4.1 Introduction to clipping

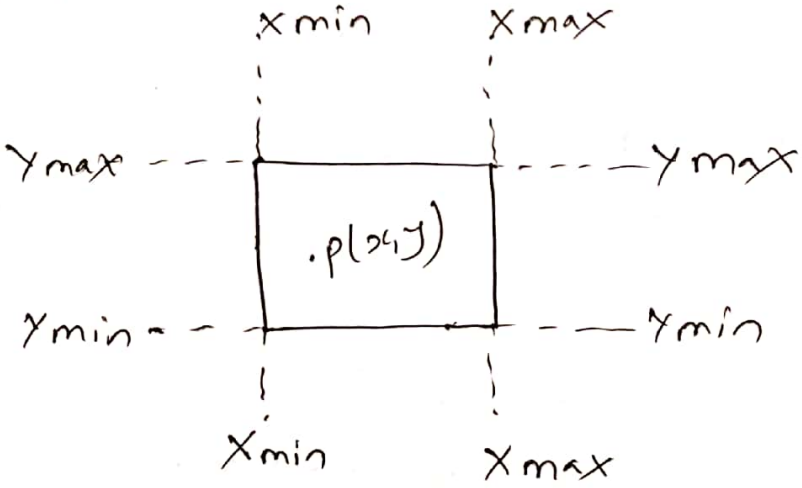
Clipping is a fundamental concept in computer graphics that involves determining which parts of a graphical object lie within a specified region (usually a rectangle or viewport).

The portion left outside the region of the window in computer graphics is called the clipped portion and the process of displaying inside image of the window is called clipping.

Types of clipping

(i) point clipping
(ii) Line clipping
(iii) polygon clipping
(iv) text clipping
(v) curve clipping

(i) Point clipping :- point clipping determines whether a point lies inside or outside the clipping window.



Algorithm:

1. Define the clipping window boundaries (e.g xmin, xmax, ymin, ymax).
2. for a given point (x,y), check if:

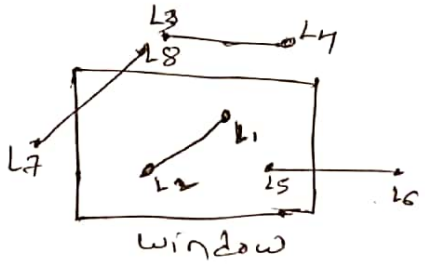
$x_{min} \leq x \leq x_{max}$

$y_{min} \leq y \leq y_{max}$

3. If both coordinates are true, the point is inside the clipping window; otherwise, it is outside.

(ii) **Line clipping** :- Line clipping involves determining which parts of a line segment lie within the clipping window.

Removing lines/portions outside of the clipping window.



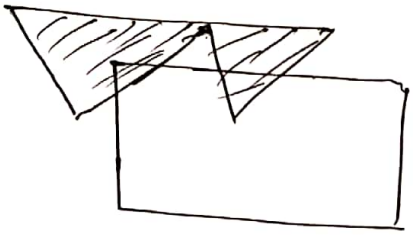
(i) Visible :- both end points inside window
(ii) Not visible :- both points outside of window
(iii) partially visible :- 1 point inside and 1 point outside the window

one of the most popular algorithms for line ~~drawing~~ clipping is the Cohen-Sutherland Algorithm.

(iii) polygon clipping :- polygon clipping involves determing which parts of a polygon lie within the clipping window. The Sutherland-Hodgeman Algorithm is widely used method for this purpose.

Polygon Clipping

Done by sutherland - Hodgeman Algorithm.



Before clipping

After clipping

Cohen sutherland Line Clipping Algorithm

TBRL followed (top, bottom, right, and left).

Sutherland line clipping Algorithm has 9 regions on the screen. out of them, one region is assigned for the window and the rest 8 regions are around it given by 4 digit binary. The division of the regions are based on x_{max}, y_{max} and x_{min}, y_{min}. The central part is the viewing region or window, all the lines which lie within this region are completely visible.

