2-D Clipping


THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Outline

- Clipping
- 2-D Clipping
- Clipping Algorithms
- Cohen Sutherland Algorithm

Clipping

- The procedure that identifies the portions of a picture that is either inside or outside of a specified region of space is refereed to as clipping.
- The region against which an object to be clipped is called a clip window or clipping window
  - Usually clip window is in rectangular shape

2-D Clipping

- Point Clipping
- Line Clipping
  - Cohen-Sutherland
  - Liang-Barsky
  - Nicholl-Li-Nichol
- Fill Area Clipping
  - Sutherland-Hodgeman
  - Weiler-Atherton
- Text Clipping

Clipping Algorithms

- A procedure that eliminates those portions of a picture that are either inside or outside a specified region of space is referred to as a **clipping algorithm** or simply **clipping**
- Everything outside the clipping window is eliminated from scene description (i.e. not scan converted) for efficiency
- Applications
  - Viewing pipeline
  - antialias object boundaries
  - construct objects using solid-modeling methods
  - manage a multiwindow environment, and to allow parts of a picture to be moved, copied, or erased in drawing and painting programs

•Point Clipping

•Given a clipping rectangle in standard position, we save a two-dimensional point **P** = *(x, y)* for display if

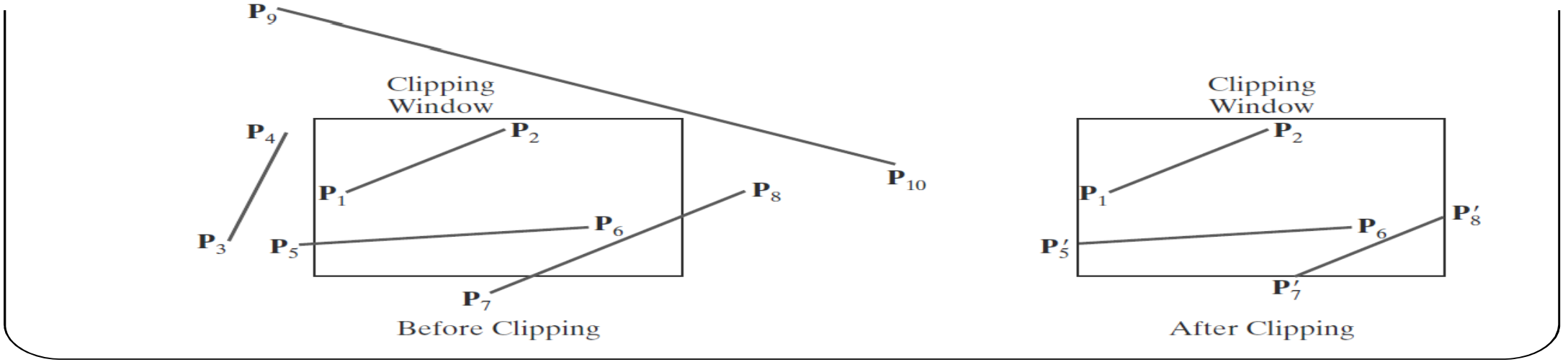
$$xw_{min} \le x \le xw_{max}$$

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

- If any of these inequalities is not satisfied, the point is clipped
  - not saved for display

Line Clipping

- A line-clipping algorithm processes each line in a scene to determine whether the entire line or any part of it is to be saved for display



•The lines are said to be interior to the clipping window if both end points are interior to the window i.e. P1 P2 in above fig.

• If both end points are exterior to window, the line is not completely exterior to window i.e. P7P8

•The lines which across one or more clipping boundaries require calculation of multiple intersection points to decide the visible portion of them. To minimize the intersection calculations and to increase the efficiency of clipping algorithms, initially completely visible and invisible lines are identified and then intersection points are calculated for remaining lines. There are many clipping algorithms. Let us focus few of them

Cohen-Sutherland Algorithm

This is one of the oldest and most popular line clipping algorithm developed by Dan Cohen and Ivan Sutherland. To speed up the processing this algorithm performs initial tests that reduce the number of intersections that must be calculated. This algorithm uses a four digit (bit) code to indicate which of nine regions contain the end point of line. The four bit codes are called **region codes** or **outcodes**. These codes identify the location of the point relative to the boundaries of the clipping rectangle as shown in the Fig. 5.9.

