BMS INSTITUTE OF TECHNOLOGY & MANAGEMENT

YELAHANKA, BENGALURU – 560064

Department of Computer Science and Engineering

Report on

"Cohen Sutherland line clipping algorithm AND Sutherland-Hodgman polygon clipping algorithm"

Computer Graphics and Visualization – 18CS64

VI Semester, 6A CSE

Submitted By

Bhavana N S	USN: 1BY20CS038
Brundaja D N	USN: 1BY20CS041
B C Narendra	USN: 1BY21CS402
Chethan Kumar N	USN: 1BY21CS405

Under the Guidance of

Mrs. AMBIKA G N
Assistant Professor

2022-2023

ABSTRACT:

Clipping algorithms play a crucial role in computer graphics for efficiently determining visible portions of lines and polygons within specified regions or windows. Two widely used clipping algorithms are the Cohen-Sutherland line clipping algorithm and the Sutherland-Hodgman polygon clipping algorithm.

The Cohen-Sutherland algorithm is designed to efficiently clip line segments against a rectangular clipping window. By assigning endpoint codes based on the relative position of the endpoints to the clipping window boundaries, the algorithm determines the visibility of the line segment and computes intersection points with the window edges. It iteratively clips the line against each window boundary, resulting in efficient rendering of only the visible portions of the line.

The Sutherland-Hodgman algorithm, on the other hand, is used for clipping convex or concave polygons against arbitrary convex clipping regions. It works by iteratively clipping the polygon against each edge of the clipping region, resulting in a new polygon that represents the visible portion. The algorithm preserves the original polygon's shape and topology, offering accurate geometric results.

Both algorithms have various applications in computer graphics. The Cohen-Sutherland line clipping algorithm is utilized for rendering, clipping in 2D graphics, windowing systems, line segment intersection detection, hidden surface removal, and augmented reality/virtual reality experiences. The Sutherland-Hodgman polygon clipping algorithm finds applications in rendering, clipping in 2D graphics, hidden surface removal, geometric operations, image processing, CAD, GIS, simulations, and computational geometry tasks.

INTRODUCTION:

Clipping algorithms are vital tools in computer graphics for determining which parts of lines and polygons are visible within specific regions or windows. The Cohen-Sutherland line clipping algorithm and the Sutherland-Hodgman polygon clipping algorithm are widely used techniques that have greatly influenced the field.

These clipping algorithms have numerous applications in computer graphics. The Cohen-Sutherland line clipping algorithm is widely used in rendering, 2D graphics clipping, windowing systems, intersection detection, hidden surface removal, and augmented reality/virtual reality experiences. The Sutherland-Hodgman polygon clipping algorithm finds application in rendering, 2D graphics clipping, hidden surface removal, geometric operations, image processing, CAD, GIS, simulations, and computational geometry tasks.

Understanding the principles and applications of these algorithms is crucial for efficient graphics processing, precise clipping, and accurate visualizations in computer graphics. The Cohen-Sutherland line clipping algorithm and the Sutherland-Hodgman polygon clipping algorithm have greatly advanced the field of computer graphics and continue to play a significant role in various applications.

COHEN SUTHERLAND LINE CLIPPING ALGORITHM

OBJECTIVES:

The primary objective of the Cohen-Sutherland algorithm is to quickly identify line segments that are either completely outside, partially inside, or entirely inside a rectangular clipping region. By efficiently discarding line segments that are

entirely outside the region, the algorithm reduces the computational burden associated with rendering only the visible portions of lines.

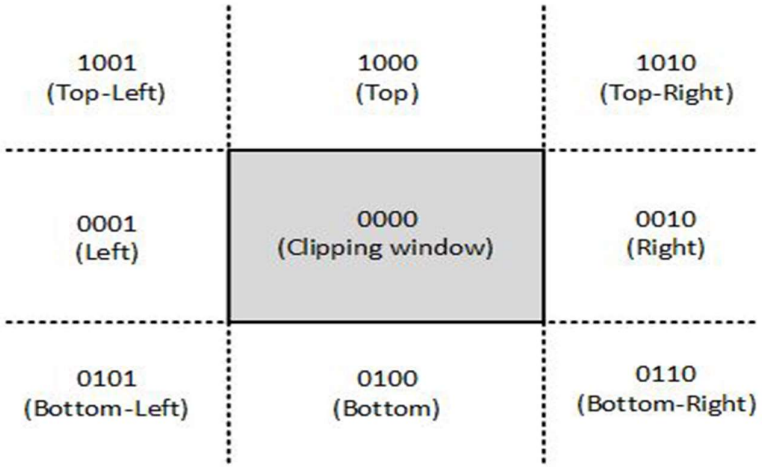
In the algorithm, first of all, it is detected whether line lies inside the screen or it is outside the screen. All lines come under any one of the following categories:

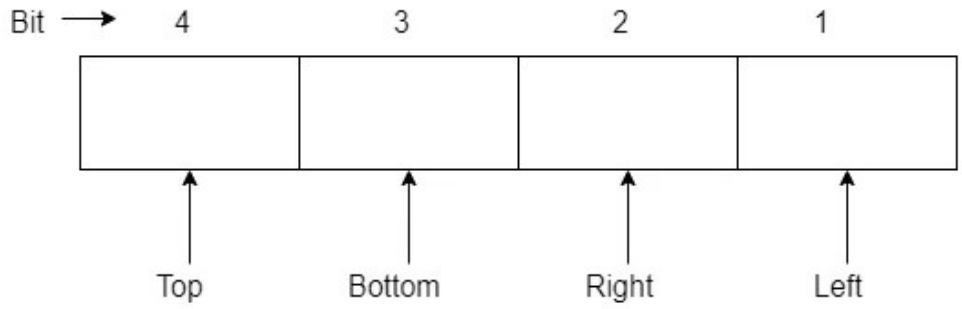
1. **Visible**
2. **Not Visible**
3. **Clipping Case**

1. Visible: If a line lies within the window, i.e., both endpoints of the line lies within the window. A line is visible and will be displayed as it is.

2. Not Visible: If a line lies outside the window it will be invisible and rejected. Such lines will not display.

3. Clipping Case: If the line is neither visible case nor invisible case. It is considered to be clipped case. First of all, the category of a line is found based on nine regions given below. All nine regions are assigned codes. Each code is of 4 bits. If both endpoints of the line have end bits zero, then the line is considered to be visible.





Four-bit Region Code

- In this algorithm, we are given 9 regions on the screen. Out of which one region is of the window and the rest 8 regions are around it given by 4 digit binary.
- The central part is the viewing region or window, all the lines which lie within this region are completely visible. A region code is always assigned to the endpoints of the given line.

ALGORITHM:

Sutherland and Cohen subdivision line clipping algorithm :

1. Read two end points of the line say P_1 (x_1, y_1) and P_2 (x_2, y_2).
2. Read two corners (left-top and right-bottom) of the window, say $(Wx_1, Wy_1$ and $Wx_2, Wy_2)$.
3. Assign the region codes for two endpoints P_1 and P_2 using following steps :

   Initialize code with bits 0000

   Set Bit 1 - if $(x < Wx_1)$

   Set Bit 2 - if $(x > Wx_2)$

   Set Bit 3 - if $(y < Wy_2)$

   Set Bit 4 - if $(y > Wy_1)$

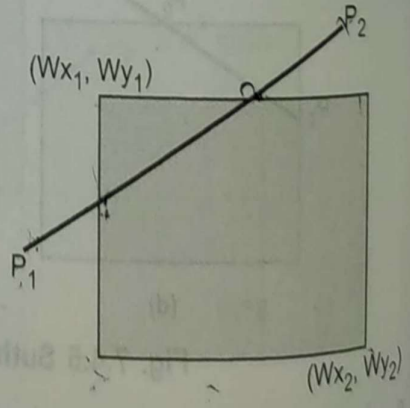


Fig. 7.4.6

Line clipping and Polygon clipping

4. Check for visibility of line P_1 P_2
   a) If region codes for both endpoints P_1 and P_2 are zero then the line is completely visible. Hence draw the line and go to step 9.
   b) If region codes for endpoints are not zero and the logical ANDing of them is also non-zero then the line is completely invisible, so reject the line and go to step 9.
   c) If region codes for two endpoints do not satisfy the conditions in 4a) and 4b)the line is partially visible.
5. Determine the intersecting edge of the clipping window by inspecting the region codes of two endpoints.
   a) If region codes for both the end points are non-zero, find intersection points P_1' and P_2' with boundary edges of clipping window with respect to point P_1 and point P_2, respectively.
   b) If region code for any one end point is non-zero then find intersection point P_1' or P_2' with the boundary edge of the clipping window with respect to it.
6. Divide the line segments considering intersection points.
7. Reject the line segment if any one end point of it appears outsides the clipping window.
8. Draw the remaining line segments.
9. Stop.