Generate TBRL (Region code) → 4 bit code
(TOP Bottom Left Right)

if
- $x < x_{min}$ → Left of window
- $x > x_{max}$ → Right of window
- $y < y_{min}$ → Bottom of window
- $y > y_{max}$ → TOP of the window

Cohen-sutherland Line Clipping Algorithm

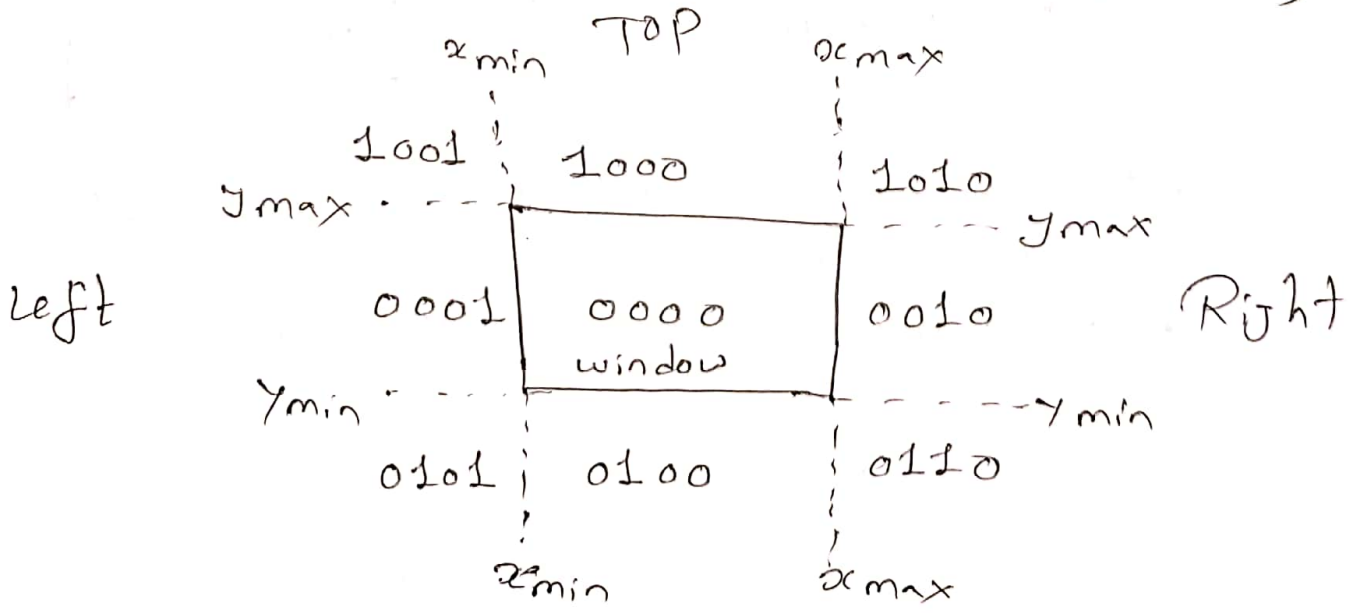
The Cohen-sutherland algorithm is used to clip a line inside a rectangular window. It checks whether a line is

(i) Completely inside,
(ii) Completely outside,
OR
(iii) partially inside and clips it accordingly.

Clipping window: clipping window is defined by four boundaries:

(i) Left : x_{min}
(ii) Right : x_{max}
(iii) Bottom : y_{min}
(iv) TOP : y_{max}

Region codes: Each point is assigned a 4-bit binary code based on its position relative to the window. (TBRL)



TBRL-Region codes

Algorithm

1. Compute Region codes of both endpoints.
2. Check Trivial Cases:
   - if both codes are 0000, the line is inside (Accepted)
   - If AND of both codes is nonzero, the line is outside (Rejected).
3. Clip the line if paritially inside:
   - Choose the endpoint outside.
   - find intersection with the nearest boundary using formulas.

