UNIT 8: CLIPPING ALOGRITHMS

UNIT STRUCTURE

8.1 Learning Objectives

8.2 Introduction

8.3 Clipping

8.4 Line Clipping

- 8.4.1 Cohen-Sutherland Algorithm
- 8.4.2 Cyrus-beck line clipping algorithms

8.5 Polygon Clipping

- 8.5.1 Sutherland -Hodgeman algorithm

8.6 Let Us Sum Up

8.7 Further Reading

8.8 Answer to Check Your Progress

8.9 Model Questions

8.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- Understand the idea of clipping in graphics.
- Learn about line and polygon clipping.
- Know how the line clipping algorithms works.
- Learn the Sutherland -Hodgeman polygon clipping algorithm

8.2 INTRODUCTION

In the previous unit we have learnt about the 2D viewing which is a transformation process of real world objects into position point which is relative to the viewing volume. The viewing functions were also discussed.

Often a time in computer graphics or a drawing, we may be interested in a portion of the picture or a scene instead of the whole picture. This portion is the region of interest. In such a situation, we need a method to display only those portions of the drawing that are of immediate interest. This is just like looking at a large image through a window. The process of

selecting and viewing the picture is called windowing. Thus, there are two region that we have - visible and non visible. The part we are interested in the picture should be visible. The part that is not the region of interest should not be visible. This is called ***clipping***. In this unit, we will introduce you to clipping in computer graphics. Here we will find how line clipping and polygon clipping are done, along with algorithms.

8.3 CLIPPING

Picture consists of objects having curves and boundaries. Line segments and polygons usually make up the surface curves and the boundaries. Polygons are also formed by line segments. The word 'Clipping' refers to the removal of objects, lines, or line segments that are outside the viewing pane. Comninos [2006] describes clipping in the following, more general, way:

'**Clipping** is a process that subdivides each element of a picture, which is to be displayed, into its visible and invisible parts. In simple words, any process that identifies parts of a picture that lie either inside or outside a specified region is called clipping algorithm. This allows discarding the invisible parts of the picture and keeps only the parts that are to be considered. Clipping may be internal or external. An internal clipping removes parts of a picture outside a given region and an external clipping removes parts inside a region.

The clipping process can be applied to a variety of graphics primitives such as lines, points, polygons and curves. Clipping can be performed with respect to a clipping boundary, which may be a convex or concave polygonal boundary.

The specified region of the scene is called a clip window or simply window. Clipping is used to extract parts of a defined scene for viewing, anti-aliasing lines or object boundaries, identifying visible surfaces in three-dimensional views and many other applications.

The clip window can be a general polygon or a window that have curved boundaries, depending on the type of the application. In this unit,

we would be considering only rectangular shaped windows.

Clipping algorithms can be applied in world coordinates so that the contents of the window interior are mapped to device coordinates.

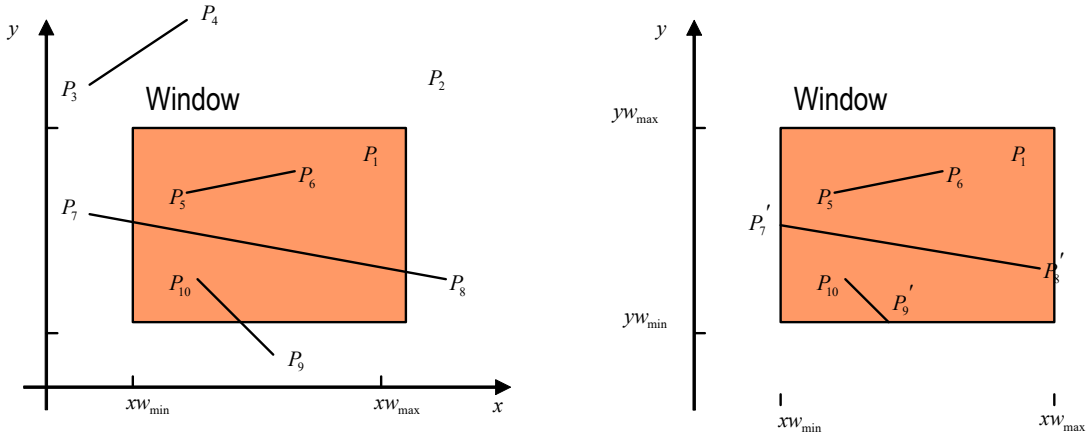
In the following sections we will discuss clipping that are applied to the following primitive types:

- Lines
- Polygon

8.4 LINE CLIPPING

A line clipping is the process of removing portions of line segments outside an area of interest that is, the clip window. The lines that intersect the clip window at some point have some portion of the line inside the clip window. The lines that do not intersect the window at any point are totally inside the window or totally outside the clip window. Thus, any line or a part of it that is outside the viewing area is removed.

