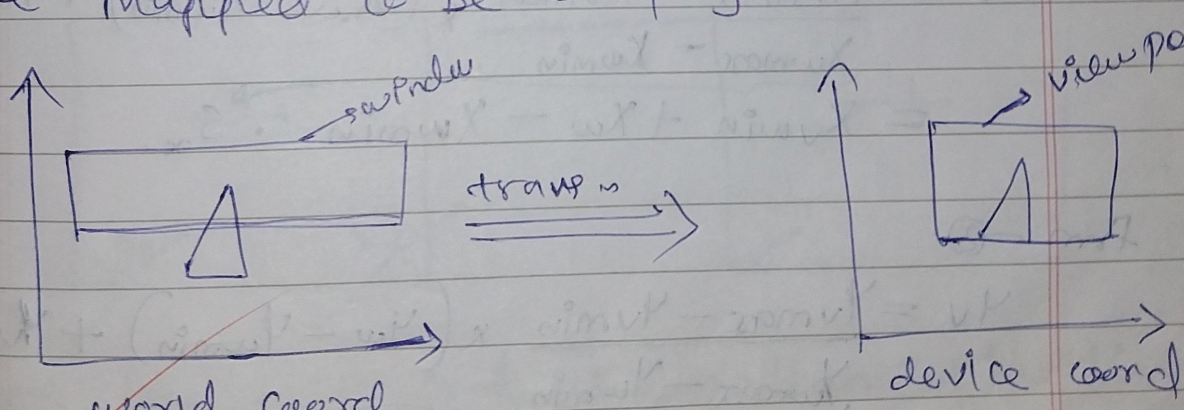
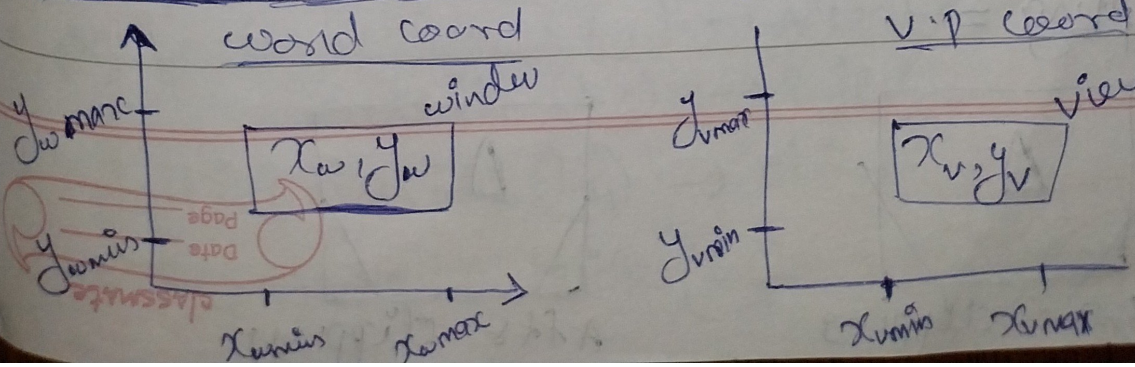
Module - IV

window to viewport trans^n

* **window** : Area on world coord selected for display / simply we can say window define wht to be displayed.
* **viewport** : Area / device coord where graphics ist to be display / simply we can say it define where to be displayed.
* **w-to viewport trans^n** : process of transforming a world coord obj to device coord obj inside the window are mapped to the viewpoint which is the area on the screen where world coord are mapped to be displayed.