   To calculate intersections with the boundaries:

for Vertical (Left $x = x_{min}$ or Right $x = x_{max}$) boundaries

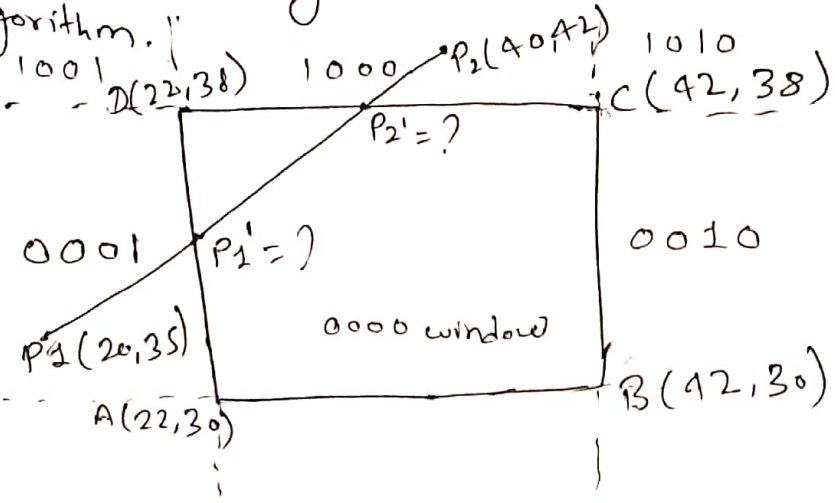
$$y = y_1 + m * (x - x_1)$$ where $x = x_{min}$ OR $x = x_{max}$

for horizontal boundaries (TOP $y = y_{max}$ or Bottom $y = y_{min}$)

$$x = x_1 + \frac{1}{m} * (y - y_1)$$ where $y = y_{min}$ OR $y = y_{max}$

   - Replace the endpoints with the intersection point and recompute the code.
   - Repeat until the line is fully inside.

In given figure coordinates of windows ABCD, clip the Line P1, P2 using Cohen sutherland line clipping Algorithm.



1. Compute Region code of both endpoints using TBRL find region code

P1 → 0001

P2 → 1000 perform AND operation

0000 So, trivially accepted Now,

find intersections point of P_1' and P_2''

for P_1', find $m(\text{slope}) = \frac{y_2 - y_1}{x_2 - x_1} = \frac{42-35}{40-20} = 0.35$

$y = y_1 + m * (x - x_1)$ where $x = x_{min} / x_{max}$

$= 35 + 0.35 * (22-20)$

$= 35 + 0.35 * 2$

$= 35 + 0.7$

$= 35.7$

Hence, P_1' intersection point is (22, 35.7)

for P_2', find $m(\text{slope}) = \frac{x_2 - x_1}{y_2 - y_1}$ | $\frac{1}{m}$ use directly in the formula

$x = x_1 + \frac{1}{m} * (y - y_1)$

$= 20 + 2.857 * (38-35)$

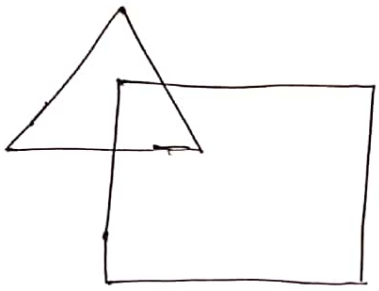
$= 20 + 8.571$

$= 28.57$

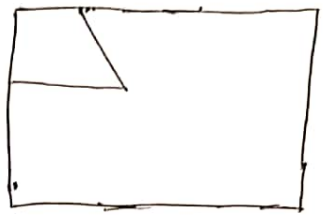
Hence, P_2' intersection point is (28.57, 38).

Polygon clipping :-

Polygon clipping is the process of removing certain parts of a polygon that fall outside of a specified rectangular boundary, (window). The aim is to display only the polygon that is within the window.