To clip a line against a clip window, only the end-points are tested rather than testing all the points on the line. The figure below shows the line clipping against a clip window.



(a) Before Clipping **(b) After Clipping**

A line clipping algorithm has to test the following before clipping a line or any part of the line.

- Whether the line lies completely inside the window.
- Whether the line is completely outside the window

- If the line lies partially inside and partially outside the specified window.

Lines that are not intersecting any border of the clipping window are either completely outside or completely inside the window boundary. The intersecting lines have one segment of the line inside and the rest outside.

The clipping region is commonly called as clipping window, or if the window is a standard rectangle, it is called as clipping rectangle. There are many line clipping algorithms. The Cohen-Sutherland Line clipping algorithm is generally used for line clipping.

8.4.1 COHEN-SUTHERLAND ALGORITHM

This algorithm is one of the simple and most widely used. To clip a line, the end-points of the line are only considered. If both the end-points lie inside the boundary, it is accepted that the line is inside the clip window. Thus there is no clipping procedure to be performed. In the case, where both end-points lie entirely to one side of the clip window, the line must be entirely outside the window. It is trivially rejected and the line needs neither clipping nor is to be displayed. The Cohen-Sutherland algorithm works in two phases:

- It identifies those lines which are intersecting the window boundary
- Perform the clipping operation.

Let P(x,y) be a line. Say (xs,ys) and (xe, ye) be the line end-points. A line is said to be completely inside if the following condition holds:

$$Xw_{min} \leq xs \leq xw_{max}$$

$$Xw_{min} \leq xe \leq xw_{max}$$

and $yw_{min} \leq ys \leq yw_{max}$

$$yw_{min} \leq ye \leq yw_{max}$$

A line is said to be completely outside if the following condition holds :

$$Xs < xw_{min} \text{ and } xe < xw_{min}$$

Or, $Xs > xw_{max}$ and $xe > xw_{max}$

The algorithm assigns an 8 digit binary code to each of the two end-points of the line. This code is called the region code. The region code identifies the location of the point relative to the boundaries of the clipping window. Each bit position in the region code is used to indicate one of the four relative coordinate positions of the point with respect to the clip window: to the left, right, top or bottom.

The figure below shows the binary codes that are assigned to the end-points of a line.

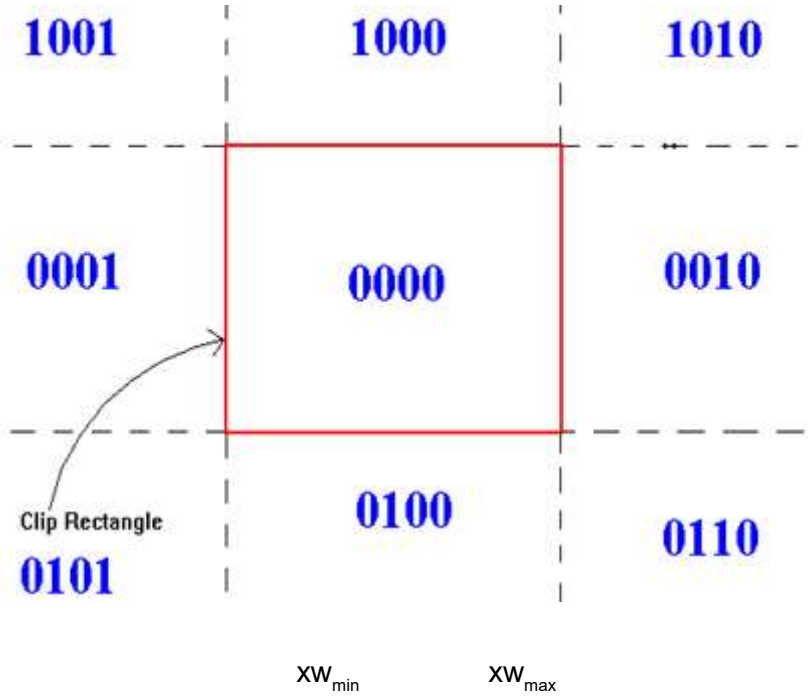
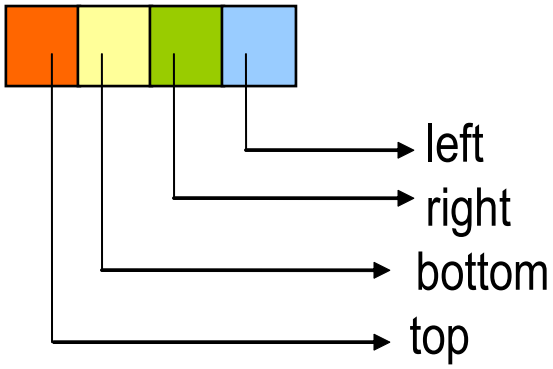


