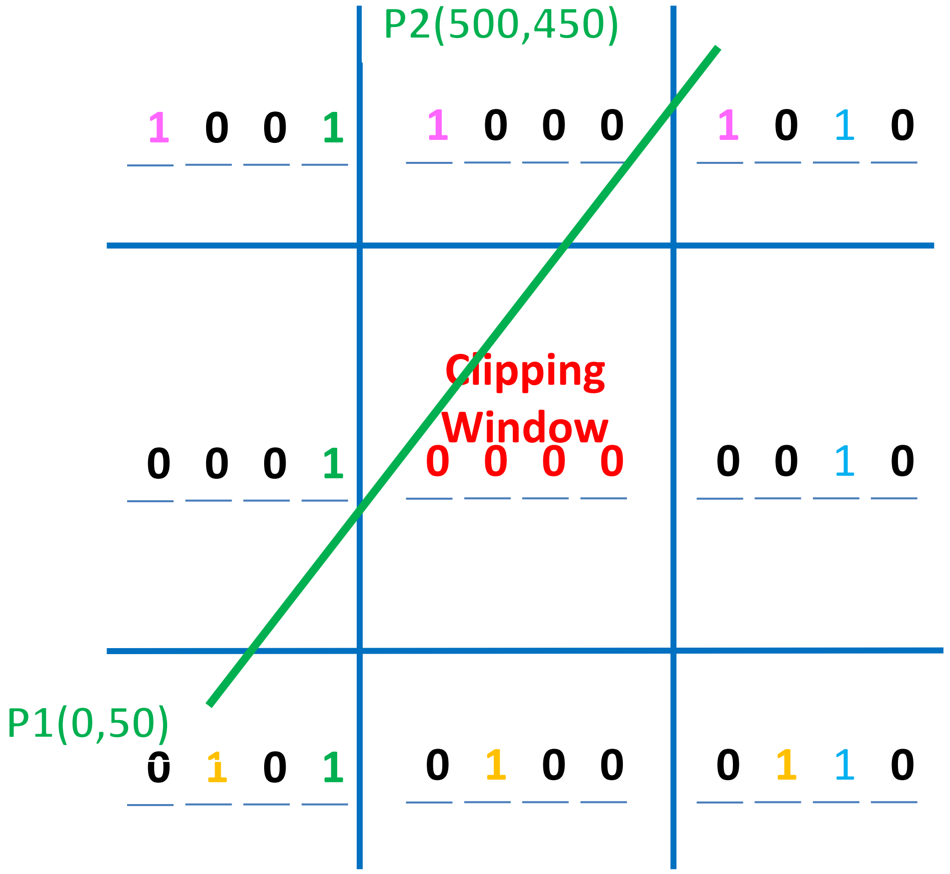
Nicholl-Lee-Nicholl Line (NLN) Clipping

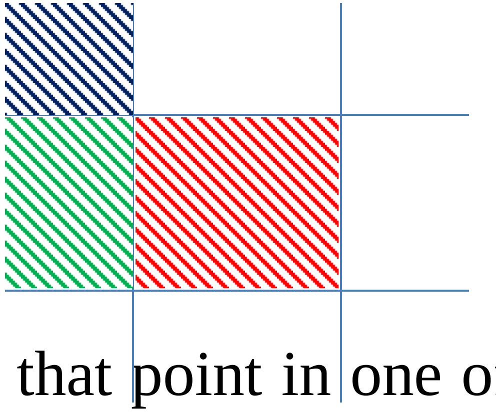
- In Cohen-Sutherland line clipping sometimes multiple calculation of intersection point of a line is done before actual window boundary intersection.
- These multiple intersection calculation is avoided in NLN line clipping procedure.
- By creating more regions around the clip window the NLN algorithm avoids multiple clipping of an individual line segment.
- NLN line clipping perform the fewer comparisons and divisions so it is more efficient.
- But NLN line clipping cannot be extended for three dimensions.



Cohen Sutherland line Clipping

Contd.