algorithm to clip two lines

EXAMPLE:

ple 7.4.2 *Use the Cohen-Sutherland outcode algorithm to clip two lines (40, 15) – P_2 (75, 45) and P_3 (70, 20) – P_4 (100, 10) against a window A (50, 10), (80, 10), C (80, 40), D(50,40).*

on : Line 1 : P_1 (40, 15) P_2 (75, 45) $x_L = 50$ $y_B = 10$ $x_R = 80$ $y_T = 40$

Point	Endcode	ANDing	Position
P_1	0 0 0 1	0 0 0 0	Partially visible
P_2	0 0 0 0		

$$m = \frac{45-15}{75-40} = \frac{6}{7}$$

$$L, y = m(x_L - x) + y_1 = \frac{6}{7}(50 - 40) + 15$$

$$= 23.57$$

$$x = x_1 + \left(\frac{1}{m}\right)(y_T - y_1) = 40 + \left(\frac{7}{6}\right)(40 - 15)$$

$$= 69.16$$

I_1 = (50, 23.57) I_2 = (69.06, 40)

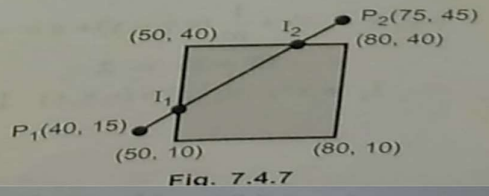


Fig. 7.4.7

Line 2 : P_3 (70, 20) P_4 (100, 10)

Point	End code	ANDing	Position
P_3	0 0 0 0	0 0 0 0	Partially visible
P_4	0 0 1 0		

$$\text{Slope } m' = \frac{10-20}{100-70} = \frac{-10}{30} = \frac{-1}{3}$$

$$x_R, y = m(x_R - x_1) + y_1 = \frac{-1}{3}(80 - 70) + 20 = 16.66$$

I = (80, 16.66)

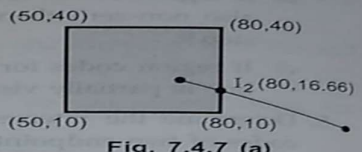


Fig. 7.4.7 (a)

ADVANTAGES:

- ✓ It is simple to learn and apply.
- ✓ It is best suited for lines that are totally inside or outside.
- ✓ It can easily be extended for 3D line clipping.

DISADVANTAGES:

- ✓ Clipping is costly when done repeatedly.
- ✓ This only applies to rectangular clipping windows.
- ✓ It is incapable of handling any other shape.

APPLICATION:

1. **Rendering:** Efficiently determining which parts of line segments should be rendered, optimizing rendering performance.
2. **Clipping in 2D graphics:** Precisely defining the visible portions of line segments within a specified region or viewport.
3. **Windowing systems:** Managing windows and graphical elements within defined areas in graphical user interfaces (GUIs).
4. **Augmented Reality (AR) and Virtual Reality (VR):** Determining visible parts of virtual objects for immersive user experiences.

SUTHERLAND-HODGMAN POLYGON CLIPPING

OBJECTIVES:

The main objective of the Sutherland-Hodgman algorithm is to identify the intersection points between the polygon edges and the boundaries of the clipping window. By iteratively clipping each edge of the polygon against the window boundaries, it creates a new polygon that represents the visible portion of the original polygon

