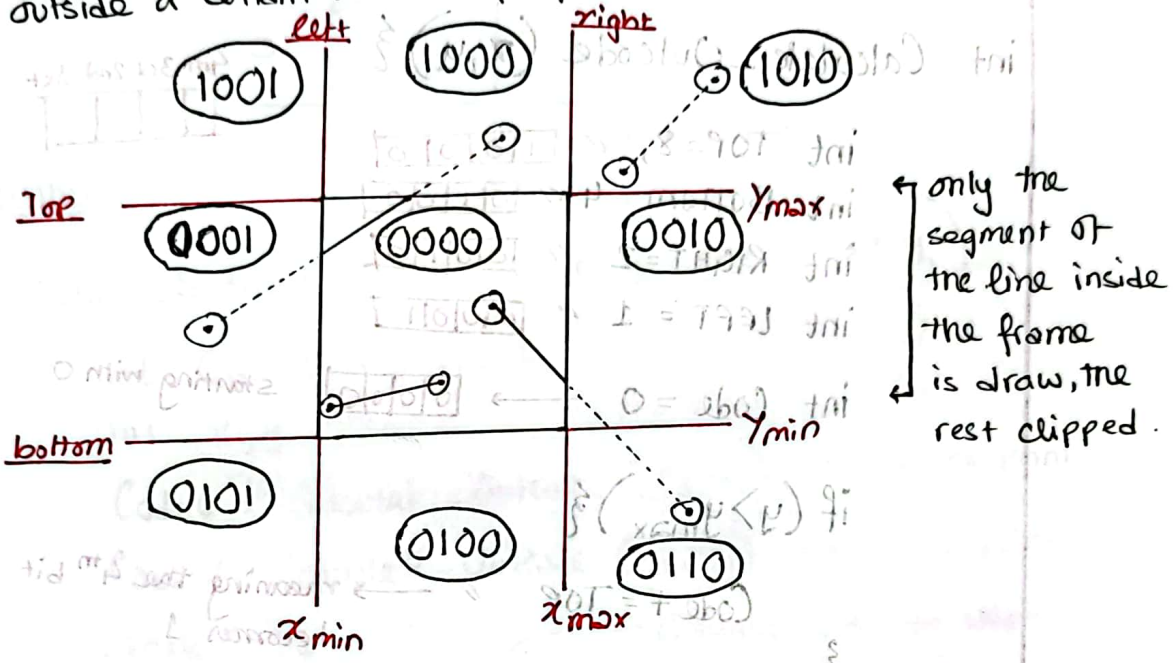
LINE CLIPPING ALGORITHMS:

COHEN-SUTHERLAND

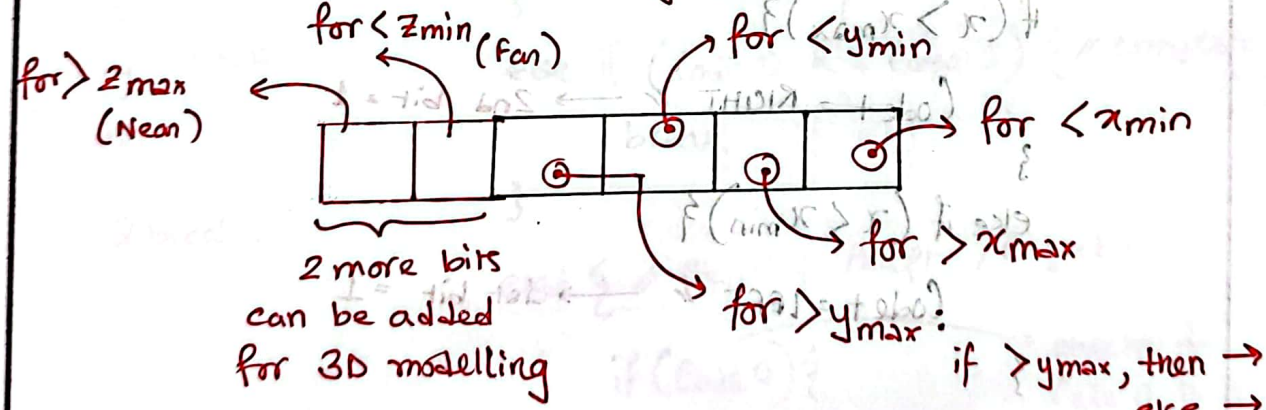
LECTURE 7

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



In Cohen-Sutherland 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)



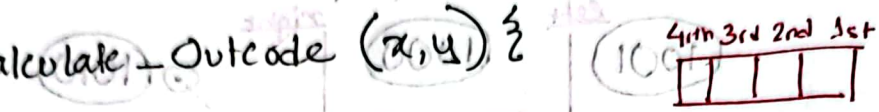
if > ymax, then → 1
else → 0

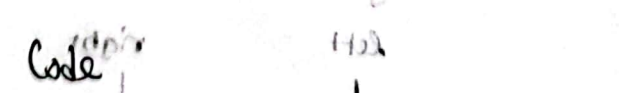
(Pseudo Code):

