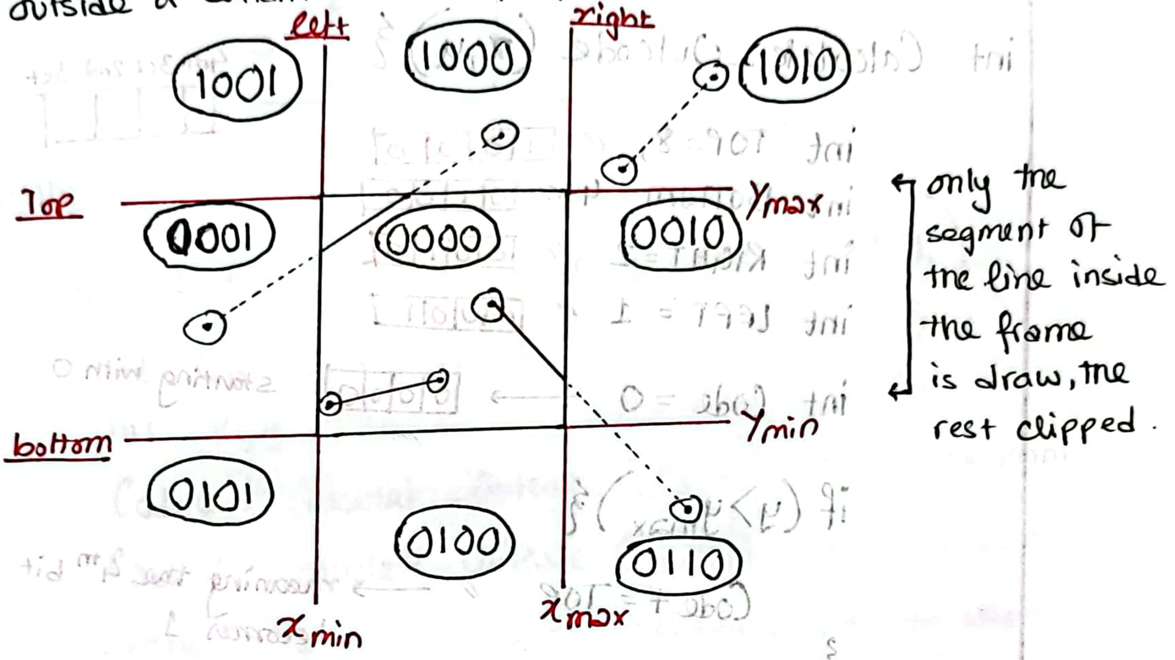
LINE CLIPPING ALGORITHMS :

COHEN-SUTHERLAND

LECTURE 7

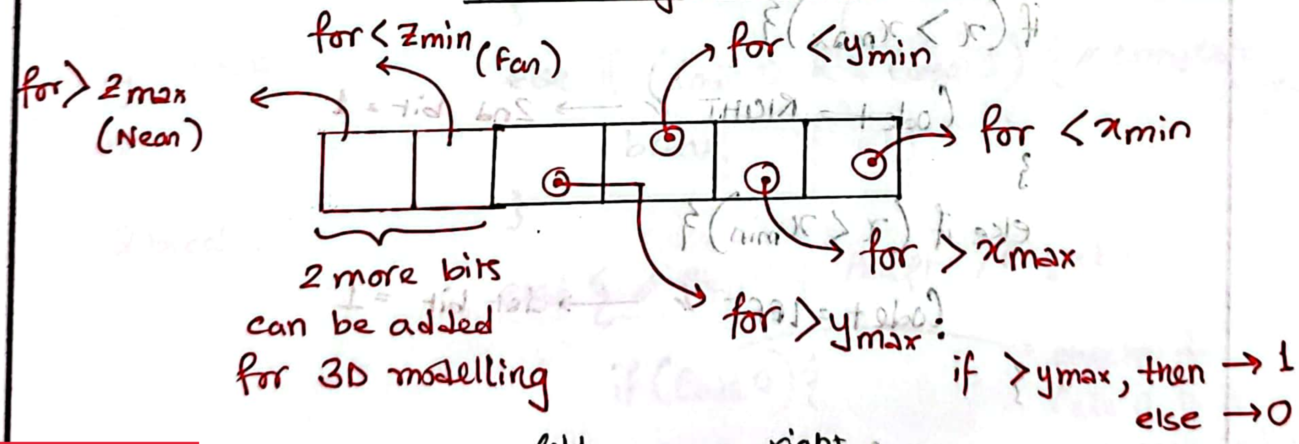
⊞ line clipping algorithms are used to clip lines that lie outside a certain window of operation.