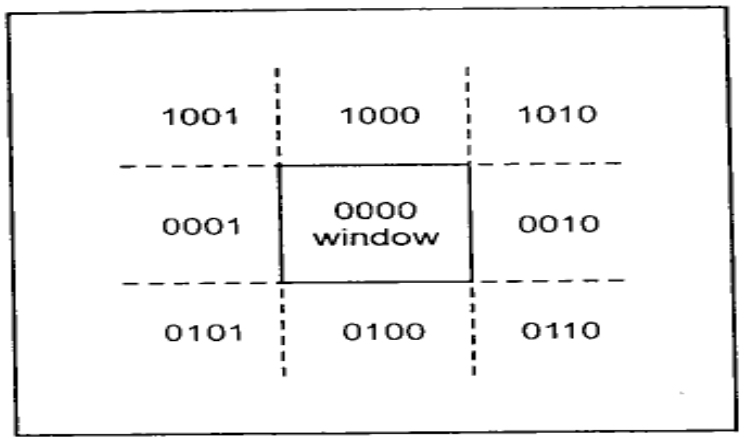


Fig. 5.9 Four-bit codes for nine regions

Each bit position in the region code is used to indicate one of the four relative coordinate positions of the point with respect to the clipping window : to the left, right, top or bottom. The rightmost bit is the first bit and the bits are set to 1 based on the following scheme :

Set Bit 1 – if the end point is to the **left** of the window

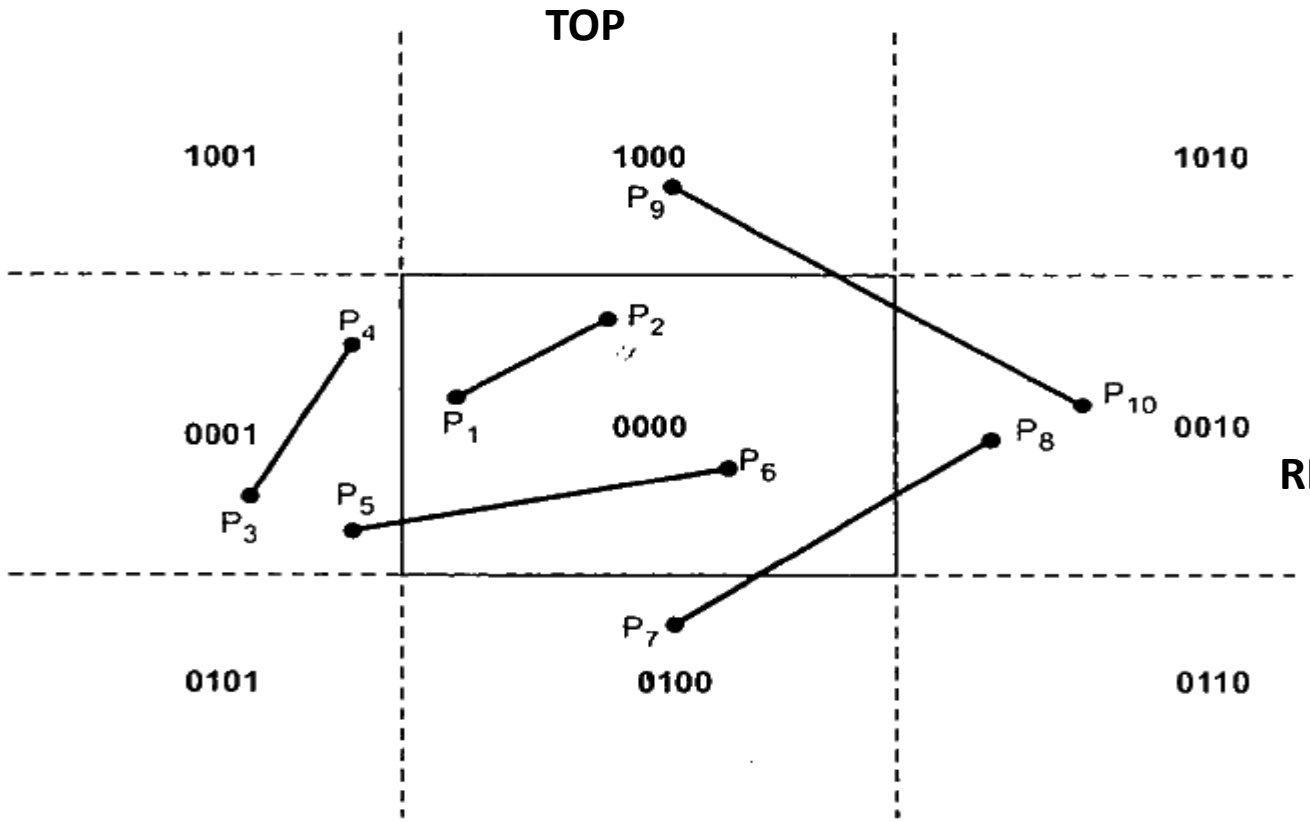
Set Bit 2 – if the end point is to the **right** of the window

