Appl. Math. J. Chinese Univ.
2022, 37(1): 147-158

An efficient algorithm to clip a 2D-polygon against a rectangular clip window

Sushil Chandra Dimri Umesh Kumar Tiwari Mangey Ram

Abstract. Polygon clipping is of great importance in computer graphics. One of the popular algorithms to clip a polygon is Cohan–Sutherland Hodgeman algorithm which is based on line clipping. Cohan–Sutherland Hodgeman algorithm clips the polygon against the given rectangular clip window with the help of line clipping method. Cohan–Sutherland algorithm requires traversing the polygon in anti clockwise direction (positive orientation). In this work we propose an efficient polygon clipping algorithm against a rectangular clip window. Proposed algorithm uses parametric representation of polygon edges. Using the concept of point clipping, we can find required intersection points of edges of polygon with clip window boundaries. Well suited numerical illustrations are used to explain the proposed polygon clipping method. The proposed algorithm is computationally less expensive and comprehensive.

§1 Introduction

Clipping is treated as crucial operation in computer graphics and related technologies. Clipping can be categorised as point clipping, line clipping, area clipping, curve clipping or text clipping. Clipping is one of the basic operations that always attracts researchers and practitioners for the development of new, innovative and efficient algorithms. Clipping methods basically focus on identifying a particular part of a substance. That identified part may belong to the inner or outer area of the substance. Most of the clipping algorithms available in literature are based on comparisons and computations. These comparisons and computations not only increase the computational complexity of the algorithms but sometimes make implementation less effective. Most of the polygons clipping methods are used for convex polygons and these convex polygons are restricted to rectangular in shape.

The term clip window is used to denote the rectangular parallelogram or polygon clipping object. We can use any type of clipping window according to the requirement. Clipping denotes the process of discarding unnecessary objects from the viewing plane. This unnecessary object may be point, line, or polygon. In this paper our focus is on polygon clipping.

Received: 2021-03-29. Revised: 2021-09-07.
MR Subject Classification: 68U05.
Keywords: point clipping, polygon clipping, clip window, line segment, computational complexity.
Digital Object Identifier(DOI): https://doi.org/10.1007/s11766-022-4556-0.

Polygon clipping is one of the most useful operations in computer graphics. Applications of polygon clipping are not limited to computer graphics only. Now days polygon clipping is used in geographic information system domains, VLSI circuit designs, computational sciences, and similar applications. In Figure 1, a rectangular clip window is shown where boundaries are defined on 2D-XY plane as, $x = x_{min}$, $x = x_{max}$, $y = y_{min}$ and $y = y_{max}$.

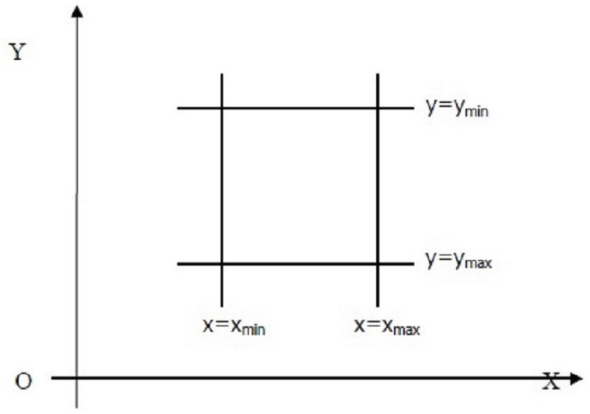

Figure 1. A clip window.

Eminent researchers proposed efficient methods for point clipping, line clipping and polygon clipping. Most of the clipping methods are inspired or directly drawn from renowned algorithms like Cohen Sutherland [1], Liang and Barsky [2], [3] and Cyrus Beck [4]. Most of the methods available in literature have O (N) computational time complexity. Whereas the method suggested by Rappaport [5] and Skala [6], [7] have O(log N) computational time complexity. Structure of the proposed work is as follows: Section 1 discusses the introduction of clipping especially in the context of polygons. Section 2 discusses related works done by eminent researchers in the field of clipping. Section 3 proposes a new clipping algorithm and discusses its computational complexity. Section 4 presents a numerical illustration to implement the proposed algorithm and finally conclusion of the work is given in Section 5.