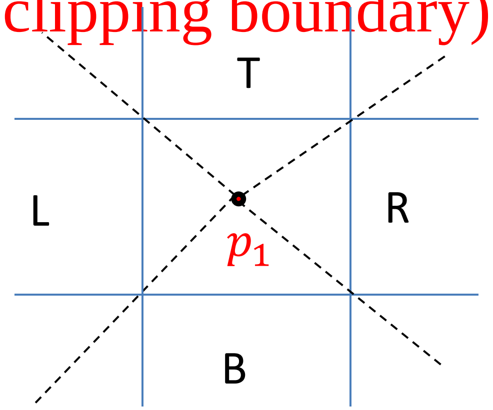
- For given line we find first point falls in which region out of nine region shown in figure.
- Only three region are considered which are.
  - Window region
  - Edge region
  - Corner region



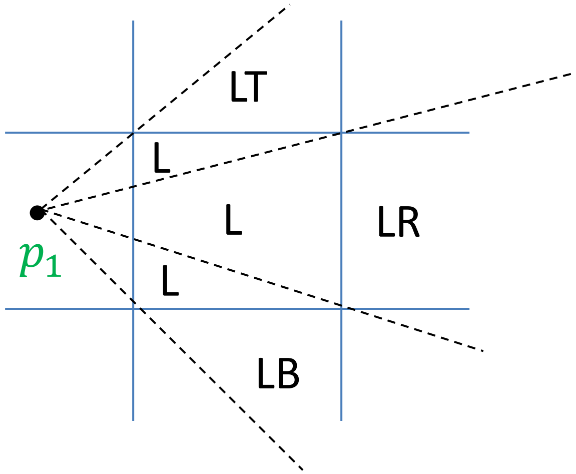
- If point falls in other region than we transfer that point in one of the three region by using transformations.
- We can also extend this procedure for all nine regions.

Dividing Region in NLN

- Based on position of first point out of three region highlighted we divide whole space in new regions.
- Regions are name in such a way that name in which region p2 falls is gives the window edge which intersects the line.
- $p_1$is in window region(P1 is inside clipping boundary)

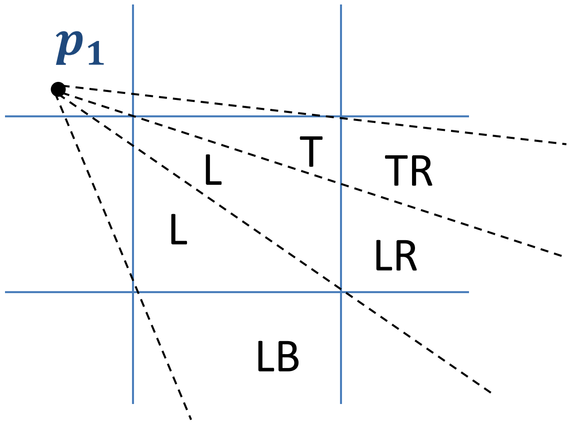


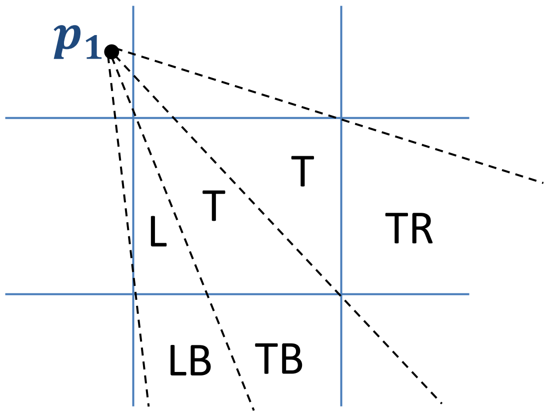
- $p_1$is in edge region



Contd.

