School of Computing Science and Engineering

Program: B.C.A.

Course Code: BCAS3003

Course Name: Computer Graphics

Vision

To be known globally as a premier department of Computer Science and Engineering for value-based education, multidisciplinary research and innovation.

Mission

- **M1:** Developing a strong foundation in fundamentals of computing science with responsiveness towards emerging technologies.
- **M2:** Establishing state-of-the-art facilities and adopt education 4.0 practices to analyze, develop, test and deploy sustainable ethical IT solutions by involving multiple stakeholders.
- **M3:** Establishing Centers of Excellence for multidisciplinary collaborative research in association with industry and academia.

Course Outcomes (COs)

CO Number	Title
CO1	Describe the fundamental concepts of Computer Graphics. (K1)
CO2	To demonstrate with the relevant mathematics of computer graphics, ex. line, circle and ellipse drawing algorithms. (K3)
CO3	To understand the attributes of output primitives of Graphics. (K2).
CO4	Apply simple and composite transformation on graphic objects/elements in two dimensions. (K3).
CO5	Analyze two dimensions modeling and clipping techniques. (K4).
CO6	List out the various contemporary research areas and tool in graphics domain. (K2).

Course Prerequisites

- Knowledge of Mathematics
- Fundamental knowledge of Computer

Syllabus

Unit 5 – Two Dimensional Viewing (6 hours)

- **Viewing Pipeline**
- **Viewing Transformation**
- **Line Clipping**
- **Cohen Sutherland Line Clipping**
- **Liang Barsky Line Clipping**
- **Polygon Clipping: Sutherland-Hodgman Polygon Clipping**
- **Weiler Amerton Polygon Clipping**

Recommended Books

Text books

- D. Hearn, P. Baker, "Computer Graphics - C Version", 2nd Edition, Pearson Education, 1997

Reference Book

- Heam Donald, Pauline Baker M: "Computer Graphics", PHI 2nd Edn. 1995.
- Harrington S: "Computer Graphics - A Programming Approach", 2nd Edn. Mc GrawHill.
- Shalini Govil-Pai, Principles of Computer Graphics, Springer, 2004

Additional online materials

- Coursera - https://www.coursera.org/learn/fundamentals-of-graphic-design
- https://www.youtube.com/watch?v=fwzYuhduME4&list=PLE4D97E3B8DB8A590
- NPTEL - https://nptel.ac.in/courses/106/106/106106090/
- https://www.coursera.org/learn/research-methods
- https://www.coursera.org/browse/physical-science-and-engineering/research-methods

Window:

1.A world-coordinate area selected for display is called a window.

2.In computer graphics, a window is a graphical control element.

3.It consists of a visual area containing some of the graphical user interface of the program it belongs to and is framed by a window decoration.

4.A window defines a rectangular area in world coordinates. You define a window with a GWINDOW statement. You can define the window to be larger than, the same size as, or smaller than the actual range of data values, depending on whether you want to show all of the data or only part of the data.

Viewport:

1.An area on a display device to which a window is mapped is called a viewport.

2.A viewport is a polygon viewing region in computer graphics. The viewport is an area expressed in rendering-device-specific coordinates, e.g. pixels for screen coordinates, in which the objects of interest are going to be rendered.

3.A viewport defines in normalized coordinates a rectangular area on the display device where the image of the data appears. You define a viewport with the GPORT command. You can have your graph take up the entire display device or show it in only a portion, say the upper-right part.