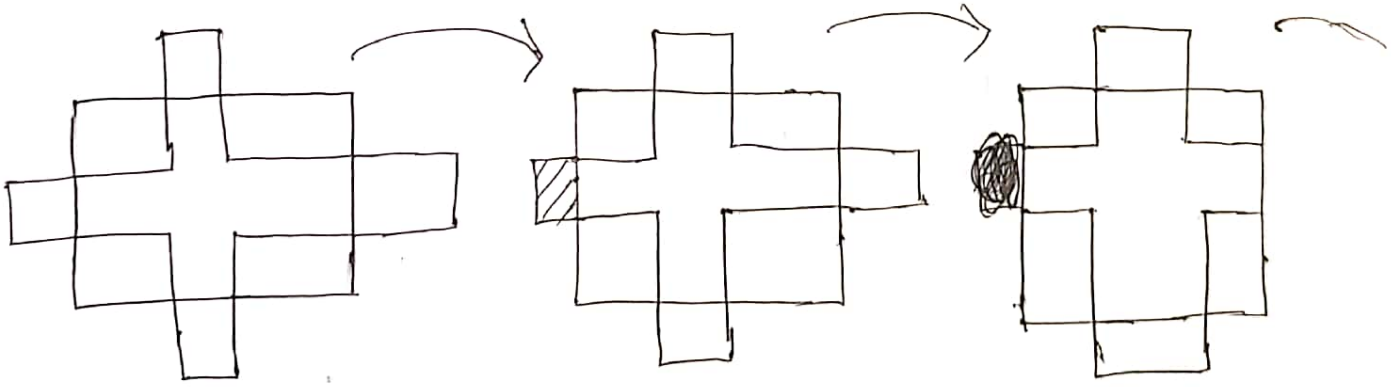


Polygon before clipping



Polygon After clipping

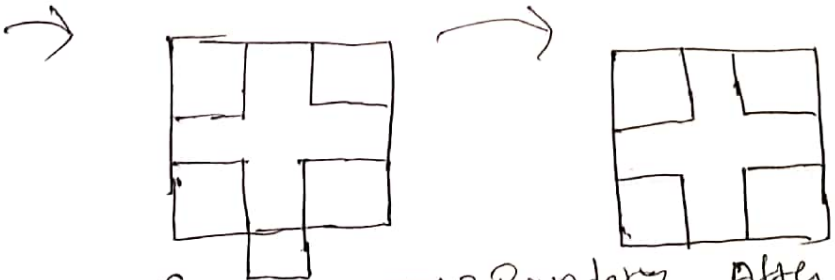
Sutherland - Hodgeman polygon Algorithm is popular polygon clipping Algorithm.