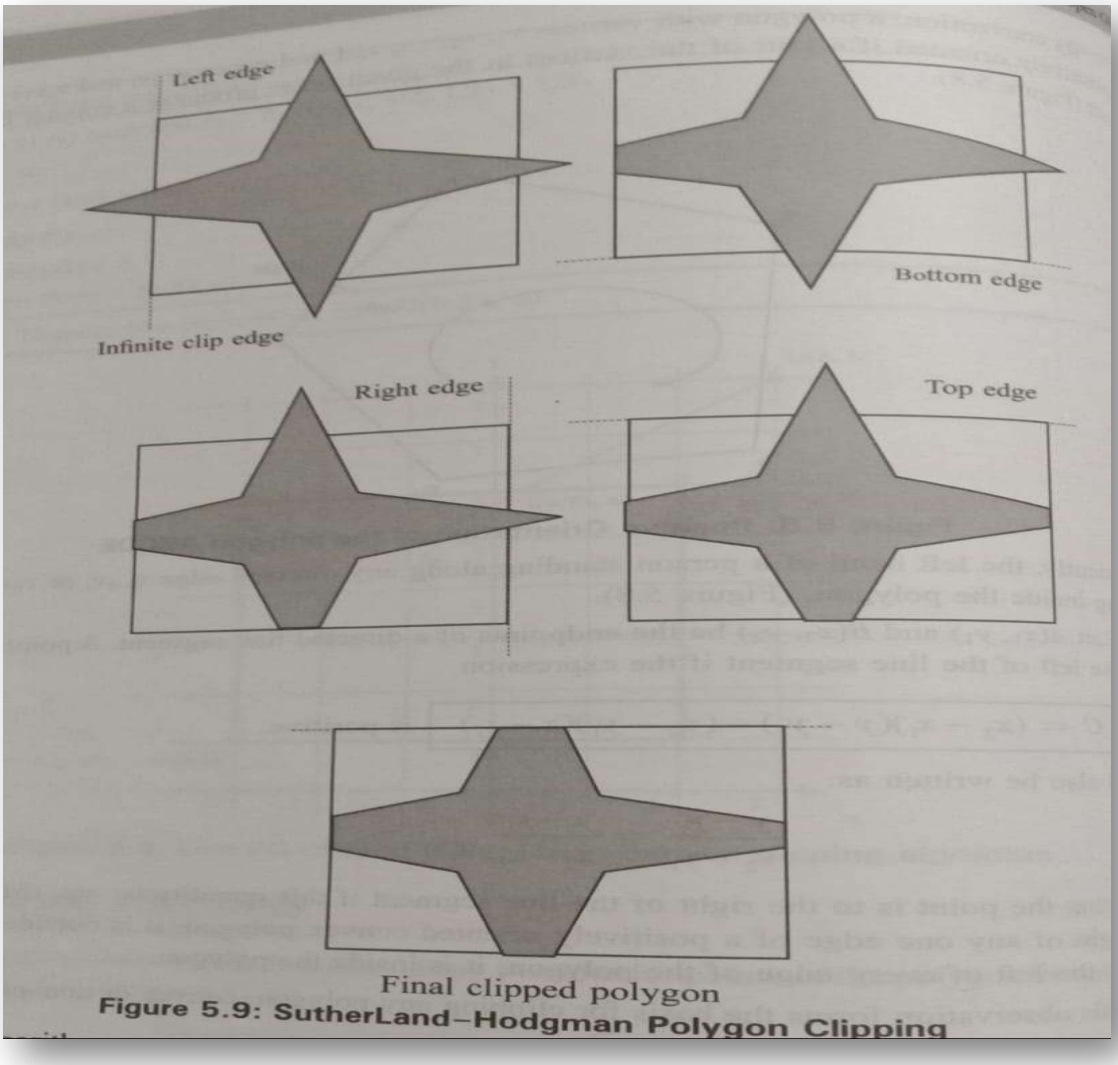


Figure 5.9: SutherLand–Hodgman Polygon Clipping

Case 1: Vertex v_1, v_2 both inside, v_2 is the output.

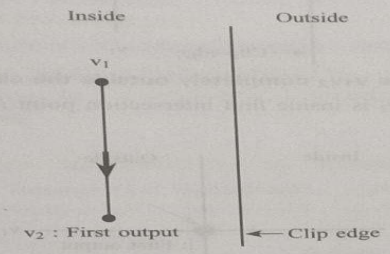


Figure 5.10: Line v_1v_2 completely inside the clip edge (case 1)

Case 2: Vertex v_1 is inside, v_2 is outside, find the intersection point i against that edge and output is *i*.

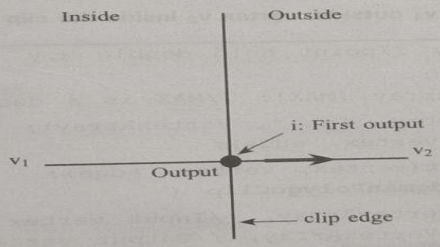


Figure 5.11: Vertex v_1 inside, vertex v_2 outside clip edge (case 2)

Case 3: Vertex v_1, v_2 both outside, no operation.