- p_1 is in Corner region (one of the two possible sets of region can be generated)





Finding Region of Given Line in NLN

- For finding that in which region line $\boldsymbol{p_1p_2}$ falls we compare the slope of the line to the slope of the boundaries:

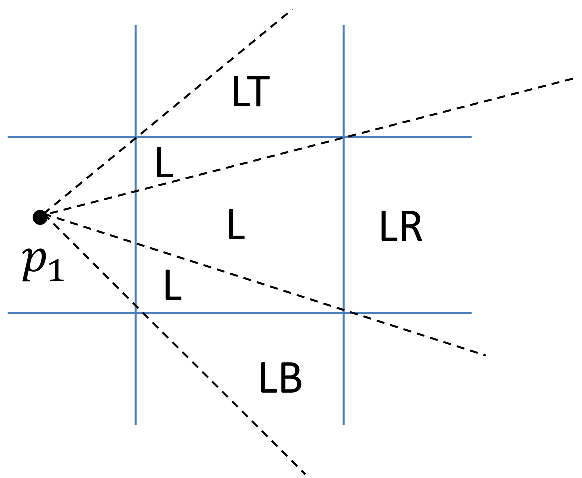
  $$\boldsymbol{slope\,\overline{p_1p_{B1}} < slope\,\overline{p_1p_2}} < slope\,\overline{\boldsymbol{p_1p_{B2}}}$$

  Where $\overline{\boldsymbol{p_1p_{B1}}}$ and $\overline{\boldsymbol{p_1p_{B2}}}$ are boundary lines.

- For example p1 is in edge region and for checking whether p2 is in region LT we use following equation.

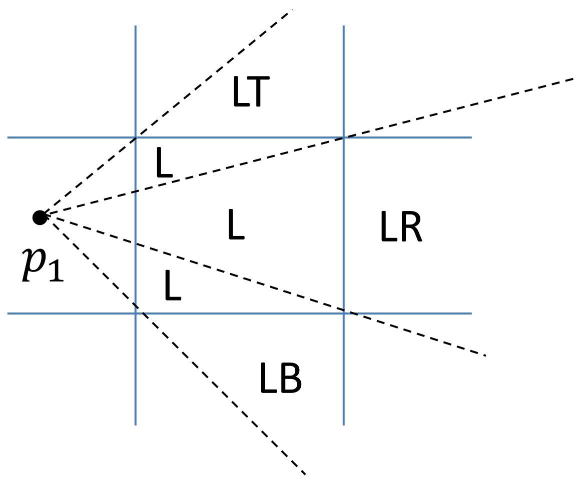
  $$\boldsymbol{slope\,\overline{p_1p_{TR}} < slope\,\overline{p_1p_2} < slope\,\overline{p_1p_{TL}}}$$

  $$\boldsymbol{\frac{y_T - y_1}{x_R - x_1} < \frac{y_2 - y_1}{x_2 - x_1} < \frac{y_T - y_1}{x_L - x_1}}$$



Contd.

- After checking slope condition we need to check weather it crossing zero, one or two edges.
- This can be done by comparing coordinates of p_2 with coordinates of window boundary.
- For left and right boundary we compare x coordinates and for top and bottom boundary we compare y coordinates.
- If line is not fall in any defined region than clip entire line.
- Otherwise calculate intersection.



Intersection Calculation in NLN

- After finding region we calculate intersection point using parametric equation which are:

$$\boldsymbol{x = x_1 + (x_2 - x_1)t}$$
$$\boldsymbol{y = y_1 + (y_2 - y_1)t}$$

- For left or right boundary $x = x_l \; or \; x_r$ respectively, with t $= (x_{l/r} - x_1)/(x_2 - x_1)$, so that y can be obtain from parametric equation as below:

$$\boldsymbol{y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x_L - x_1)}$$

- Keep the portion which is inside and clip the rest.

Contd.