Set Bit 3 – if the end point is **below** the window

Set Bit 4 – if the end point is **above** the window

Otherwise, the bit is set to zero.

Cohen-Sutherland Algorithm(Cont.....)



LEFT

RIGHT

TBRL = B4B3B2B1

BOTTOM

Cohen-Sutherland Algorithm(Cont.....)

Line	End Point Codes		Logical ANDing	Result
P_1 P_2	0000	0000	0000	Completely visible
P_3 P_4	0001	0001	0001	Completely invisible
P_5 P_6	0001	0000	0000	Partially visible
P_7 P_8	0100	0010	0000	Partially visible
P_9 P_{10}	1000	0010	0000	Partially visible

Table 5.1

The Sutherland - Cohen algorithm begins the clipping process for a partially visible line by comparing an outside endpoint to a clipping boundary to determine how much of the line can be discarded. Then the remaining part of the line is checked against the other boundaries, and the process is continued until either the line is totally discarded or a section is found inside the window.

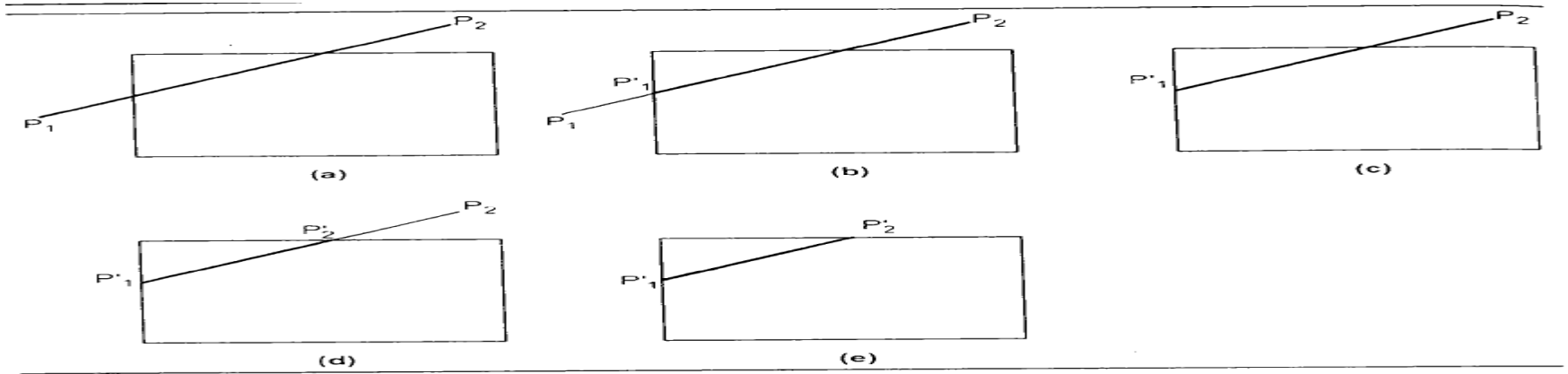


Fig. 5.12 Sutherland-Cohen subdivision line clipping

Cohen-Sutherland Algorithm(Cont.....)

Sutherland and Cohen subdivision line clipping algorithm :

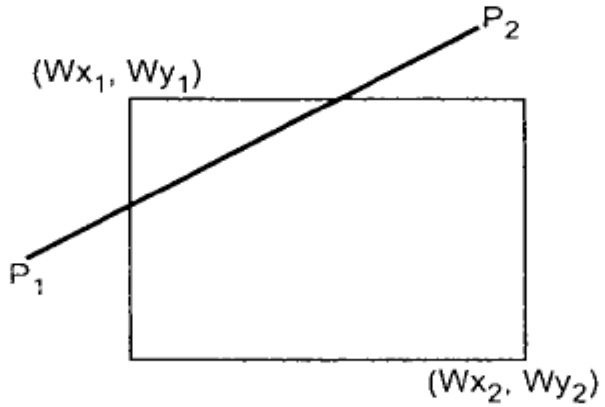


Fig. 5.13

1. Read two end points of the line say P_1 (x_1, y_1) and P_2 (x_2, y_2).
2. Read two corners (left-top and right-bottom) of the window, say (Wx_1, Wy_1 and Wx_2, Wy_2).
3. Assign the region codes for two endpoints P_1 and P_2 using following steps :