* **mathematical calcul^n of w-to-viewport trans^n**



* It may be possible that the size of v.p is much >/< than the window.
* Here, we hve to rse / dse the size of window according to the v.p & for this we need some mathematical cal.
* For every point (x_w, y_w) in the window there is a relative point in the v.p. (x_v, y_v).
* So we want to calculate (x_v, y_v)

$$\frac{x_w - x_{wmin}}{x_{wmax} - x_{wmin}} = \frac{x_v - x_{vmin}}{x_{vmax} - x_{vmin}} \quad \text{---} (1)$$

$$\frac{y_w - y_{wmin}}{y_{wmax} - y_{wmin}} = \frac{y_v - y_{vmin}}{y_{vmax} - y_{vmin}} \quad \text{---} (2)$$

frm (1)

$$x_v = \frac{x_{vmax} - x_{vmin}}{x_{wmax} - x_{wmin}} \times (x_w - x_{wmin}) + x_{vmin}$$

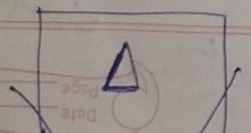
$$= x_{vmin} + x_w - x_{wmin} \cdot s_x.$$

frm (2)

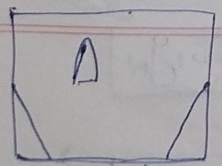
$$y_v = \frac{y_{vmax} - y_{vmin}}{y_{wmax} - y_{wmin}} \times (y_w - y_{wmin}) + y_{vmin}$$

$$= y_{vmin} + y_w - y_{wmin} \cdot s_y$$

⇒ Clipping =



bfre clipping



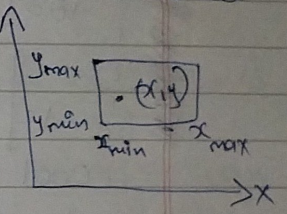
Afts clipping.

* It is a type of trans n used in comp graphics to remove the lines, obj, segments of line that are outside the comp screen or window.
* Types —

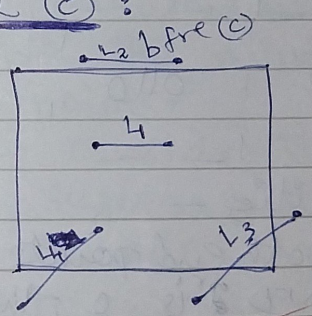
1) Point (c) :

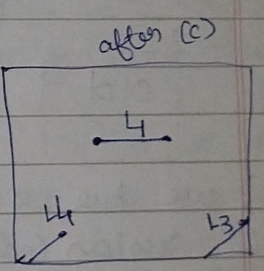
- used to determine whether the point is inside the window / not.
- For this the follo condin are checked —

$$x_{min} \le x \le x_{max}$$
$$y_{min} \le y \le y_{max}.$$



2) Line (c) :



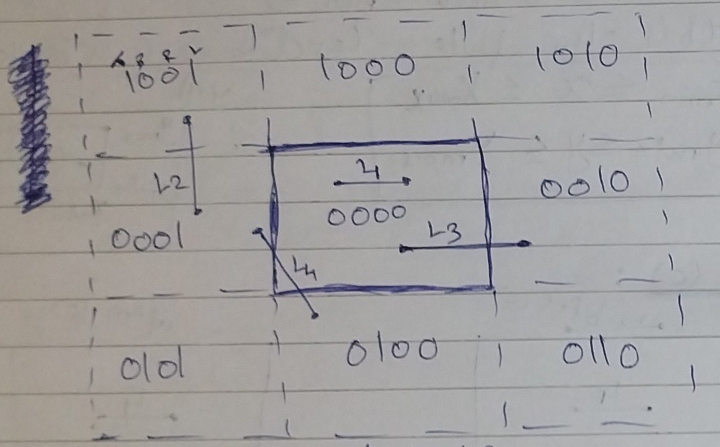


* Here, we will cut the portion of the line which is outside the window & give only the portion that is inside the window.
* perfect inside (L_1) → Accepts
* " outside (L_2) → Reject
* partial inside (L_4) → clipping
* Endpoints outside (L_3) → clipping.

★ Cohen Sutherland Line clipping (a1):

* Here, the window is divided into 9 regions, each region has a 4 bit region code, this 4 bits represents the top, bottom, R & L of the region as follows —

Region Code pattern:



* Here, we have 3 possibilities —

1) If the region code of both endpoints of the line is 0 & their AND opr is 0; then the line is completely inside of the window. So accept the line.

eg → Consider the line 'L1', the R-code of 2 endpoints are 0000, 0000 & perform the AND opr.