§2 Related work

In this section we discuss some methods and algorithms available for clipping different shapes including polygon, line and circle. Wijeweera et al. [8] proposed a line clipping method for a convex polygon having computational time complexity of O(N). Authors argued that their method can be used for any type of polygon having any number of vertices and the vertices can follow any order. They compared their algorithm with existing and well known clipping algorithms including Cyrus Beck, ECB, Rappaport, and Skala. Klamer [9] defines an algorithm for clipping couple of concave polygons. Klamer used the concept of labelling of polygon edges to classify the resultant polygons in the set. In his algorithm, Klamer classified three types of edges in every polygon, inside, shared and outside. On the basis of edges, minimal polygons are

identified. Inputted 2D polygons in the clipping algorithm analyses the intersection and minus of polygon sets. Klamer also suggested some improvements in the average time complexity.

Foster [10] proposed an extended version of GreinerCHormann clipping algorithm [11] to improve the shortcomings of degenerate intersection in polygons. Foster uses the work of Kim and Kim [12] as the base and proposes the extensions in GreinerCHormann algorithm. Foster divided his algorithm into three phases; intersection, labelling and tracing. Authors work focuses on the detection of degenerate intersections and providing mechanisms to handle them properly in the first phase that is intersection phase. Foster categorizes intersections into three types; X-intersections, T-intersections, and V-intersections. Further in the labelling phase, three cases, left turn, straight, and right turn are identified.

Liua et al. [13] proposed a solution for clipping of concave polygons containing holes. In their algorithm authors used single-linked list as the data structure. They verified their work on different sets of polygons. Their algorithm consists of three main steps; i) intersection is identified and entry/exit points are stored, ii) line clipping is applied on edges and iii) clipped polygon is achieved by traversal of single-linked list. Authors argued that their clipping algorithm requires only few arithmetic operations with basic data structure that results in fast execution of the algorithm. Raja [14] defined an algorithm for clipping natural images. His work is based on line clipping algorithms of Cohen Sutherland, Liang Barsky and Nicholl Lee [15]. Raja also used the polygon clipping method suggested by Sutherland and Hodgman. Raja analysed his clipping algorithm on three basic parameters; time complexity, space complexity and the accuracy of the object. Author argued that algorithm performs better in all these three parameters. Elliriki et al. [16] proposes a mathematical model based on integral computations to evaluate the point of intersection. According to authors, proposed algorithm does not need many computations to find the point of intersection. They analysed the algorithm using three different cases; "positive slope lines, negative slope lines and portion of line inside window apart from completely accepted and rejected lines" [16].

Dimitrios and Vasileios [17] introduced a 2D line clipping for a window having rectangular in shape. Authors claim that this algorithm is better in performance, simple and easy to implement in comparison to other algorithms. Their algorithm consists of three major steps. First step checks that if the end points lie outside the clipping window or it lies in the same region. If any of the two are true then the line is discarded and algorithm halts. Second step is used to compare the coordinates of points along with the clip window boundary. Finally third step checks that whether the modified points lie inside the clipping area, if true, then the line is drawn between the new points. In order to prove the efficiency of the algorithm they compared it with six algorithms.

Dimri [18] suggested an algorithm that presents a line in parametric form. This parametric representation denotes line as infinite point collection. Proposed algorithm checks coordinates with the boundary as well as the intersection of boundary. During the process of checking, the parameter value u is stored. Value of the parameter u lies within the range 0 and 1. Any point outside this range is simply discarded. According to the author this algorithm can be extended for polygon clipping.

Sharma and Jasmeen [19] proposed a polygon clipping algorithm for self intersecting and

multi polygons. Their algorithm uses affine transformation that discards degeneracies. They applied their algorithm on datasets and claim that the algorithm performs better in comparison to available algorithms. Datasets used in the algorithm are real time datasets. They used stack based approach to spit polygons having self-intersections and multi-polygons. To implement the algorithm authors used JAVA programs and NetBeans IDE. They conducted 480 tests having convex and concave polygons with varying number of vertices.