Viewing Transformation

- **World coordinate System (WCS)**
- **Normalized Coordinate System (NCS)**
- **Screen Coordinate System (SCS)**
- **Physical Device Coordinate System (PDCS)**

- **Normalized Transformation**
- **Workstation Transformation**

Normalized Screen Coordinate (Normalized Device Coordinate System)

Clipping

Normalized Point on Window $\left(\frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}}, \frac{Y_w - Y_{wmin}}{Y_{wmax} - Y_{wmin}} \right)$

Normalized Point on Viewport $\left(\frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}}, \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}} \right)$

For X Coordinate

$$\frac{X_w - X_{wmin}}{X_{wmax} - X_{wmin}} = \frac{X_v - X_{vmin}}{X_{vmax} - X_{vmin}}$$

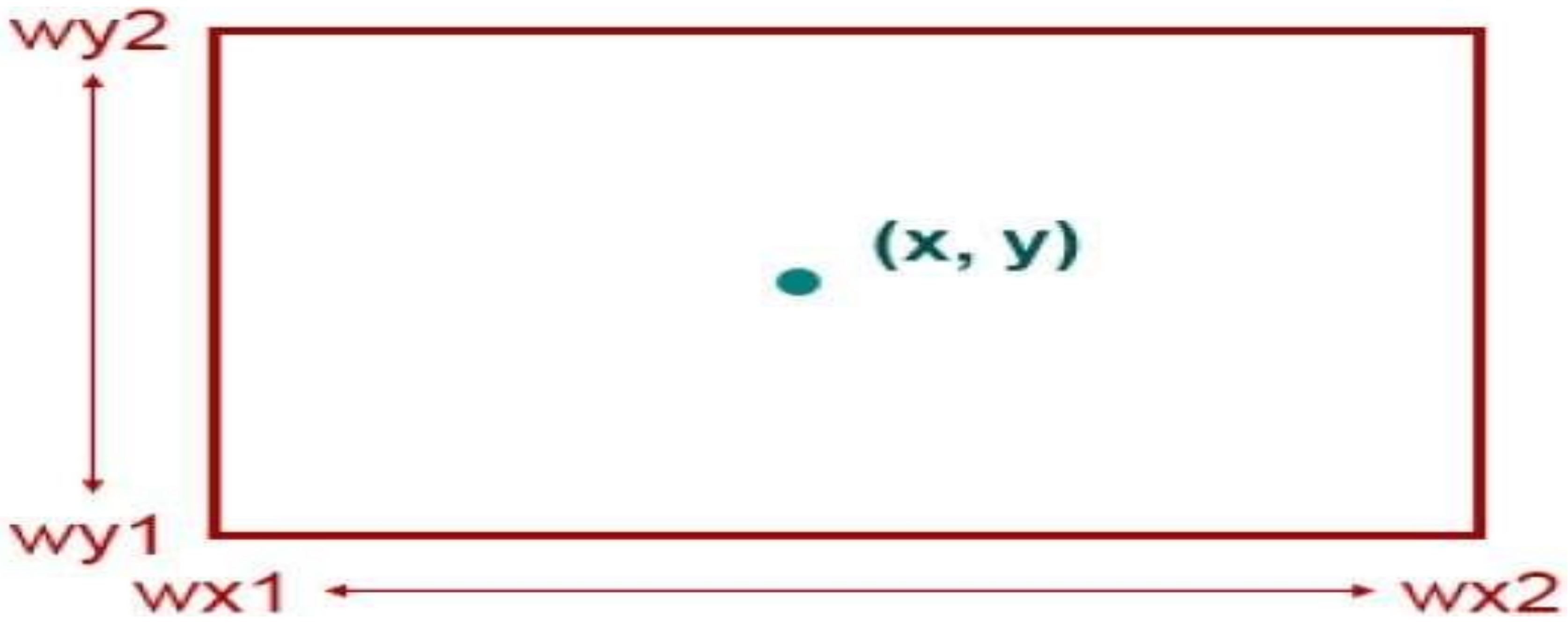
For Y Coordinate

$$\frac{Y_w - Y_{wmin}}{Y_{wmax} - Y_{wmin}} = \frac{Y_v - Y_{vmin}}{Y_{vmax} - Y_{vmin}}$$

Point Clipping

Clipping a point from a given window is very easy. Consider the following figure, where the rectangle indicates the window. Point clipping tells us whether the given point X,YX,Y is within the given window or not; and decides whether we will use the minimum and maximum coordinates of the window.