only the segment of the line inside the frame is draw, the rest clipped.

In Cohen-Sotherland we need to determine a region Outcode to determine if the line crosses the window or not.

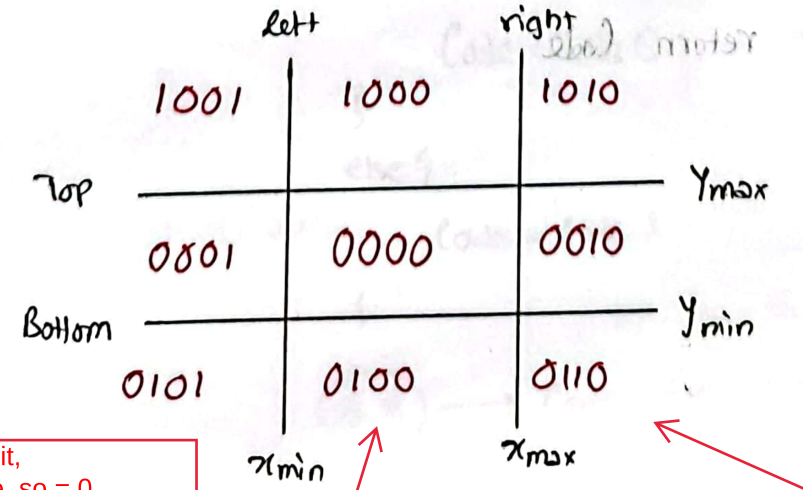
4 bit Region Outcode: (For a 2D plane)



if > ymax, then → 1
else → 0

4 bit are-
top bottom right left

for each bit, if it is true then the bit is 1, else 0



for 0110 bit,
top = false, so = 0
bottom = true, so = 1
right = true, so = 1
left = false, so = 0

for 0110 bit,
top = false, so = 0
bottom = true, so = 1
right = false, so = 0
left = false, so = 0

(Pseudo Code):

1) Calculating the Outcode:

4th 3rd 2nd 1st

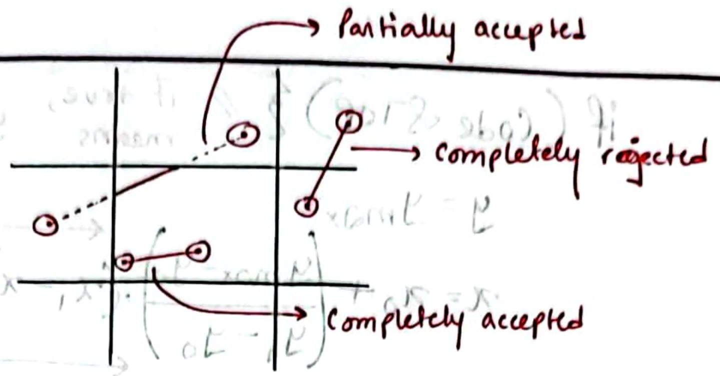

```
int Calculate_Outcode (x,y) {

    int TOP = 8,    // 1|0|0|0
    int BOTTOM = 4  // 0|1|0|0
    int RIGHT = 2   // 0|0|1|0
    int LEFT = 1    // 0|0|0|1

    int Code = 0  ⟶ 0|0|0|0   starting with 0

    if (y > y_max) {
        Code + = TOP     // ⟶ meaning the 4th bit becomes 1
    }
    else if (y < y_min) {
        Code + = BOTTOM  // ⟶ 3rd bit becomes 1
    }
    if (x > x_max) {
        Code + = RIGHT   // ⟶ 2nd bit = 1
    }
    else if (x < x_min) {
        Code + = LEFT    // ⟶ 1st bit = 1
    }

    return Code
}
```