Dingding et al. [20] proposed a clipping algorithm for non-intersecting polygon on vector graphics. Authors argued that their algorithm is comparatively fast and can process huge data in limited period of time and consumes less space. They proposed clipping algorithm in three steps. In first step they discard the non intersecting boundary with respect to vector graphics as well as compute effective intersections. In second step they partition the graphics into parts. Finally in third step the algorithm checks that each part belongs to the inner boundary. Authors argue that the proposed algorithm is simple and comparatively fast. Joshi et al. [21] proposes a modified algorithm for clipping loose polygons. Their algorithm is based on the modification of renowned Weiler-Athertons algorithm. Authors suggested some modifications to deal with clipping not only at intersections but also to deal with loose polygons at disjoints, contained and surroundings.

Minghua et al. [22] proposed a circle clipping algorithm under hexagonal grid system. They proposed their algorithm as a solution for poor time-effective and poor efficiency of polygon clipping algorithms available in the literature. Their approach is to translate coordinates from cartesian to hexadecimal grid. After translating coordinates, next step is to check the intersection with the help of vector operation. This checking is performed with the help of geometric relations and tries to identify points of intersection. This process results in finding the arc. In this way the clipping is achieved. Authors claim that this method consumes comparatively low time and is effective.

Mingjun et al. [23] presented a line clipping algorithm and further extended it for polygon clipping. They used Java applets to implement and compare their proposed algorithm. Authors have not used the concept of the intersection points rather its sorting. They have implanted the line clipping concept to clip a polygon. According to authors, clipping edges categorised as "no edge of subject polygon passes through the vertices of the clip polygon; and one or more edges of subject polygon pass through the vertices of the clip polygon" [23]. Authors had given computational complexity of their algorithms for worst, best average cases. Further, they applied proposed algorithms on random test sets.

One of the most popular algorithms for polygon clipping is Sutherland–Hodgeman polygon clipping which is based on line clipping and generates vertex output list at every step of clipping. This property makes the algorithm computationally challengeable in terms of time and space complexity.

§3 Proposed Polygon Clipping Algorithm

A polygon can be considered as a combination of different edges, for example, a triangle has three edges. To describe a polygon we need vertex points. For an 'n' vertex polygon we

have P_1, P_2, $\cdots$, P_n, where P_i P_{i+1} are the edges of polygon. Figure 2 shows a polygon with 8 vertices.

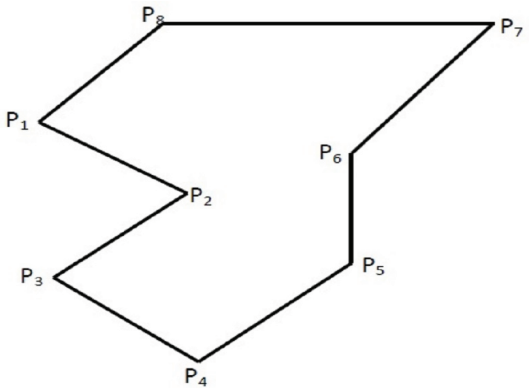

Figure 2. A Polygon with 8 vertices.

Each edge P_i P_{i+1} of a polygon is a line segment. An 'n' vertex polygon has 'n' edges that are line segments.

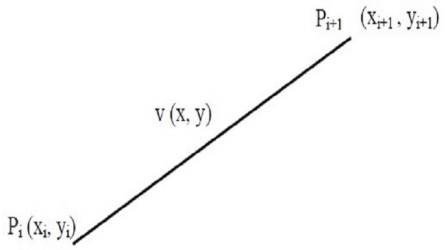

Figure 3. A Line Segment.