The X-coordinate of the given point is inside the window, if X lies in between $Wx1 \leq X \leq Wx2$. Same way, Y coordinate of the given point is inside the window, if Y lies in between $Wy1 \leq Y \leq Wy2$.


wy2
wy1
(x, y)
wx1
wx2

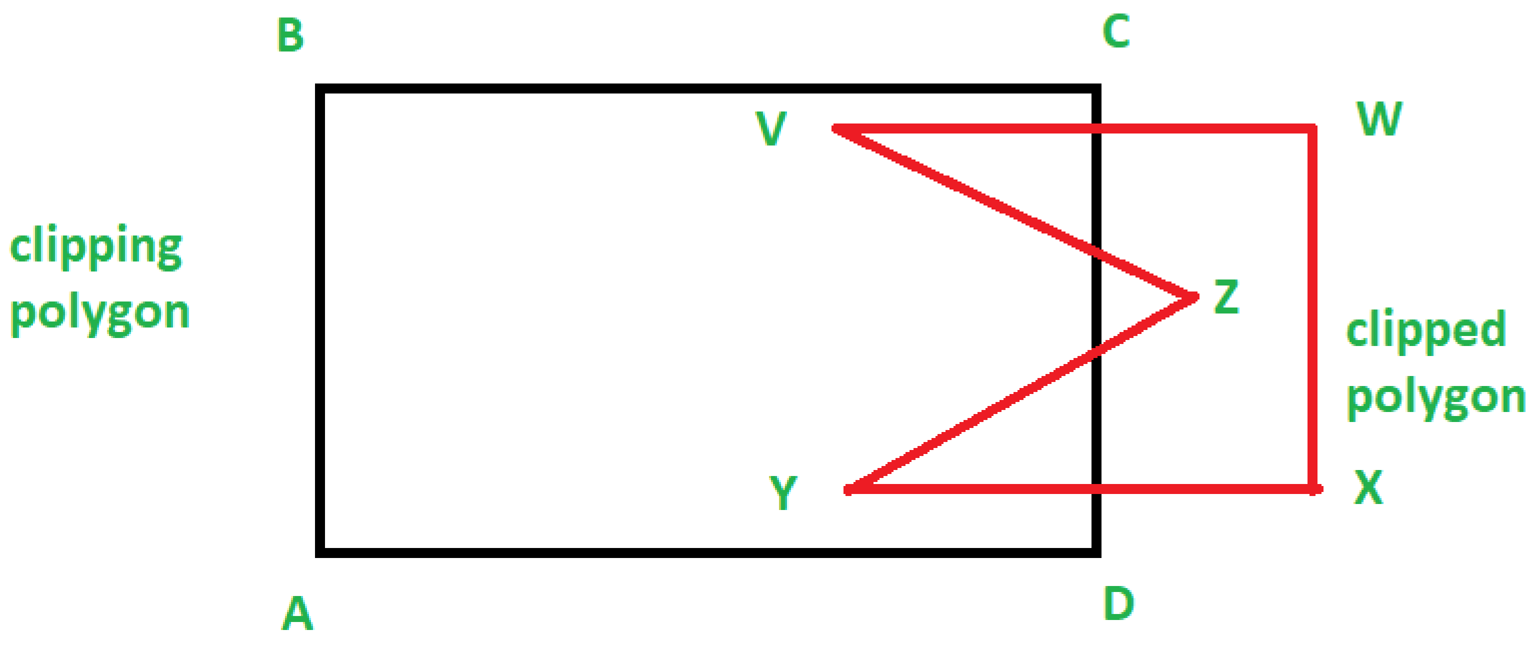
Weiler Atherton – Polygon Clipping Algorithm