Figure : The binary region codes assigned to the line end-points.

This algorithm has extended the window boundaries to define 9 regions :

Top-left, top-center, top-right, center-left, center, center-right, bottom-left, bottom-center, bottom-right.

These 9 regions are uniquely identified by the given region code. The order --- left, right, bottom, top (LRBT) will be used for the 8-bit region code.



The sequence for reading the code bits is Left, Right, Bottom and Top, that is from the least-significant bit to the most-significant bit.

A value of 1 in any bit position indicates that the point lies in that position; otherwise that bit position is set to a value of 0. For any point P (x,y) :

- Left (first) bit is set to 1 if P lies to left of window, that is, when x < xwmin
- Right (second) bit is set to 1 if P lies to right of window,
- Bottom (third) bit is set to 1 if P lies below window, that is, when y < ywmin
- Top (fourth) bit set is set to 1 if P lies above window, that is, when y > ywmax

If the point P lies within the clip window, then all the bit positions are set to a value 0, that is, 0000.

For a point that lies to the top-left corner of the clip window, the region code is 1001.

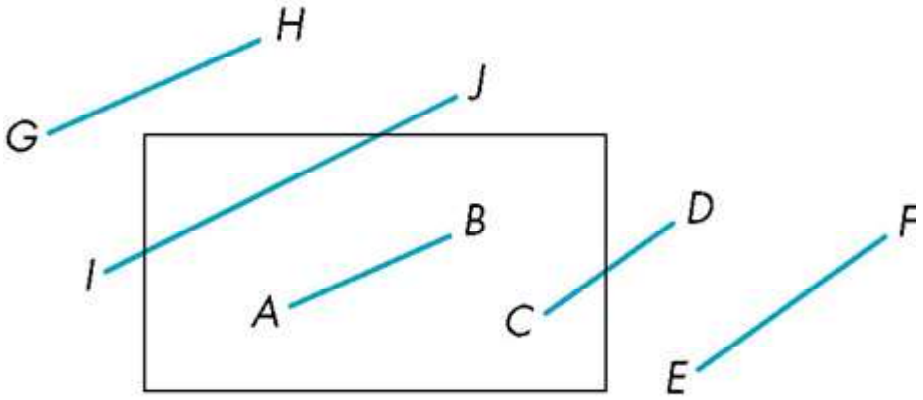
Thus the region codes for the line end-points are established. Using this region code it is now easier to find out which of the lines are inside the clip window and which are not.

- If the region code for both the end-points of the line is 0000, then the line is completely inside the clip window and is accepted.
- If the region codes for both the end-points have a 1 in the same bit position, then the line is completely outside the clip window and it is trivially rejected. For example, for the line where one

end-point is having the region code as 1001 and the other end-point is having the code 0101.

- A logical AND operation can also be performed on the two region codes of the two end-points of the line to check for clipping the lines.

The lines which are not completely inside or completely outside the clip window are tested for intersection with the window boundaries. The figure below shows some lines and the clip window.



Let us consider the five cases above where we have to perform line clipping.

AB : The region code for line A is 0000.
The region code for B is 0000.
So the line AB is accepted as both the region codes are 0000.

CD : C has code 0000.
D has 0010.

The line CD thus neither lie completely inside nor completely outside as one of the end-point lie inside the window and the other lie to the left of the window (as the right bit is 1). The line thus has to be shortened by intersecting with one of the window edge. Compute the code of intersection that is the new end-point of the shortened line segment and then check the end-points (C and new one for D) again.

The location of the bit 1 determines which edge of the window to intersect. If there were two 1's in the region codes of the end-

points then two intersections might have to be done.