Here the rslt is 0000, so the L1 is Completly inside the window.

```
0000 AND
0000
----
0000
```

2) If the R-code of 2 endpoints are non-zero & their AND is non-zero; Then the line is completely outside the window, so reject the line.

eg → Consider the line 'L2', the R-C of 2 endpoints are ~~0001~~ → 0001, 1001 & perform AND opr —

1001 AND
0001

0001

0001 → reject the line.

3) a) If the R-C of 1 endpoint is 0 & 1 is non-zero & their AND is non-zero. Then we have to find the intersection point on window boundary & click the line outside the window finally find the new point.

eg → Consider the line 'L3' the end region of 2 endpoints is 0000 & 0010 & AND opr is —

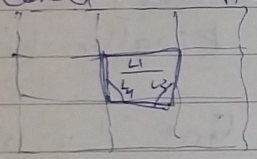
After (c) the points of the lines are 0000 →

```
0000
0010
----
0000
```

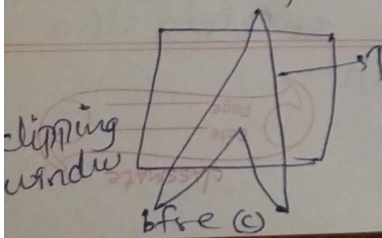
b) If the R-C of both endpoints is non-zero & their AND is 0 & we have to find the intersection point on window boundry & click the line outside the window, find the new endpoints.

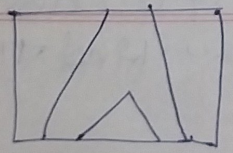
eg → L_4 the value of endpoints are
0001 & 0100 & AND opr →
0001 AND
0100
0000

The line is partially inside the window so find the intersec^n point & click the line.
After (c), the new epoints are 0001 & 0000
Still the line is partially inside the window.
So after (c) new epoints are 0000 & 0000.
Now the line is completely inside the window, accepts the line.
(c) will continue until either trively rejected.



3) Polygon clipping:
A set of connected lines are considered as polygon. polygon are clipped based on the window & the portion which is inside the window is kept as it is & the outside portion is clipped.





I Sutherland – Hodgeman polygon (c):

* Here, ~~each~~ check each edge of the polygon against all window boundaries.
Modify the vertices based on transitions.
Transfer the new edges to the next (c) boundary.

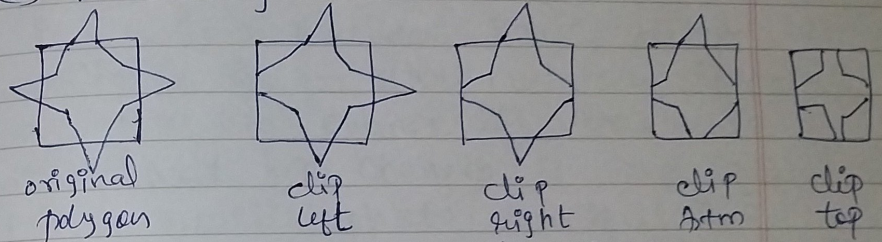
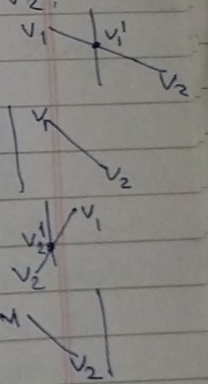


Fig: (c) window boundaries

* 4 cases —
1) V_1 outside, V_2 inside: take V_1' & V_2.
2) V_1 inside, V_2 ~~inside~~ inside: take V_1 & V_2
3) V_1 inside, V_2 outside: take V_1 & V_2'
4) V_1 outside, V_2 outside: take none



* Issues of this (c) → (c) the exteriors
→ (c) othr shapes – O, ellipse, curves.
→ (c) a shape against another shape.