2 is visible and $\overline{AP_1}$ ans $\overline{BP_2}$ is clipped off

If for line CD, region code of

C: 0000 (C_n side window)

D: 1010

Logical AND of C and D is given as

	C : 0000
logical AND	D : 1010
	0000 (zero)

As logical AND of C and D is equal to zero *i.e.* line is partially visible

Now, slope of line is $\quad m = \dfrac{y_2 - y_1}{x_2 - x_1} = \dfrac{8-5}{3-(-1)} = \dfrac{3}{4}$

for intersection point P_3 which is intersecting with line $y = y_{max} = 6$ and its y coordinate can be calculated as

$$m = \frac{3}{4} = \frac{y - y_1}{x - x_1}$$

$$\frac{3}{4} = \frac{6-5}{x-(-1)}$$

$$\frac{3}{4} = \frac{1}{x+1}$$

$$3x + 3 = 4$$

$$3x = 4 - 3$$

$$3x = 1$$

$$x = \frac{1}{3}$$

So the intersection point P_3 is

$$P_3 = \left(\frac{1}{3}, 6\right)$$

As we can see from the figure, line CD is intersecting with two lines $y_{max} = 6(P_3)$ and $x_{max} = 2(P_4)$ but we calculate the intersection point P_3 only because it is actual boundary of the window while P_4 point is intersecting with extended boundary, so we dont calculate Point P_4.

Line portion $\overline{CP_3}$ is visible while $\overline{P_3P_4}$ and $\overline{P_4D}$ segment will be clipped off and not visible inside window.

2. **Using Cohen-Sutherland line clipping algorithm, clip the following line against a window which has lower left corner at (–4, 1) and upper right corner at (2, 7). Line $A(1, -2)$ $B(3, 3)$.**

Ans.

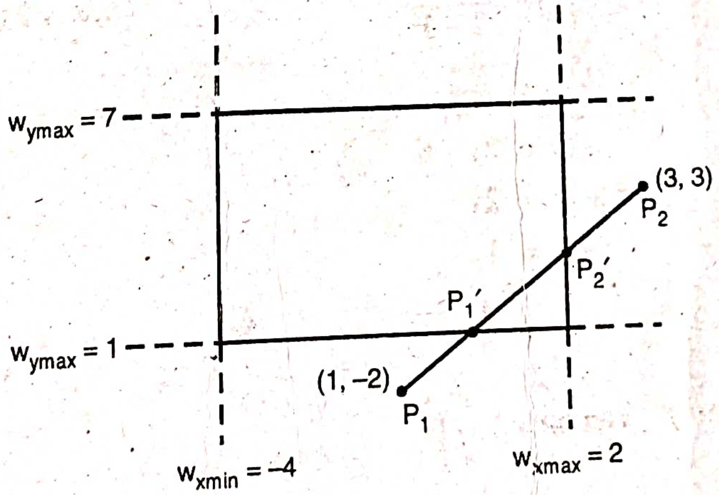


Fig. 5.37

We assign a 4-bit outcode to end points of line

P_1 : 0100 (if our 4-bits are TBRL then $B = 1$)

P_2 : 0010 (if our 4-bits are TBRL then $R = 1$)

Now P_1 and P_2 = 0 1 0 0

Logical AND 0 0 1 0

0 0 0 0 = Zero : *i.e.* line is partially visible.

Now slopr of line can be calculated as

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{3-(-2)}{3-1} = \frac{3+2}{2} = \frac{5}{2} = 2.5$$

Now $$\frac{5}{2} = \frac{y - y_1}{x - x_1}$$

$$\frac{5}{2} = \frac{1-(-2)}{x-1} \quad \text{[as } y = 1\text{] lower boundary}$$

$$\frac{5}{2} = \frac{3}{x-1}$$

$$5x - 5 = 6$$

$$5x = 6 + 5 = 11$$

$$x = \frac{11}{5}$$

So point $$P_1' = \left(\frac{11}{5}, 1\right).$$

Now for Intersecting point P'_2 with right boundary, can be calculated as

$$m = \frac{y - y_2}{x - x_2}$$

$$\frac{5}{2} = \frac{y-3}{2-3} \quad \text{(as } x = 2\text{, right boundary value)}$$

$$\frac{5}{2} = \frac{y-3}{-1}$$

$$-5 = 2y - 6$$

$$-5 + 6 = 2y$$

$$2y = 1$$

$$y = \frac{1}{2}$$

So the intersection point P_2' can be $\left(2, \frac{1}{2}\right)$.

So $$P_1' = \left(\frac{11}{5}, 1\right)$$

$$P_2' = \left(2, \frac{1}{2}\right)$$

Now the part $\overline{P_1' P_2'}$ of line $\overline{P_1 P_2}$ will be visible and $\overline{P_1 P_1'}$ and $\overline{P_2 P_2'}$ will be clipped off against the window.