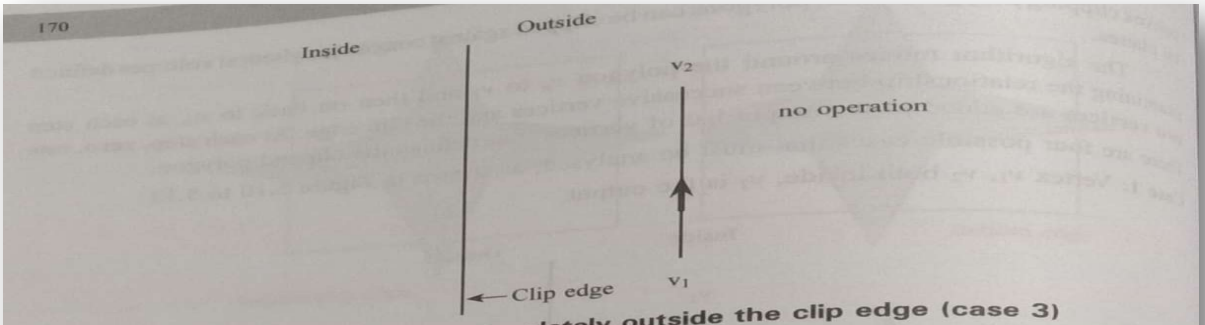


Figure 5.12: Line v_1v_2 completely outside the clip edge (case 3)

Case 4: Vertex v_1 is outside, v_2 is inside find intersection point *i* against that edge and outputs are both *i* and v_2.

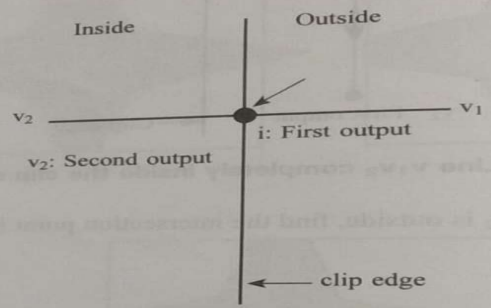


Figure 5.13: vertex v_1 outside, vertex v_2 inside the clip edge (case 4)

ALGORITHM:

1. Read coordinates of all vertices of the polygon.
2. Read coordinates of the clipping window
3. Consider the left edge of the window
4. Compare the vertices of each edge of the polygon, individually with the clipping plane
5. Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary discussed earlier.
6. Repeat the steps 4 and 5 for remaining edges of the clipping window. Each time the resultant list of vertices is successively passed to process the next edge of the clipping window.
7. Stop.

ADVANTAGES:

- ✓ **Versatility:** Can clip any type of convex or concave polygon against an arbitrary convex clipping region.
- ✓ **Geometric Accuracy:** Preserves the shape and topology of the original polygon during clipping.
- ✓ **Efficiency:** Operates in linear time complexity with respect to the number of vertices.
- ✓ **Handles Multiple Clipping Regions:** Can handle multiple clipping regions sequentially.

DISADVANTAGES:

- ✓ **Degenerate Cases:** Susceptible to issues with polygons entirely outside or with edges parallel to clipping boundaries.
- ✓ **Concave Polygons:** Requires polygon decomposition into convex sub-polygons for concave cases.

- ✓ **Processing Order:** Result can be affected by the order in which vertices are processed.
- ✓ **Inefficiency for Convex Polygons:** May not offer significant advantages over simpler methods for convex polygons.

APPLICATIONS:

1. **Computer-aided design (CAD):** It is employed in CAD systems for accurate polygon clipping against specified regions or boundaries, ensuring precise representation and visualization of complex shapes.
2. **GIS (Geographic Information Systems):** The algorithm is used for clipping and displaying map features within defined areas, enabling efficient visualization of geographical data.
3. **Polygon-based simulations:** It finds applications in simulations involving polygonal entities, such as physics simulations, architectural modeling, or virtual environments
4. **Image processing:** It finds applications in image processing tasks where polygonal regions need to be clipped against a specified area, enabling precise extraction or manipulation of image regions.

CONCLUSION:

This report concludes that both the Cohen-Sutherland line clipping algorithm and the Sutherland-Hodgman polygon clipping algorithm are valuable tools in computer graphics. The Cohen-Sutherland algorithm excels in efficiently clipping line segments against rectangular clipping windows, while the Sutherland-Hodgman algorithm is versatile in clipping convex or concave polygons against arbitrary convex clipping regions. Understanding their principles, advantages, disadvantages, and applications is crucial for selecting the appropriate algorithm based on specific requirements in computer graphics applications.

REFERENCES:

[1] Edward Angel: Interactive Computer Graphics- A Top-Down approach with OpenGL, 5 th edition. Pearson Education, 2008.

[2] M M Raikar & Shreedhara K S Computer Graphics using OpenGL, Cengage publication.