Main Algorithm

```
Void Cohen_Suthenland (int x0, int y0, int x1, int y1)
    int Code, Code0, Code1
    int x, y
    Code0 = Calculate_Outcode (x0, y0) → starting point
    Code1 = Calculate_Outcode (x1, y1) → ending point.
    While (1){
        if (Code0 || Code1 == 0){ // complete Accept
            drawline (x0, y0, x1, y1)
            break
        }
        else if (Code0 && Code1){ // complete reject
            break
        }
        else { // Partially Accept /Reject.
            if (Code0){
                Code = Code0
            }
            else{
                Code = Code1
            }
```

We have to calculate this for every line which parameter is start point (x0, y0) and ending point (x1, y1)

Bitwise OR operation

Bitwise AND operation

checks if Code 0 is a non-zero value,
✓ 0 = false
✓ 1 = True.

3 conditions to get out of this infinite loop

1. Completely accept → drawline & break
2. Completely reject → do nothing & break
3. Partially accept → Clip line, drawline & break.

if starting and ending both exist in the 0000 outcode, this line is accepted

starting and ending point outcode bitwise multiplication or AND operation is 1, then that line is rejected.
Because, in same position of both starting and ending point outcode is 1, that means that line never intercept 0000 outocde. Example 1001 and 0101, 4th position code is true or 1 for both for the outcode. rejected

choosing the starting point between two points for cliping.
if any of this two point don't exist in 0000 outcode , that will selected as new starting point

** → Continued in next Page.

```
if (Code & TOP) { // if true, means y > ymax
    y = ymax
    $x = x_0 + \left(\frac{y_{max} - y_0}{y_1 - y_0}\right) \cdot (x_1 - x_0)$
}
else if (Code & BOTTOM) {
    y = ymin
    $x = x_0 + \left(\frac{y_{min} - y_0}{y_1 - y_0}\right) \cdot (x_1 - x_0)$
}
else if (Code & RIGHT) {
    x = xmax
    $y = y_0 + \left(\frac{x_{max} - x_0}{x_1 - x_0}\right) \cdot (y_1 - y_0)$
}
else if (Code & LEFT) {
    x = xmin
    $y = y_0 + \left(\frac{x_{min} - x_0}{x_1 - x_0}\right) \cdot (y_1 - y_0)$
}
if (Code == Code0) {
    x0 = x , y0 = y
    Code0 = Calculate_Outcode (x0, y0)
}
else { // means we took Code 1
    x1 = x , y1 = y
    Code 1 = Calculate_Outcode (x1, y1)
}
}
}
}
```

were basically using parametric equations of line.

the x & y we have got is the new x_0 & y_0 after clipping

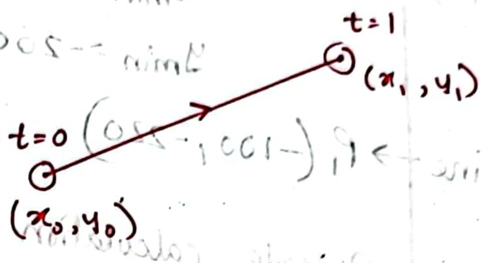
This equation is used to find the new end point which intercept.