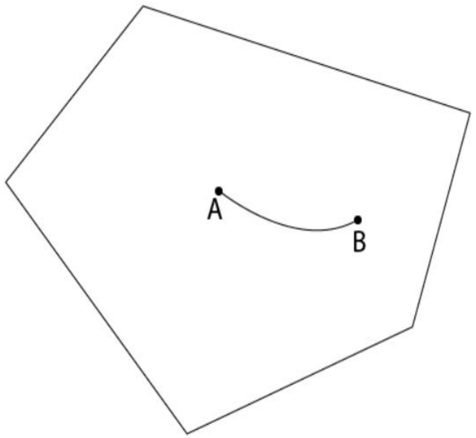
- Similarly for top or bottom boundary $y = y_t \, or \, y_b$ respectively, and t$=$ $(y_{t/b} - y_1)/(y_2 - y_1)$, so that we can calculate x intercept as follow:

$$\boldsymbol{x = x_1 + \frac{x_2 - x_1}{y_2 - y_1}(y_T - y_1)}$$

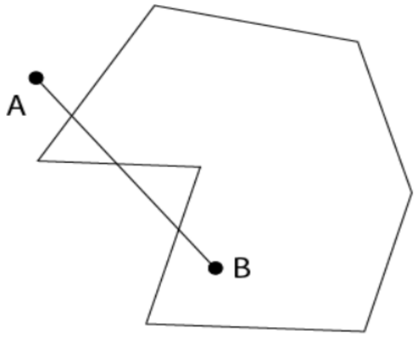
polygon

- Types of Polygons

1. Convex: A polygon is called convex of line joining any two interior points of the polygon lies inside the polygon. All interior angles are less than 180°.
2. Concave: A non-convex polygon is said to be concave. A concave polygon has one interior angle greater than 180°.



Convex polygon

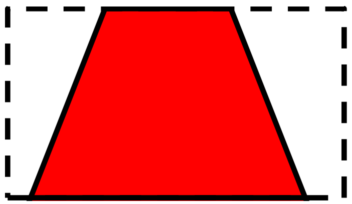


Concave polygon

Polygon Clipping

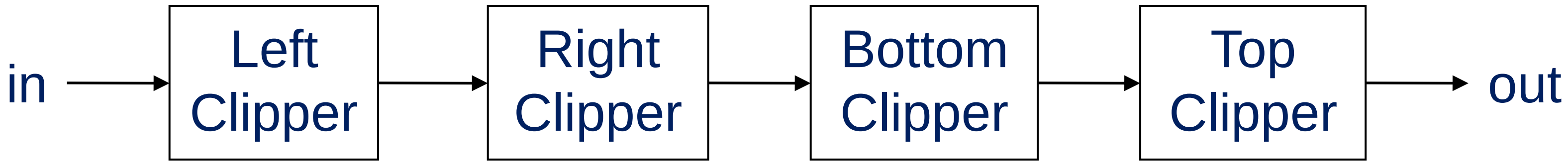
- For polygon clipping we need to modify the line clipping procedure.
- In line clipping we need to consider about only line segment.
- In polygon clipping we need to consider the area and the new boundary of the polygon after clipping.
- Various algorithm available for polygon clipping are:

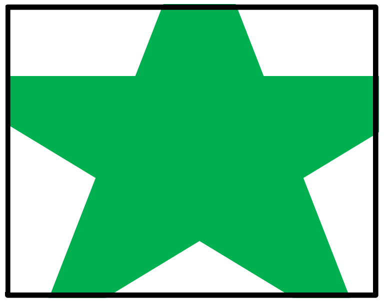
1. Sutherland-Hodgeman Polygon Clipping
2. Weiler-Atherton Polygon Clipping etc.



Sutherland-Hodgeman Polygon Clipping

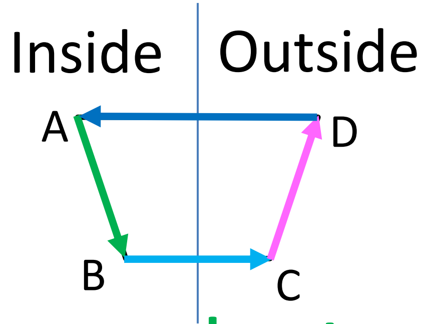
- For correctly clip a polygon we process the polygon boundary as a whole against each window edge.
- This is done by whole polygon vertices against each clip rectangle boundary one by one.