Line segment is continues geometrical point. But the parametric equation of a line segment changes it into collection of infinite points that lie on the line. The parametric equation for the line segment shown in Figure 2, can be given as-

x= x_i + u.(x_{i+1} - x_i), and y= y_i + u.(y_{i+1} - y_i), where 0 $\leq$ u $\leq$ 1

or x= x_i+u.Δx, and y= y_i+ u.Δy(v (x, y) is a varying point of line P_i P_{i+1})

Each edge of polygon can be expressed by parametric equation. For each clip window (x_{min}, y_{min}, x_{max}, y_{max}), a point (x, y) lies in or on clip window if it satisfies the following inequalities-

x_{min} $\leq$ x $\leq$ x_{max}, and y_{min} $\leq$ y $\leq$ y_{max}.

The point (x, y) be in or on clip window. To clip a polygon with respect to a given clip window we clip each edge of the polygon one by one. Each edge of polygon can be identified with the help of coordinates of end points. For an edge P_i P_{i+1} of the polygon,

$$x_{min} \leq x_i, x_{iH} \leq x_{max} \quad and \quad y_{min} \leq y_i, y_{iH} \leq y_{max} \tag{1}$$

Then the edge is completely inside the clip window otherwise edge is either out of clip window or clipping coordinates. Again if,

$$\begin{aligned} &x_i, x_{i+1} < x_{min}, \quad y_i, y_{i+1} < y_{min}, \quad or \\ &x_i, x_{i+1} < x_{max}, \quad y_i, y_{i+1} < y_{max} \end{aligned} \tag{2}$$

The line segment (edge) $P_i\ P_{i+1}$ is outside the clip window and hence discarded. In this case both eq. (1) or eq. (2) is true and then the edge $P_i\ P_{i+1}$ is clipping coordinate.

Equation of edge $P_i\ P_{i+1}$ can be given as-

x= x_i + u.(x_{i+1} - x_i) and y = y_i + u.(y_{i+1} - y_i), where $0 \leq u \leq 1$.

The clip window has 4 boundaries: if $\{x_i < x_{min}$ / boundary x= $x_{max}\}$, then at point of intersection with u (x_{i+1} - x_i = x_{min} - x_i), or u=$\{x_{min}$ - x_i / (x_{i+1} - x_i)$\}$.

Similarly for the same edge $P_i\ P_{i+1}$, we will compute parameter values as-

u^{ii} for x = x_{max} (right boundary),

u^{iii} for y = y_{min} (bottom boundary), and

u^{iv} for y = y_{max} (upper boundary).

Then ignore those values of 'u' that are either less than 0 or greater than 1. These values of u denoted as u^k belong to [0, 1]. We compute the point of intersection of edge with corresponding boundary using equation:

[x' = x_i + u^k . (x_{i+1} - x_i)] / [y' = y_i + u^k . (y_{i+1} - y_i)], where $0 \leq u^k \leq 1$.

If point (x', y') satisfies the 'm' equality:

$x_{min} \leq$ x $\leq x_{max}$ and $y_{min} \leq$ y $\leq y_{max}$.

Then preserve only the intersection point (x', y') otherwise discard it. We repeat these steps for other points of intersection one by one and collect the point of intersections. Once all the edges have been processed, connect all points of intersection and ultimately gives the clipped polygon.

3.1 Algorithm: Polygon Clipping

In this section we propose an algorithm for polygon clipping that takes a rectangular clip window as an input and produces the clipped window.

Input: Rectangular clip window (x_{min}, y_{min}, x_{max}, y_{max}) and a polygon (P_1, P_2, $\cdots$, P_n)

Output: The clipped polygon

S_1: Read- (x_{min}, y_{min}, x_{max}, y_{max})

S_2: Read- (P_1, P_2, $\cdots$, P_n)

For each edge $P_i\ P_{i+1}$ of polygon

S_3: If either x_i, $x_{i+1} < x_{min}$ or y_i, $y_{i+1} < y_{min}$

Print- edge $P_i\ P_{i+1}$ is out of clip window, discard both end points.

If $x_{min} \leq x_i$, $x_{i+1} \leq x_{max}$ and $y_{min} \leq y_i$, $y_{i+1} \leq y_{max}$

Print- edge $P_i\ P_{i+1}$ is inside the clip window, keep both end points of edge.

Else the edge is clipping candidate.