Before clipping

After clipping left boundry

After clipping Right boundry

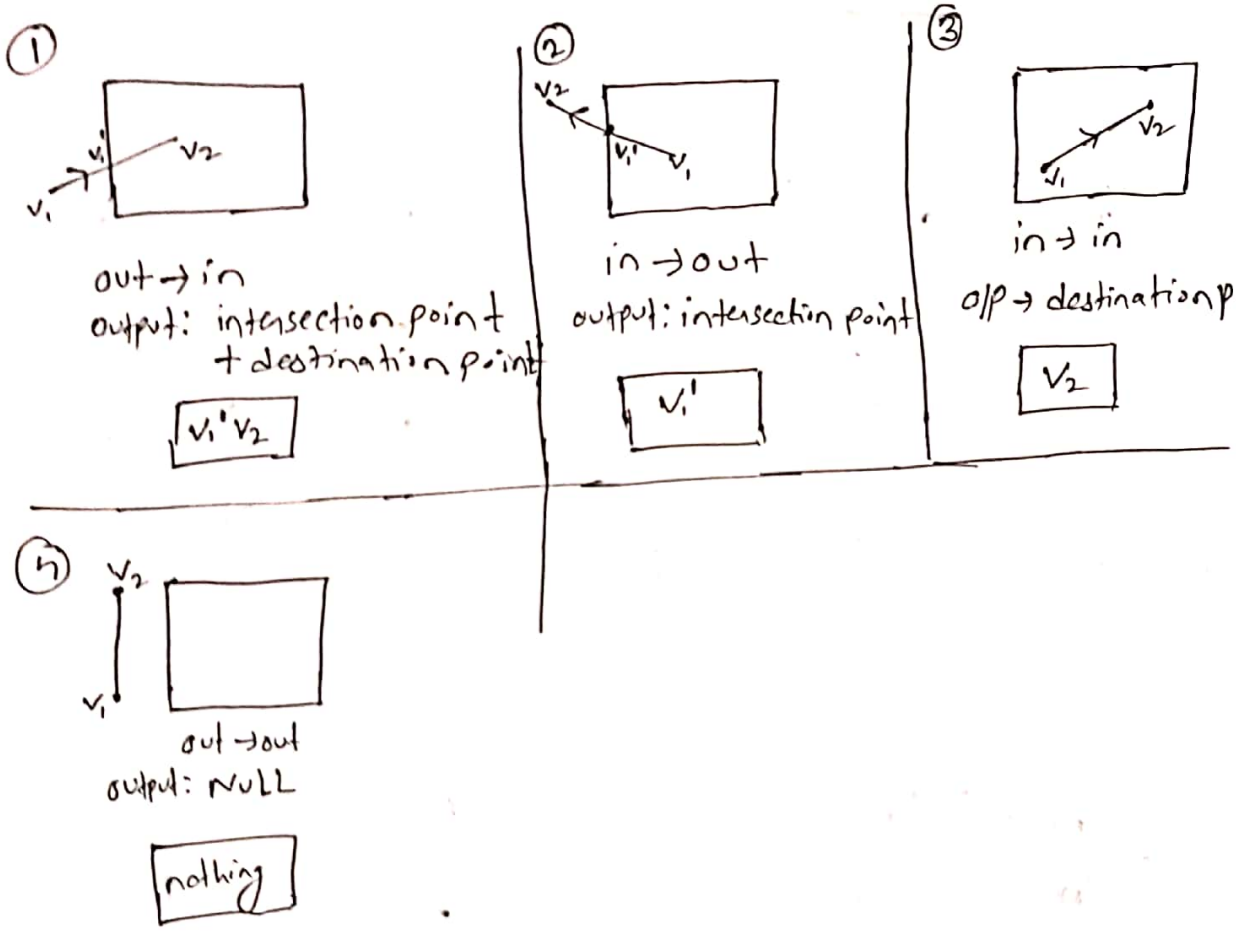


After clipping TOP Boundary

After clipping Bottom Boundary

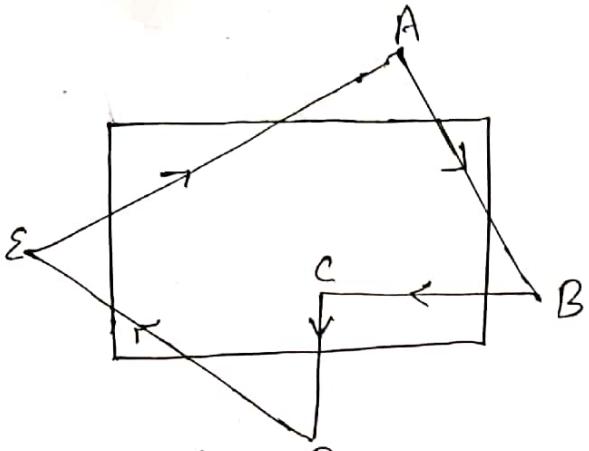
Sutherland-Hodgman polygon clipping Algorithm follow divide-conquer strategy.