EF : The end-point E has the code 0010
F has the code 0010.

As there is a 1 in the same bit position of the two end-points of the line, the line EF lies completely outside the window and is thus discarded.

GH : The region code for point G is 1001
Code for the point H is 1000.

As there is a 1 bit in the same position of G and H, the line is completely outside and is discarded.

IJ : Region code for I is 0001
Region code for J is 1000.

As there is a 1 bit in the left bit position of I, it means that it lies to the left of the clip window.

Again as a 1 exists in the top bit position of J so it lies to the top of the window.

This means that the line IJ might have intersected the window in two different positions. The intersection point of I with the window say I/ is first calculated. So the new line segment is now I/J.

Then the intersection point of J with the window is calculated, say we have J/. So, I/J/ is the new line segment.

The region codes for I/ and J/ is now checked. For I/ it is 0000 and for J/ it is also 0000. so the line I/J/ is accepted and lies inside the clip window.

This algorithm has many advantages. It is the simplest one, and easy to understand and program. It does not have any unnecessary computations to determine fully visible and invisible lines. The memory usage of the algorithm is economical.

However, it is not left without any shortcoming too. For non-trivial cases, finding an intersection point might need to do four times intersection calculation per vector.

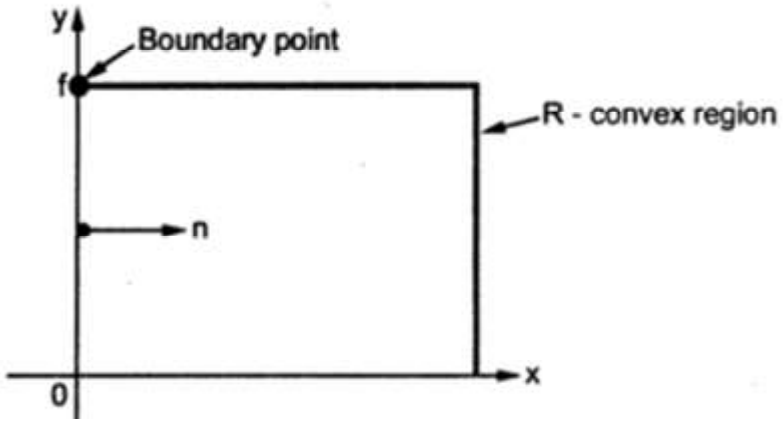
8.4.2 Cyrus-beck Line Clipping Algorithm

The Cyrus-Beck line clipping algorithm is a generalized line clipping algorithm that is applicable to arbitrary convex polygons and not necessarily rectangular window. Not only a rectangular window, but also it works fine with non-rectangular windows. It is more efficient than the Cohen-Sutherland algorithm which uses repetitive clipping. It uses parametric line representation and simple dot products.

The algorithm explained above assumes that the clipping window is a regular rectangle. Cyrus and Beck have developed a generalized line clipping algorithm. It is applicable to an arbitrary convex region. It uses a parametric equation of a line segment to find the intersection points of a line with the clipping edges.

This algorithm generally clips only once or twice unlike the Cohen Sutherland algorithm where the lines are clipped about four times.

The algorithm is applicable to an arbitrary convex region. A parametric equation of a line segment is used in this algorithm to find the intersection points of a line with the clipping edges.

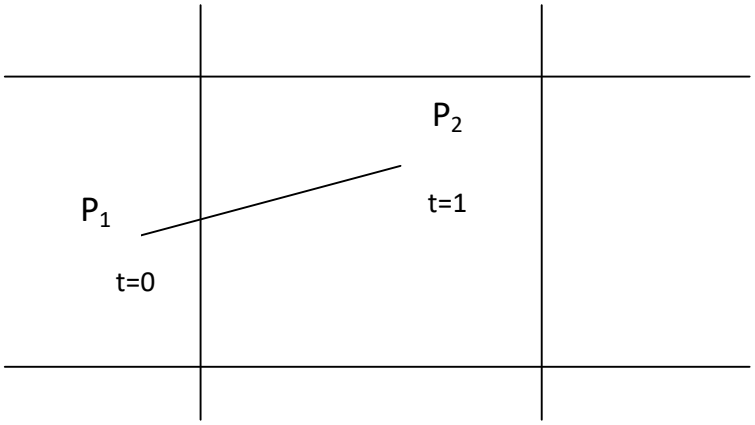


The parametric equation of a line segment from P1 to P2 is

$$P(t) = P_1 + (P_2 - P_1)\, t \qquad ; 0 \le t \le 1$$

$$= (1\text{-}t)\, P_1 + tP_2$$

Where t is a parameter, t=0 at P_1 and t=1 at P_2.