   Initialize code with bits 0000

   Set Bit 1 – if $(x < Wx_1)$

   Set Bit 2 – if $(x > Wx_2)$

   Set Bit 3 – if $(y < Wy_2)$

   Set Bit 4 – if $(y > Wy_1)$

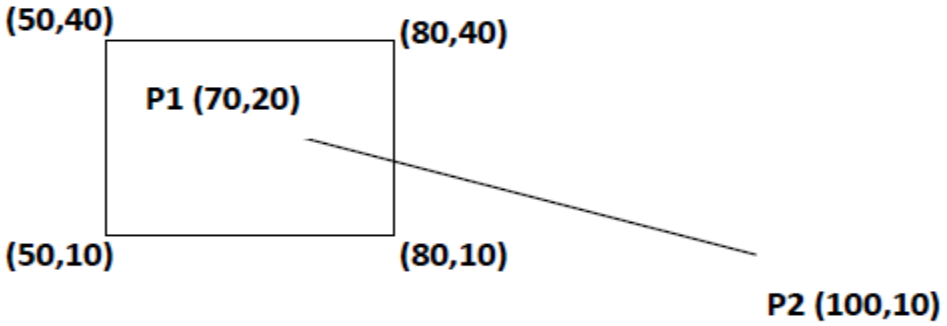
Cohen-Sutherland Algorithm(Cont.....)

4. Check for visibility of line $P_1 P_2$
   a) If region codes for both endpoints P_1 and P_2 are zero then the line is completely visible. Hence draw the line and go to step 9.
   b) If region codes for endpoints are not zero and the logical ANDing of them is also nonzero then the line is completely invisible, so reject the line and go to step 9.
   c) If region codes for two endpoints do not satisfy the conditions in 4a) and 4b)the line is partially visible.
5. Determine the intersecting edge of the clipping window by inspecting the region codes of two endpoints.
   a) If region codes for both the end points are non-zero, find intersection points P_1' and P_2' with boundary edges of clipping window with respect to point P_1 and point P_2, respectively
   b) If region code for any one end point is non zero then find intersection point P_1' or P_2' with the boundary edge of the clipping window with respect to it.
6. Divide the line segments considering intersection points.
7. Reject the line segment if any one end point of it appears outsides the clipping window.
8. Draw the remaining line segments.
9. Stop.