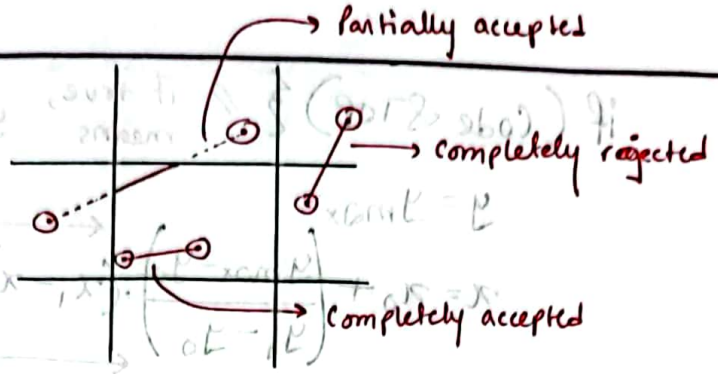
1) Calculating the Outcode:

```
int CalculateOutcode (x,y) {
    int TOP = 8,    //  1000
    int BOTTOM = 4  //  0100
    int RIGHT = 2   //  0010
    int LEFT = 1    //  0001
    int Code = 0    ->  0000   starting with 0

    if (y > ymax) {
        Code += TOP  //  -> meaning the 4th bit
    }                       will become 1
    else if (y < ymin) {
        Code += BOTTOM //  -> 3rd bit becomes 1
    }
    if (x > xmax) {
        Code += RIGHT  //  -> 2nd bit = 1
    }
    else if (x < xmin) {
        Code += LEFT   //  -> 1st bit = 1
    }
    return Code
}
```







Main Algorithm

```
Void Cohen_Sutherland (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) {                          → Bitwise OR operation
        if (Code0 || Code1 == 0) { // complete Accept
            drawline (x0, y0, x1, y1)
            break
        }
        else if (Code0 && Code1) { // complete reject   → Bitwise AND operation
            break
        }
        else { // Partially Accept / Reject.
            if (Code0) {                 → checks if Code 0 is a non-zero value,
                Code = Code0                  ✓ 0 = false
            }                                 ✓ 1 = True.
            else {
                Code = Code1
            }
            (**) → Continued in next Page.
```

3 conditions to get out of this infinite loop

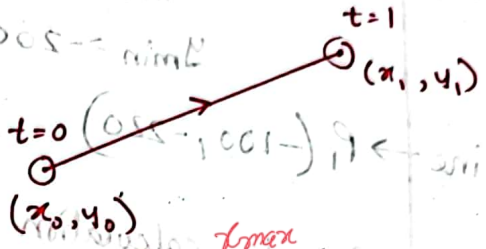
1. Completely accept → drawline & break
2. Completely reject → do nothing & break
3. Partially accept → Clip line, drawline & break.

```
if (Code & TOP) {   // if true, means y > ymax
    y = ymax
    x = x0 + ((ymax - y0) / (y1 - y0)) * (x1 - x0)
}
else if (Code & BOTTOM) {
    y = ymin
    x = x0 + ((ymin - y0) / (y1 - y0)) * (x1 - x0)
}
else if (Code & RIGHT) {
    x = xmax
    y = y0 + ((xmax - x0) / (x1 - x0)) * (y1 - y0)
}
else if (Code & LEFT) {
    x = xmin
    y = y0 + ((xmin - x0) / (x1 - x0)) * (y1 - y0)
}
if (Code == Code0) {
    x0 = x , y0 = y      // the x & y we have got is the new x0 & y0 after clipping
    Code0 = Calculate_Outcode (x0, y0)
}
else {   // means we took Code 1
    x1 = x , y1 = y
    Code1 = Calculate_Outcode (x1, y1)
}
}
```

were basically using parametric equations of line.

Parametric equations of a line:

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



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

  x_{max} → refers to the position of x.

  $t = \dfrac{x - x_0}{x_1 - x_0}$

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

  y_{max} → refers to the position of y.

  $t = \dfrac{y - y_0}{y_1 - y_0}$

→ for known value of x we can get y.

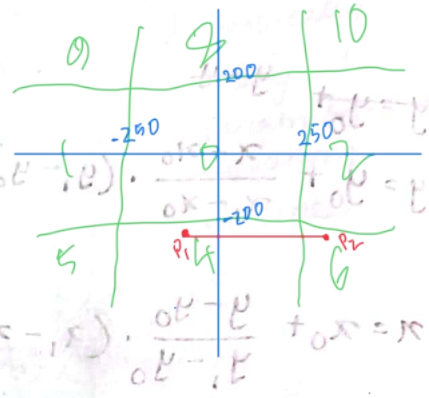
OR

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 → 0110

OR operation:

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

→ not 0, so not completely accepted.

AND operation:

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

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

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.

AND op, 1001 & 0110 = 0000 → so, not completely rejected as value is zero

Hence, partially accepted.

Step 1 (taking P1)

for TOP : 1001 & 1000 = 1000 → 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)

FOR TOP : 0110 & 1000 = 0000 → Not Accepted

$y = y_{min} = -120$

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

$= 179$

FOR BOTTOM : 0110 & 0100 = 0100 → 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 : $\begin{array}{r} 0010 \\ 0010 \\ \hline 0010 \end{array}$ → Accepted

$x = x_{max} = 150$

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

$y_2 = -83.75$

Now $P_2(150, -83.75)$

code for P_2 → (0000) → DONE

Drawline → (−77, 200) to (150, −83.75)