The parametric equation of edge is-

x = x_i + u.(x_{i+1} - x_i) and y = y_i + u.(y_{i+1} - y_i), where $0 \leq u \leq 1$

S_4: Compute the value of parameter 'u' for all four boundary x = x_{min}, y = y_{min}, x = x_{max},

y = y_{max} of clip window; ignore those values of 'u' which are not in closed set [0, 1].

For accepted values of 'u' compute the point of intersection of edge with corresponding boundary.

S_5: Now if these points of intersections satisfy inequality-

$x_{min} \leq x_i \leq x_{max}$ and $y_{min} \leq y_i \leq y_{max}$

Retain the point of intersection otherwise discard them.

S_6: Connect all the retaining points of each other.

When all these edges of polygon are processed, connect retained points and the figure obtained is clipped polygon.

Else repeat all the steps S_1 to S_6 for next consecutive edge.

3.2 Computational Complexity

The computational complexity of proposed algorithm is low in comparison to other polygon clipping algorithms. Proposed algorithm computes value of parameter 'u' for point of intersections and then easily compute the coordinates of point of intersection. Then this algorithm tests these points in an equality to check whether these points could be retained or not. Once we get the set of retained points we get the clipped polygon. The Sutherland-Hodgeman algorithm only determines the retaining points with respect to boundaries of clip window but not computes the coordinates of points of intersection for which we need to apply a line clipping algorithm separately. On the other hand a 2D polygon clipping algorithm suggested by WeilerCAtherton form a list of point of intersections. This algorithm requires some preconditions to be fulfilled like clock wise orientation and condition for candidate polygon.

§4 Numerical Illustration

Consider a clip window given by boundaries $x_{min} = 2$, $x_{max} = 10$, $y_{min} = 4$, $y_{max} = 8$, as shown in Figure 4.

In Figure 4, polygon (P_1, P_2, P_3, P_4, P_5, P_6) has 6 edges and A, B, C, D denotes the clip window. We apply the proposed algorithm to clip the given polygon. We take edges of polygon one by one starting from edge P_1P_2. Parametric equation of this edge can be given as-

x = 1 + u (7 - 1) = 1 + 6u,

x = 6 + u (6 - 6) = 6,

$0 \leq u \leq 1$, $\Delta y = 0$, therefore edge P_1P_2 is parallel to x axis.

$x_{min} = 2$, $x_{max} = 10$ and $y_{min} = 4$, $y_{max} = 8$.

Point $P_1(1, 6)$ do not satisfy inequality x_{min} (2) $\leq 1 \leq x_{max}$ (10), therefore P_1 is not inside the clip window. Point $P_2(7, 6)$ satisfies the inequality $x_{min} \leq 7 \leq x_{max}$ and $y_{min} \leq 6 \leq y_{max}$. So the point $P_2(7, 6)$ is inside the clip window, and edge P_1P_2 is clipping coordinate. The parametric equation of P_1P_2 is given as-

x = 1 + 6u, where $0 \leq u \leq 1$,

y = 6

Putting x = x_{min} = 2

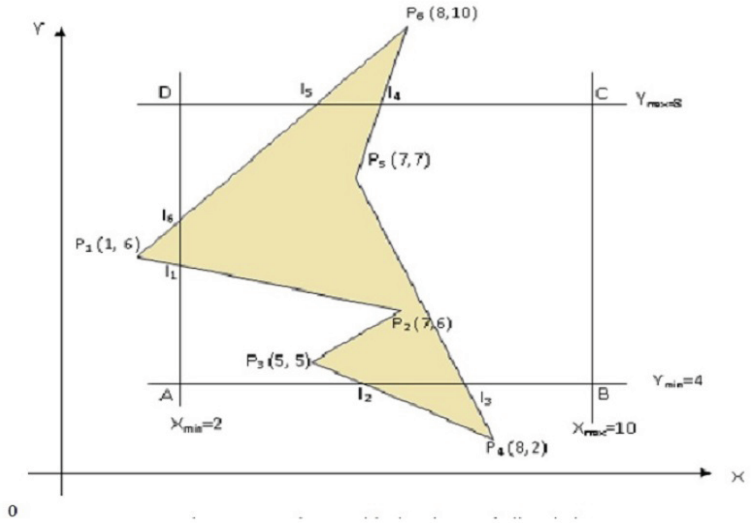

Figure 4. A polygon with six edges and clip window.