Cohen-Sutherland Algorithm(Cont.....)

Q53. Use the Cohen Sutherland algorithm to clip line P1 (70,20) and p2(100,10) against a window lower left hand corner (50,10) and upper right hand corner (80,40).

Ans:



Therefore, the intersections with the clipping boundaries of the window are given as :

Left : $x_L, y = m(x_L - x_1) + y_1 \quad ; \quad m \neq \infty$

Right : $x_R, y = m(x_R - x_1) + y_1 \quad ; \quad m \neq \infty$

Top : $y_T, x = x_1 + \left(\frac{1}{m}\right)(y_T - y_1) \; ; \quad m \neq 0$

Bottom : $y_B, x = x_1 + \left(\frac{1}{m}\right)(y_B - y_1) \; ; \quad m \neq 0$

Given , P1(70,20) and p2(100,10)
Window lower left corner = (50,10)
Window upper right corner = (80,40)

Now, we assign 4 bit binary outcode.
Point P1 is inside the window so the outcode of P1 = 0000 and the outcode for P2 = 0010.
Logical AND operation will give , 0000
0010
0000

Slope of the line P1P2 is $m = \frac{y2 - y1}{X2 - x1} = \frac{10 - 20}{100 - 70} = \frac{-10}{30} = \frac{-1}{3}$

Cohen-Sutherland Algorithm(Cont.....)

We, have to find intersection of line P1 P2 with right edge of window i.e P2 (x,y).
Here x=80 , we have to find the value of y.
We use the point P2(x2,y2) = P2(100,10)\

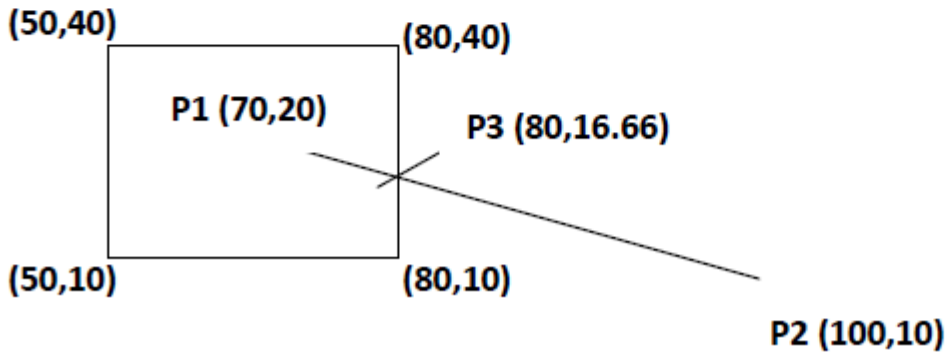
$$M = \frac{y - y2}{x - x2}$$

$$-1/3 = \frac{y - 10}{80 - 100}$$

$$y-10 = 20 / 3$$

$$y=16.66$$

thus, the intersection point P3 = (80, 16.66)
So, after clipping line P1P2 against the window, new line P1P3 with co ordinates P1(70, 20) and P3 (80, 16.66)



Cohen-Sutherland Algorithm(Cont.....)

Question1: Let R be the rectangular window whose lower left-hand corner is at $L(-3, 1)$ and upper right-hand corner is at $R(2, 6)$. Find the region codes for the endpoints in Fig. 5-17.