Rules for clipping the polygon edge:



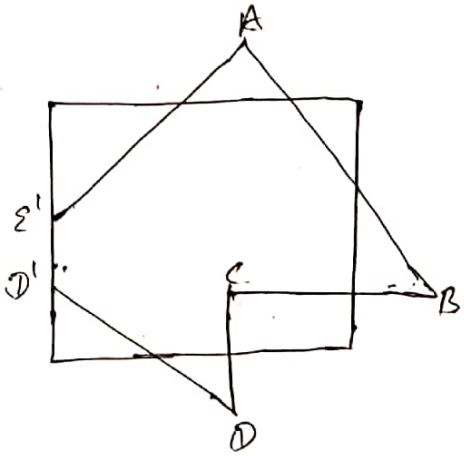
Steps of sutherland Hodgeman polygon clipping Algorithm

① Read the coordinates of all vertices of the Polygon.
② Read the coordinate of the clipping window.
③ Consider the left edge of the window.
④ Compare the vertices of the edge of the polygon, individually with the clipping plane.
⑤ save the resulting intersections and the vertices according to the rules
⑥ Repeat step ④, ⑤ for the remaining edges of the clipping window.
⑦ Each time pass the resultant list of vertices to the next edge of the clipping window.
⑧ Stop.



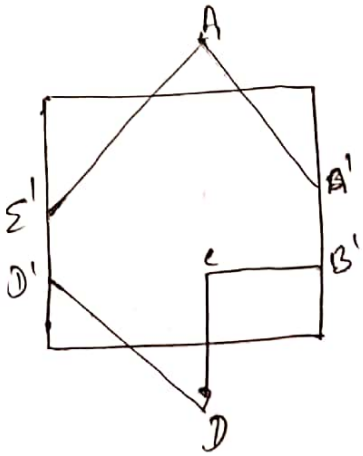
① clipping Against left edge

Vertex	Rule	output saving
AB	in → in	B
BC	in → in	C
CD	in → in	D
DE	in → out	D'
EA	out → in	E' A



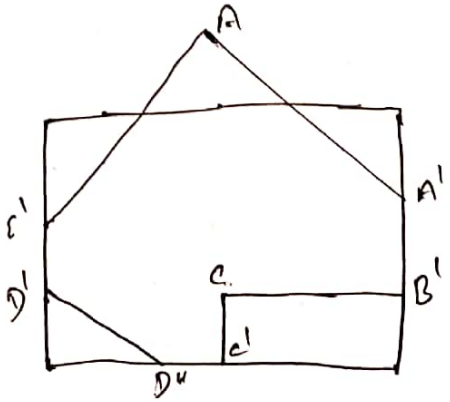
② clipping Against Right edge

vertex	Rule	output saving
AB	in → out	A'
BC	out → in	B' C
CD	in → in	D
DD'	in → in	D'
D'E'	in → in	E'
E'A	in → in	A



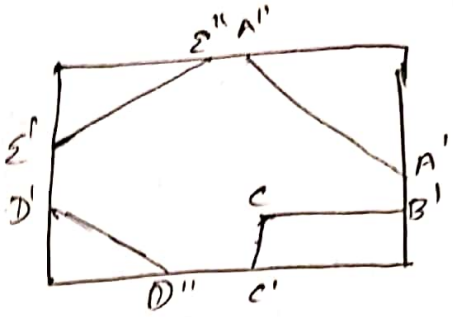
③ clipping Against Bottom edge:

vertex	Rule	output
AA'	in → in	A'
A'B'	in → in	B'
B'C	in → in	C
CD	in → out	C'
DD'	out → in	D'' D'
D'E	in → in	E'
E'A	in → in	A



clipping Against the top edge:

vertex	Rule.	output saving
AA'	out → in	A''A'
A'B'	in → in	B'
B'C	in → in	C
CC'	in → in	C'
C'D''	in → in	D''
D''D'	in → in	D'
D'E'	in → in	E'
E'A	in → out	E''



for objective-Question

Points to know

Line clipping Algorithms are:

1. Cohen-Sutherland Line clipping Algorithm
2. Liang-Barsky Line clipping Algorithm
3. Nicholl-Lee-Nicholl (NLN) Algorithm
4. Cyrus-Beck Line clipping Algorithm

polygon clipping Algorithms are:

1. Sutherland-Hodgman polygon clipping Algorithm
2. weiler-Atherton polygon clipping Algorithm
3. Greiner-Hormann Algorithm
4. Vatti clipping Algorithm
5. Martinez-Rueda clipping Algorithm