2 = 1 + 6u or 6u = 2 – 1 = 1 {u = 1/6}
x = x_{max}, that is, 10 = 1 + 6u
6u = 9 ⇒ u = 9 / 6 (discard)
Point of intersection
$I_1 \equiv (1 + 6 \times 1 / 6, 6) = (2, 6)$, keep I_1
Taking the edge P_2P_3: P_2 (7, 6) and P_3(5, 5) both end points of the edge satisfies the inequality, that is 2 ≤ x ≤ 10 and 4 ≤ y ≤ 8. Hence this line P_2P_3 (edge) is inside the clip window. We keep both the vertex points P_2 and P_3.
Now edge P_3P_4, point P_3 is inside the clip window since satisfies inequality, 2 ≤ x ≤ 10 and 4 ≤ y ≤ 8.
But point P_4(8, 2) does not satisfy the inequality. So edge P_3P_4 is clipping candidate.
Parametric equation of P_3P_4 can be given as-
x = 5 + 4 (3), or x = 5 + 3u
y = 5 + 4 (-3), or y = 5 - 3u, where 0 ≤ u ≤ 1
Now with,
Left boundary: x = x_{min} = 2,
2 = 5 + 3u - 3 = 3u → u = -1 (discard)
Right boundary: x = x_{max} = 10
10 = 5 + 3u → 3u = 5 → u = 5/3 (discard)
Bottom boundary: y = y_{min} =4
4 = 5 - 3u → -1 = -3u → u = 1/3
Upper boundary: y = y_{max} = 8
8 = 5 - 3u → 3 = -3u → u = -1 (discard)
So valid value of u = 1/3 for bottom and point of intersection I_2 is given as-

$I_2 \equiv (5 + 3 \times 1 / 3, 5 - 3 \times 1 / 3) \equiv (6, 4)$
For edge P_4P_5, end point P_4 is out of clip window but $P_5(7, 7)$ satisfies the inequality $2 \leq x \leq 10$ and $4 \leq y \leq 8$.
Point $P_5(7, 7)$ is inside the clip window and therefore P_4P_5 is clipping candidate.
Now parametric equation of P_4P_5 can be given as-
$x = 8 + u(-1) = 8 - u$ and $y = 2 + 4(5) = 2 + 5u$, where $0 \leq u \leq 1$
Left boundary: $x = x_{min} = 2$
$2 = 8 - u \rightarrow u = 6$ (discard)
Right boundary: $x = x_{max} = 10$
$10 = 8 - u \rightarrow u = -2$ (discard)
Bottom boundary: $y = y_{min} = 4$
$4 = 2 + 5u \rightarrow u = 2/5$ (accept)
Upper boundary: $y = y_{max} = 8$
$8 = 2 + 5u \rightarrow u = 6/5$ (discard)
So $u = 2/5$ only for bottom boundary. Point of intersection I_3 with bottom boundary ($u=2/5$) can be calculated as-
$I_3 \equiv (8 - 2/5 , 2 + 5 \times 2/5) \equiv (38/5, 4) \equiv (7.6, 4)$
Keep I_3 and I_5.
Taking the edge P_5P_6: Point P_5 is inside the clip window but the point $P_6(8,10)$ does not satisfy the inequality $2 \leq x \leq 10$ and $4 \leq y \leq 8$. Point P_6 is out of clip window, therefore P_5P_6 is clipping candidate.
Parametric equation of P_5P_6 is given as-
$x = 7 + 1u$ and $y = 7 + 3u$, where $0 \leq u \leq 1$.
Left boundary: $x = 2$,
$2 = 7 + u \rightarrow u = -5$ (discard)
Right boundary: $x = 10$,
$10 = 7 + u \rightarrow u = 3$ (discard)
Bottom boundary: $y = 4$,
$4 = 7 + 3u \rightarrow u = -1$ (discard)
Upper boundary: $y = 8$,
$8 = 7 + 3u \rightarrow u = 1/3$ (accepted)
Now point of intersection I_4 with upper boundary $y = 8$ is calculated as-
$I_4 \equiv (7 + 1/3, 7 + 3 \times 1/3) = (22/3, 8)$
$I_4 \equiv (7.33,8)$
We keep I_4.
Taking edge P_6P_1 of polygon: The parametric equation of P_6P_1 is-
$x = 8 + u(1 - 8) = 8 - 7u$
and $y = 10 + u(6 - 10) = 10 - 4u$, where $0 \leq u \leq 1$
Left boundary: $x = 2$
$2 = 8 - 7u \rightarrow u = 6/7$ (accepted)
Right boundary: $x = 10$
$10 = 8 - 7u \rightarrow u = -2/7$ (discard)