The value of the dot product is calculated so as to find the intersection point with the clipping window. Let PE be a point on the clipping plane E.

Calculate :

if < 0, vector pointed towards interior;

if = 0, vector pointed parallel to plane containing P;

if > 0, vector pointed away from interior.

Here n stands for normal of the current clipping plane (pointed away from interior).

By this we select the point of intersection of line and clipping window where (dot product is 0) and hence clip the line.

The algorithm is as follows:

Step1: Input the line segment P1P2 with end coordinates P1(x1,y1) and P2(x2,y2) respectively.

Step 2: Read the vertex coordinate of the clipping polygon and calculate the direction D = P2 - P1

Step 3: Assign boundary point P to the edge and compute the inner normal n for the edge.

Step 4: Compute lower and upper limits.

If n.D > 0

Else $t_L = -\frac{n - W}{n - D}$

$$t_u = -\frac{n - W}{n - D}, \text{Where } D \neq 0$$

$$t_L = -\frac{n - W}{n - D}, \text{Where } D \neq 0$$

Step 5: Repeat steps 3 to 4 for each edge of the clipping polygon.

Step 6: Find maximum lower limit and minimum upper limit if they do ot satisfy the condition 0 d" t d" 1

Step 7: Compute the intersection points using the maximum lower limit and minimum upper limit in the parametric equation.

Step 8: Draw the line segment from $P(t_L)$ to $P(t_U)$

Step 9: Finish

The Cyrus-Beck algorithm is efficient when many line segments needs to be clipped. The algorithm can also be extended to convex polygon clip windows.



CHECK YOUR PROGRESS-1

Q 1: Fill in the blanks :

a. _____ refers to the removal of objects, lines, or line segments that are outside the viewing pane.

b. Clipping can be performed with respect to a clipping boundary, which may be a _____ or _____ polygonal boundary.

c. Internal clipping removes parts of a picture _____ the window and external clipping removes those parts that are _____ a region.

d. To clip a line against a clip window, only the _____ are tested rather than testing all the points on the line.

e. The region code in the Cohen- Sutherland line clipping algorithm identifies the location of the point relative to the _____ of the clipping window.

f. The _____ line clipping algorithm is more efficient than the Cohen-Sutherland algorithm which uses repetitive clipping.

8.5 POLYGON CLIPPING

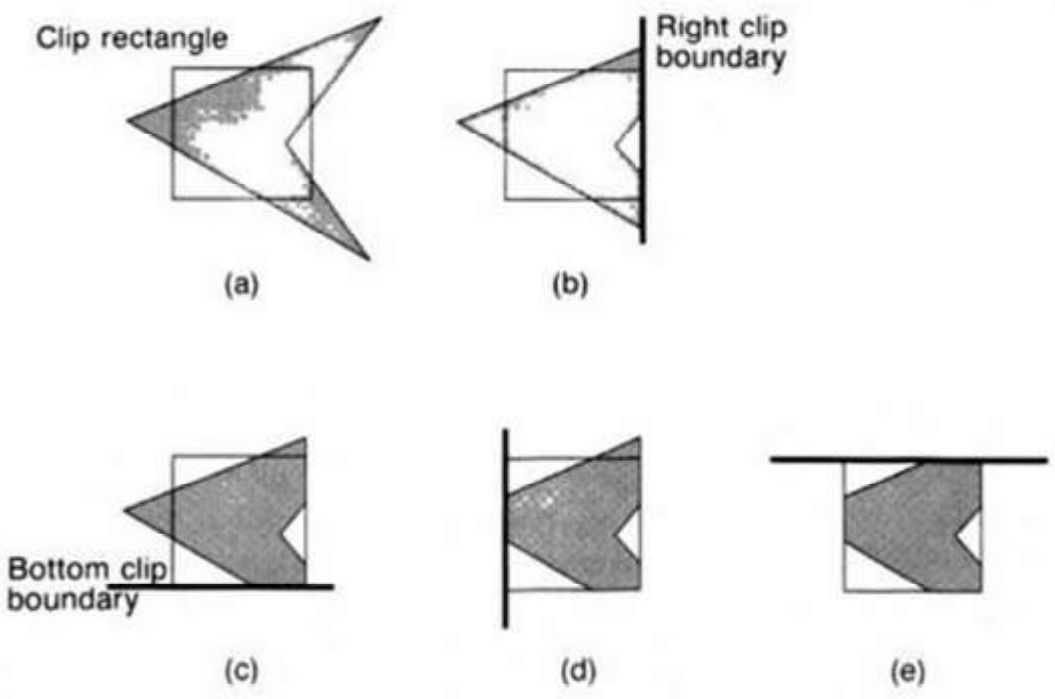
In the previous sections we have seen the line clipping algorithms. In this section we would be learning about the polygon clipping algorithm. A polygon is a closed figure that is bounded by straight lines. Polygons can be concave or convex. Computer graphics deals with both types. Although polygon is a collection of lines, the polygon clipping algorithms and the line clipping algorithms are different. If a line algorithm is used to clip a closed polygon, the original closed polygon becomes one or more open polygons or discrete lines.