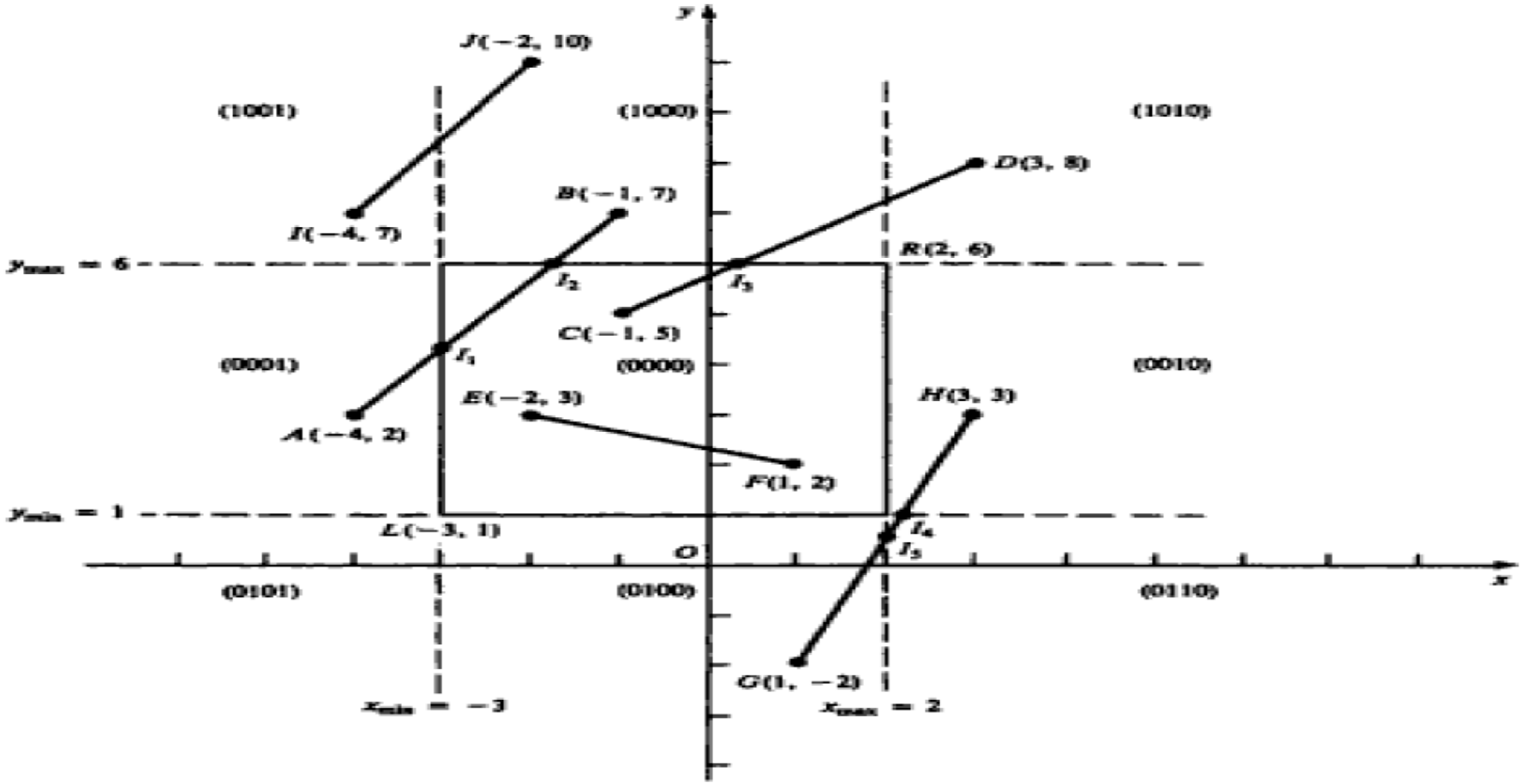


Fig. 5-17

Cohen-Sutherland Algorithm(Cont.....)