Bottom boundary: y = 4

$4 = 10 - 4u \rightarrow u = 6/4$ (discard)

Upper boundary: y = 8

$8 = 10 - 4u \rightarrow u = 2/4 = 1/2$ (accepted)

Now point intersection with upper boundary can be given as-

$I_5 \equiv (8 - 7 \times 1/2, 10 - 4 \times 6/7) = (2, 46/7) \equiv (2, 6.57)$

We keep both I_5 and I_6.

Now all the edges are processed and obtained clipped polygon is (I_1 P_2 P_3 I_2 I_3 P_5 I_4 I_5 I_6). Coordinates of these points are known to us and we join these points to get the clipped polygon.

Figure 5 shows the clipped polygon as the output of the proposed clipping method. This proposed clipping takes less time to compute and have comparatively lesser number of comparisons.

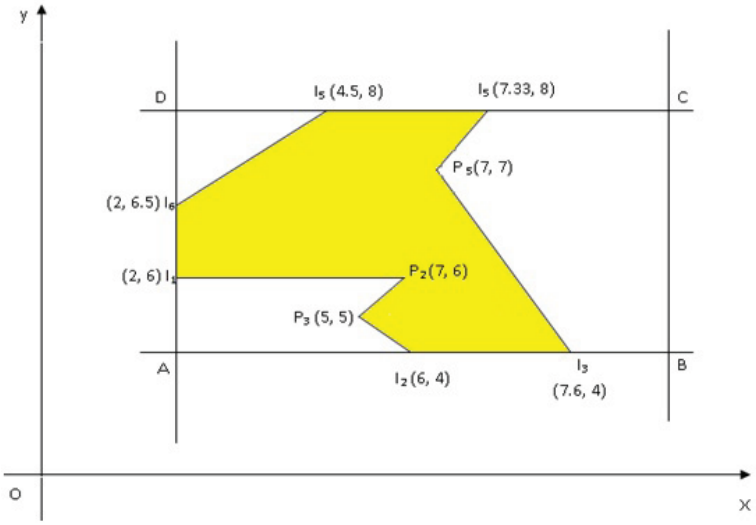

Figure 5. The clipped polygon.

§5 Conclusion

Numbers of algorithms are available in literature for polygon clipping. Most of these existing algorithms use the concept of line clipping. One of the most popular algorithms for polygon clipping is CohanCSutherland algorithm which is also based on line clipping and generates a vertex output list. Further these line clipping algorithms are used to determine the coordinates of vertices. If the number of edges in the polygon is high, the mathematical complexity of the algorithm will be very high. The proposed algorithm is a comprehensive algorithm for 2D polygon clipping. Proposed polygon clipping algorithm computes the coordinates of point of intersection of edges. This algorithm also computes the boundaries of clip window and

identifies the real intersection points and retains them to obtain the clipped polygon. Proposed algorithm uses parametric equation of edges of polygon and the concept of point clipping. Proposed algorithm is mathematically less expensive and can be useful in graphics and other applications. The limitation of the algorithm is that it is only applicable for 2D rectangular clip window.

References

[1] D Cohen. *Incremental methods for computer graphics*, PhD Thesis, University of Harvard, Massachusetts, 1969.