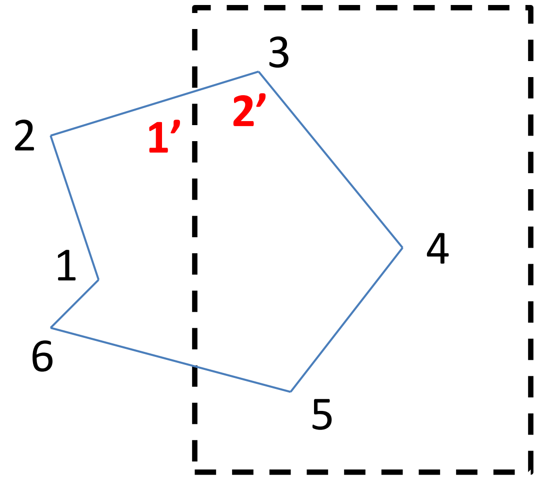
Processing Steps

- We process vertices in sequence as a closed polygon.
- Four possible cases are there.



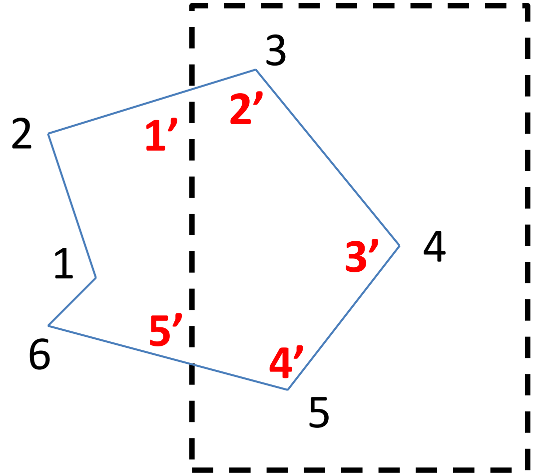
1. If both vertices are inside the window we add **only second vertex** to output list.
2. If first vertex is inside the boundary and second vertex is outside the boundary only the **edge intersection** with the window boundary is added to the output vertex list.
3. If both vertices are outside the window boundary **nothing is added** to window boundary.
4. first vertex is outside and second vertex is inside the boundary, then adds both intersection point with window boundary, and **second vertex** to the output list.

Example



- As shown in figure we clip against left boundary.
- Vertices 1 and 2 are found to be on the outside of the boundary.
- Then we move to vertex 3, which is inside, we calculate the intersection and add both intersection point and vertex 3 to output list.

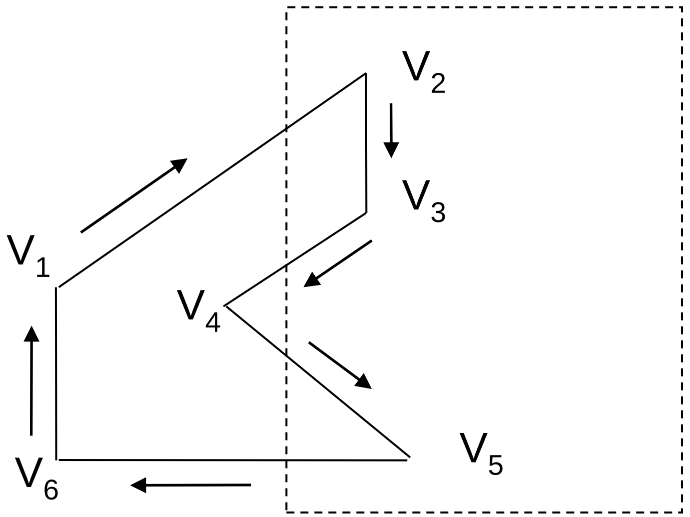
Contd.



- Then we move to vertex 4 in which vertex 3 and 4 both are inside so we add vertex 4 to output list.
- Similarly from 4 to 5 we add 5 to output list.
- From 5 to 6 we move inside to outside so we add intersection pint to output list.
- Finally 6 to 1 both vertex are outside the window so we does not add anything.

Limitatin of Sutherlan-Hodgeman Algorithm

- It may not clip concave polygon properly.