Solution:

The region code for point (x, y) is set according to the scheme

$$\text{Bit } 1 = \text{sign}(y - y_{max}) = \text{sign}(y - 6) \qquad \text{Bit } 3 = \text{sign}(x - x_{max}) = \text{sign}(x - 2)$$
$$\text{Bit } 2 = \text{sign}(y_{min} - y) = \text{sign}(1 - y) \qquad \text{Bit } 4 = \text{sign}(x_{min} - x) = \text{sign}(-3 - x)$$

Here

$$\text{Sign}(a) = \begin{cases} 1 & \text{if } a \text{ is positive} \\ 0 & \text{otherwise} \end{cases}$$

So

$A(-4, 2) \rightarrow 0001$ $\quad$ $F(1, 2) \rightarrow 0000$

$B(-1, 7) \rightarrow 1000$ $\quad$ $G(1, -2) \rightarrow 0100$

$C(-1, 5) \rightarrow 0000$ $\quad$ $H(3, 3) \rightarrow 0010$

$D(3, 8) \rightarrow 1010$ $\quad$ $I(-4, 7) \rightarrow 1001$

$E(-2, 3) \rightarrow 0000$ $\quad$ $J(-2, 10) \rightarrow 1000$

Cohen-Sutherland Algorithm(Cont.....)

Question 2: Use the Cohen –Sutherland for clipping the line segments in previous question (question1).

Solution:

In *clipping* $\overline{AB}$, the code for A is 0001. To push the 1 to 0, we clip against the boundary line $x_{min} = -3$. The resulting intersection point is $I_1(-3, 3\frac{2}{3})$. We clip (do not display) $\overline{AI_1}$ and work on $\overline{I_1B}$. The code for I_1 is 0000. The clipping category for $\overline{I_1B}$ is 3 since (0000) AND (1000) is (0000). Now B is outside the window (i.e., its code is 1000), so we push the 1 to a 0 by clipping against the line $y_{max} = 6$. The resulting intersection is $I_2(-1\frac{1}{3}, 6)$. Thus $\overline{I_2B}$ is clipped. The code for I_2 is 0000. The remaining segment $\overline{I_1I_2}$ is displayed since both endpoints lie in the window (i.e., their codes are 0000).

For *clipping* $\overline{CD}$, we start with D since it is outside the window. Its code is 1010. We push the first 1 to a 0 by clipping against the line $y_{max} = 6$. The resulting intersection I_3 is $(\frac{1}{3}, 6)$ and its code is 0000. Thus $\overline{I_3D}$ is clipped and the remaining segment $\overline{CI_3}$ has both endpoints coded 0000 and so it is displayed.

For *clipping* $\overline{GH}$, we can start with either G or H since both are outside the window. The code for G is 0100, and we push the 1 to a 0 by clipping against the line $y_{min} = 1$. The resulting intersection point is $I_4(2\frac{1}{3}, 1)$, and its code is 0010. We clip $\overline{GI_4}$ and work on $\overline{I_4H}$. Segment $\overline{I_4H}$ is not displayed since (0010) AND (0010) = 0010.

Note: The coordinates of intersection points be calculated as:

$$\begin{cases} x_i = x_{min} \text{ or } x_{max} \\ y_i = y_1 + m(x_i - x_1) \end{cases} \quad \text{if the boundary line is vertical}$$

or

$$\begin{cases} x_i = x_1 + (y_i - y_1)/m \\ y_i = y_{min} \text{ or } y_{max} \end{cases} \quad \text{if the boundary line is horizontal}$$

where $m = (y_2 - y_1)/(x_2 - x_1)$ is the slope of the line.

Summary

- One of the earliest algorithms for fast line clipping
- Processing time is reduced by performing more tests before proceeding to the intersection calculations

Resources

- https:// iq.opengenus.org/cohen-sutherland-line-clipping-algorithm/
- https:// https://www.tutorialandexample.com/line-clipping/
- https://www.cs.helsinki.fi/group/goa/viewing/leikkaus/lineClip.html