In case of polygon clipping, each edge of the polygon must be tested against each edge of the clip rectangle, new edges must be added and existing edges must be discarded, retained or divided. Sometimes, clipping one polygon may result in many polygons.

8.5.1 Sutherland -Hodgman algorithm

There are several polygon clipping algorithms. One of the earliest polygon clipping algorithms is the Sutherland-Hodgeman algorithm. It is relatively simple. The algorithm is efficient in two cases: when the polygon is completely inside the boundaries and other is when it is completely outside.

It is based on clipping the entire polygon against an edge of the window, then clipping the new polygon against the next edge of the window and so on until the polygon has been clipped against all of the four edges. That is, the algorithm starts with an initial set of polygon vertices and clipping against the left window edge to produce a new set of vertices. The new set of vertices is then clipped against the bottom edge, right edge and the top edge respectively. In other words, it uses a divide-and-conquer strategy to solve the problem.



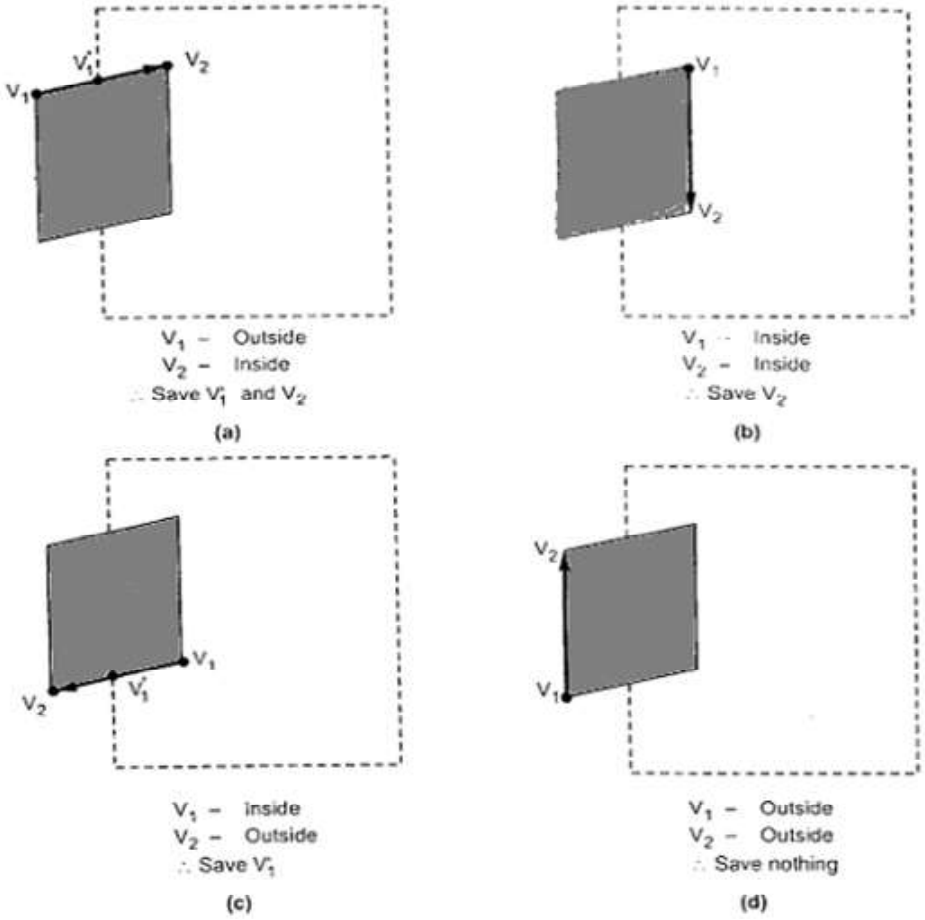
The figure above shows polygon clipping, edge by edge. Figure (a) shows before clipping. The figure (b) shows clipping on the right, (c) Clip on bottom, (d) Clip on left, (e) Clip on the top and the polygon is fully clipped.