[2] Y D Liang, B A Barsky. *An analysis and algorithms for polygon clipping*, CACM 26, 1983: 868-876.

[3] Y D Liang, B A Barsky. *A new concept and method for line clipping*, ACM Transactions on Graphics, 1984, 3(1): 1-22.

[4] M Cyrus, J Beck. *Generalized two and three-dimensional clipping*, Computers and Graphics, 1978, 3(1): 23-28.

[5] M Rappaport. *An efficient algorithm for line and polygon clipping*, The Visual Computer, 1991, 7(1): 19-28.

[6] V Skala. *An efficient algorithm for line clipping by convex polygon*, Computers and Graphics, 1993, 17(4): 417-421.

[7] V Skala. *O (lg N) line clipping algorithm in E2*, Computers and Graphics, 1994, 18(4): 517-424.

[8] K R Wijeweera, S R Kodituwakku, M P Chamikara. *A novel and efficient approach for line segment clipping against a convex polygon*, Ruhuna Journal of Science, 2019, 10(2): 161-173.

[9] K Schutte. *An edge labelling approach to concave polygon clipping*, ACM Transactions on Graphics, 1995: 1-10.

[10] E L Foster, K Hormann, R T Popa. *Clipping simple polygons with degenerate intersections*, Computers and Graphics, 2019, X(2).

[11] G Greiner, K Hormann. *Efficient clipping of arbitrary polygons*, ACM Transactions on Graphics, 1998, 17(2): 71-83.

[12] D H Kim, M J Kim. *An extension of polygon clipping to resolve degenerate cases*, Computer Aided Design Appl., 2006, 3: 447-456.

[13] Y K Liu, X Q Wang, S Z Bao, M Gomboši, B Žalik. *An algorithm for polygon clipping, and for determining polygon intersections and unions*, Computers and Geosciences, 2007, 33: 589-598.

[14] S P Raja. *Line and Polygon Clipping Techniques on Natural Images: A Mathematical Solution and Performance Evaluation*, International Journal of Image and Graphics, 2019, 19(2).

[15] T M Nicholl, T Lee, R A Nicholl. *An efficient new algorithm for 2D line clipping: Its development and analysis*, in Proc SIGGRAPH '87, Comput Graph, 1987, 21(4): 253-262.

[16] E Mamatha, C Reddy, K Anand. *An Efficient Line Clipping Algorithm in 2D Space*, The International Arab Journal of Information Technology, 2019, 16(5).

[17] D Matthes, V Drakopoulos. *Another Simple but Faster Method for 2D Line Clipping*, International Journal of Computer Graphics and Animation, 2019, 9(1/2/3).

[18] S C Dimri. *A Simple and Efficient Algorithm for Line and Polygon Clipping in 2-D Computer Graphics*, International Journal of Computer Applications, 2015, 127(3).

[19] M Sharma, J Kaur. *An Improved Polygon Clipping Algorithm Based on Affine Transformation*, S C Satapathy, et al, in Proceedings of the Second International Conference on Computer and Communication Technologies, Advances in Intelligent Systems and Computing, 2016, 379.

[20] D Yang, S Chen, Q Yang, Y Hu. *A Clipping Algorithm on Vector Graphics Based on Non-intersect Polygon Boundary*, IEEE, 2019, 854-859.

[21] T Joshi, P Badoni, A Aggarwal. *Modification of Weiler-Atherton Algorithm to Address Loose Polygons*, Journal of Scientific and Industrial Research, 2019, 78: 771-774.

[22] M Cao, H Zhang, C Zhou, Y Sun, H Yu. *Vector Circle Clipping Algorithm Based on Polygon Window of Hexagonal Grid System*, IOP Conf Series: Journal of Physics: Conf Series, 2019, 1288(012006).

[23] M Zhang, C L Sabharwal. *An Efficient Implementation of Parametric Line And Polygon Clipping Algorithm*, ACMSAC'02, 2002, 11-14.

Department of Computer Science and Engineering, Graphic Era (Deemed to be University), Dehradun, Uttarakhand, India.

Email: umeshtiwari22@gmail.com