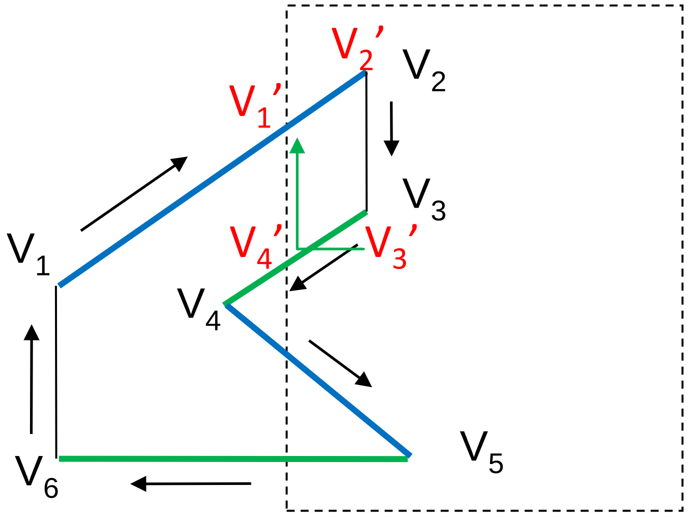


- One possible solution is to divide polygon into numbers of small convex polygon and then process one by one.
- Another approach is to use Weiler-Atherton algorithm.

Weiler-Atherton Polygon Clipping

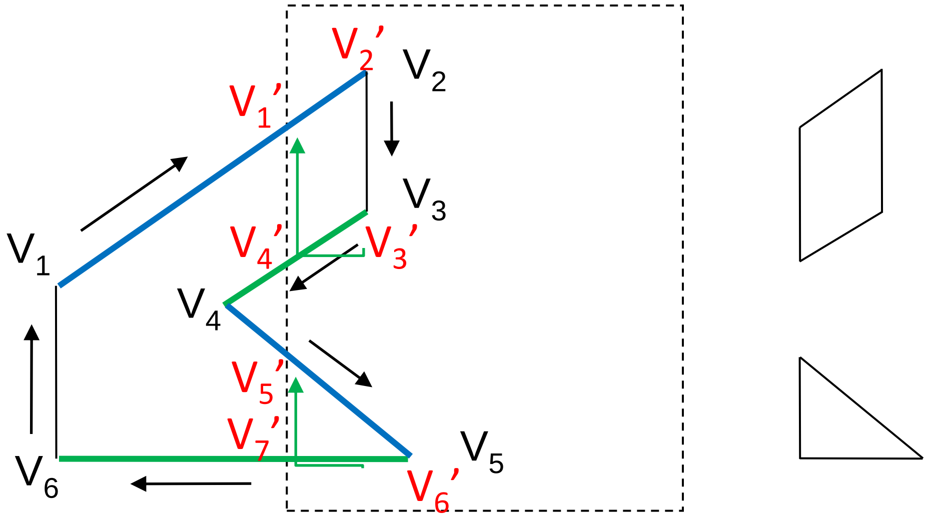
- It modifies Sutherland-Hodgeman vertex processing procedure for window boundary so that concave polygon also clip correctly.
- This can be applied for arbitrary polygon clipping regions as it is developed for visible surface identification.
- Procedure is similar to Sutherland-Hodgeman algorithm.
- Only change is sometimes need to follow the window boundaries Instead of always follow polygon boundaries.
- **For clockwise processing of polygon vertices we use the following rules:**
  1. **For an outside to inside pair of vertices, follow the polygon boundary.**
  2. **For an inside to outside pair of vertices, follow the window boundary in a clockwise direction.**

Example



- Start from v1 and move clockwise towards v2 and **<u>add intersection point and next point</u>** to output list by following **polygon boundary**,
- then from v2 to v3 we add **<u>v3 to output list</u>**.
- **From v3 to v4 we calculate <u>intersection point and add to output list</u> and follow window boundary.**

Contd.



- Similarly from v4 to v5 we **add intersection point and next point** and follow the **polygon boundary**,
- next we move v5 to v6 and add **intersection point** and follow the **window boundary**, and
- finally v6 to v1 is outside so no need to add anything.
- This way we get two separate polygon section after clipping.