First check for the top then bottom, then right, then left and find the the end point become 0000 outcode. that means it is clipped

At last update the points and outcode for every clip

Parametric equations of a line:

- $x = x_0 + t\,(x_1 - x_0)$
- $y = y_0 + t\,(y_1 - y_0)$



~~$y - y$~~

~~$y = y_0 +$~~

- $y = y_0 + \dfrac{x - x_0}{x_1 - x_0} \cdot (y_1 - y_0)$

- $x = x_0 + \dfrac{y - y_0}{y_1 - y_0} \cdot (x_1 - x_0)$

for y, $t = \dfrac{x - x_0}{x_1 - x_0}$ → refers to the position of x.

for x, : $t = \dfrac{y - y_0}{y_1 - y_0}$ → refers to the position of y.

for known value of x we can get y.

OR

for known value of y we can get x.

Q1) Window → $x_{min} = -250$, $x_{max} = 250$

$y_{min} = -200$, $y_{max} = 200$

line → $P_1(-100, -220)$ & $P_2(300, -210)$

ANS: Outcode calculation :

Outcode for P1 → 0100

Outcode for P2 → 01●10

OR operation :

$$\begin{array}{r} 0100 \\ 011●0 \\ \hline 011●0 \end{array}$$

→ not 0 , so not completely accepted.

AND operation :

$$\begin{array}{r} 0100 \\ 011●0 \\ \hline 0100 \end{array}$$

→ non-zero value so, the line is completely rejected!

The way of choosing outcode from geometery coordinate

Check if the point is above the top boundary of the clipping window:
If y > y_max, set first outcode bit to 1.
else 0

Check if the point is below the bottom boundary of the clipping window:
If y < y_min, set second outcode Bit 2 to 1.
else 0

Check if the point is to the right of the right boundary of the clipping window:
If x > x_max, set third outcode Bit 3 to 1.
else 0

Check if the point is to the left of the left boundary of the clipping window:
If x < x_min, set fourth outcode Bit 4 to 1.
else 0

Q2) Clip Region $(-100, -120)$ to $(150, 200)$

x_{min}. y_{min} x_{max} y_{max}

line given : $P_1(-125, 260)$ to $P_2(195, -140)$

x_0 y_0 x_1 y_1

Ans Outcode for P_1 = 1001

" " P_2 = 0110

OR operation = 1111 → so, not completely accepted as non-zero value.

—x—

$$\begin{array}{r} 1001 \\ 0110 \\ \hline =0000 \end{array}$$

AND op, → so, not completely rejected as value is zero

—x—

Hence, partially accepted.

Step 1 (taking P1)

$$\begin{array}{r} 1001 \\ 1000 \\ \hline 1000 \end{array}$$

For TOP : → Accepted

$y = y_{max} = 200$

$x = -125 + \frac{200-260}{-140-260} \cdot (195+125)$

$= -77$

new $P_1(-77, 200)$

code of P_1 → (0000) DONE

Step 2 (taking P2)

$$\begin{array}{r} 0110 \\ 1000 \\ \hline 0000 \end{array}$$

FOR TOP : → Not Accepted

$y = y_{min} = -120$

$x = -125 + \frac{-120-260}{-140-260} \cdot (195+125)$

$= 179$

$$\begin{array}{r} 0110 \\ 0100 \\ \hline 0100 \end{array}$$

FOR BOTTOM : → Accepted

after clipping,
0 = not accepted
1 = accepted

new $P_2(179, -120)$

code of P_2 → (0010) → the code is still not (0000) So, proceed to step 3

Step 3

FOR RIGHT :
$$\begin{array}{l} 0010 \\ 0010 \\ \hline 0010 \end{array} \rightarrow \text{Accepted}$$

$x = x_{max} = 150$

$y = 260 + \frac{150+125}{195+125} \cdot (-140 - 260)$

$y = -83.75$

Now $P_2 (150, -83.75)$

code for $P_2 \longrightarrow (0000) \longrightarrow$ <u>DONE</u>

Drawline $\longrightarrow (-77, 200)$ to $(150, -83.75)$