All the polygon vertices are processed against each clip rectangle boundary in turn. Beginning with the original polygon, it is clipped. At each step a new set of polygon vertices is generated and passed on to the next window boundary clipper, as shown in the figure above. This is the fundamental idea that is used in this algorithm.

The output of this algorithm is a list of polygon vertices all of which are on the visible side of a clipping plane. This is achieved by processing two vertices of each edge of the polygon around the clip boundary. This results in four possible relationships between the edges and clip boundary.

- If the first vertex of the edge is outside the window boundary and the second vertex of the edge is inside then the intersection point of the polygon edge with the window boundary and the second vertex are added to the output vertex list.

- If both vertices of the edge are inside the window boundary, only the second vertex is added to the output vertex list.
- If the first vertex of the edge is inside the window boundary and the second vertex of the edge is outside, only the edge intersection with the window boundary is added to the output vertex list.
- If both vertices of the edge are outside the window boundary, nothing is added to the output list.





CHECK YOUR PROGRESS-2

Q 2: Fill in the blanks:

a._____ uses a divide-and-conquer strategy.

b. All the _____ vertices are processed against

each clip rectangle boundary in turn.

c. The output of the Sutherland-Hodgman algorithm is a list of polygon vertices all of which are on the visible side of a_____.

d. If both vertices of the edge are inside the window boundary, only the _____ is added to the output vertex list.

e. If both vertices of the edge are outside the window boundary, _____ is added to the output list.



8.6 LET US SUM UP

- Clipping' refers to the removal of parts of a picture that are either inside or outside a specified region.
- Internal clipping removes parts of a picture outside the window; external clipping removes those parts that are inside a region.
- A point P(x,y) is inside the clip window if the point lies between the window boundaries. That is,

$$xw_{min} \leq x \leq xw_{max}$$
$$yw_{min} \leq y \leq yw_{max}$$

- A line clipping algorithm has to test the following before clipping a line or any part of the line.
  a. Whether the line lies completely inside the window.
  b. whether the line is completely outside the window
  c. if the line lies partially inside and partially outside the specified window.
- The algorithm assigns an 8 digit binary code to each of the two end-points of the line. This code is called the region code. The region code identifies the location of the point relative to the boundaries of the clipping window.
- The Cyrus-Beck line clipping algorithm is more efficient than the Cohen-Sutherland algorithm which uses repetitive clipping
- Cyrus-Beck line clipping algorithm generally clips only once or twice unlike the Cohen Sutherland algorithm where the lines are clipped about four times.

- Although the polygon is a collection of lines, the polygon clipping algorithms and the line clipping algorithms are different.
- One of the earliest polygon clipping algorithms is the Sutherland-Hodgeman algorithm.
- The output of Sutherland-Hodgeman polygon clipping algorithm is a list of polygon vertices all of which are on the visible side of a clipping plane.



8.7 ANSWER TO CHECK YOUR PROGRESS

Ans to Q No 1:

a. Clipping
b. Convex, concave
c. Outside, inside
d. end-points
e. boundaries
f. Cyrus-Beck

Ans to Q No 2:

a Sutherland-Hodgman algorithm
b. polygon
c. clipping plane
d. second vertex
e. nothing



8.8 FURTHER READINGS

1) Computer Graphics By Donald Hearn and M. Pauline Baker
2) Fundamentals of Computer Graphics, Third Edition, By Peter Shirley, Steve Marschner
3) Computer graphics: principles and practice By James D. Foley



8.9 MODEL QUESTIONS

Q 1: Explain clipping. What is line and polygon clipping?

Q 2: Discuss Line clipping. Explain the Cohen-Sutherland Algorithm.

Q 3: Discuss Cyrus-beck line clipping algorithm.

Q 4: Explain the Sutherland-Hodgman algorithm for polygon clipping.

*** ***** ***