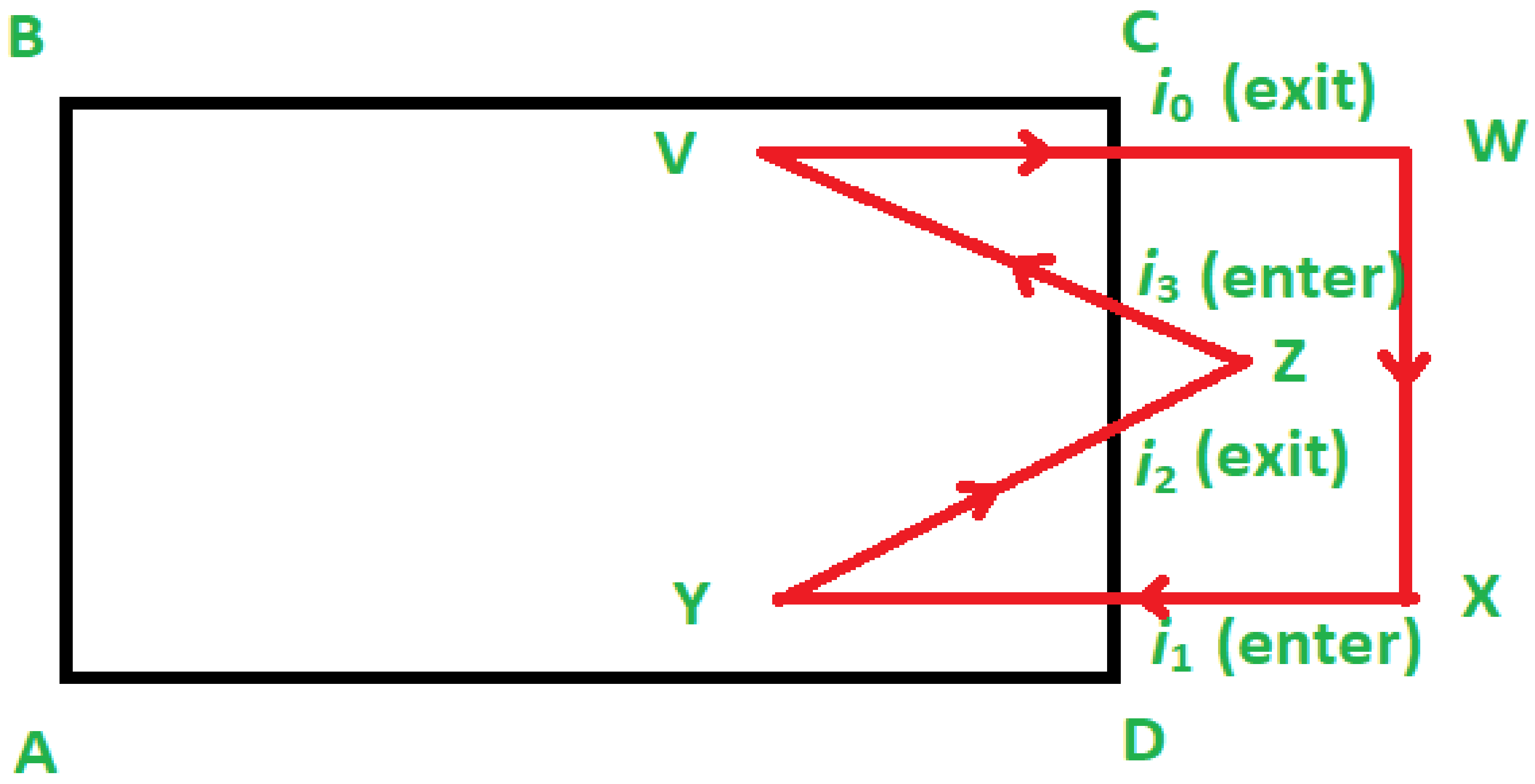
- When the clipped polygons have two or more separate sections, then it is the concave polygon handled by this algorithm.

- The vertex-processing procedures for window boundaries are modified so that concave polygon is displayed.

1. Finding all the intersection points and grouping them

Here, let there be a polygon ABCD and another polygon VWXYZ. Let ABCD be the clipping polygon and let VWXYZ be the clipped polygon.

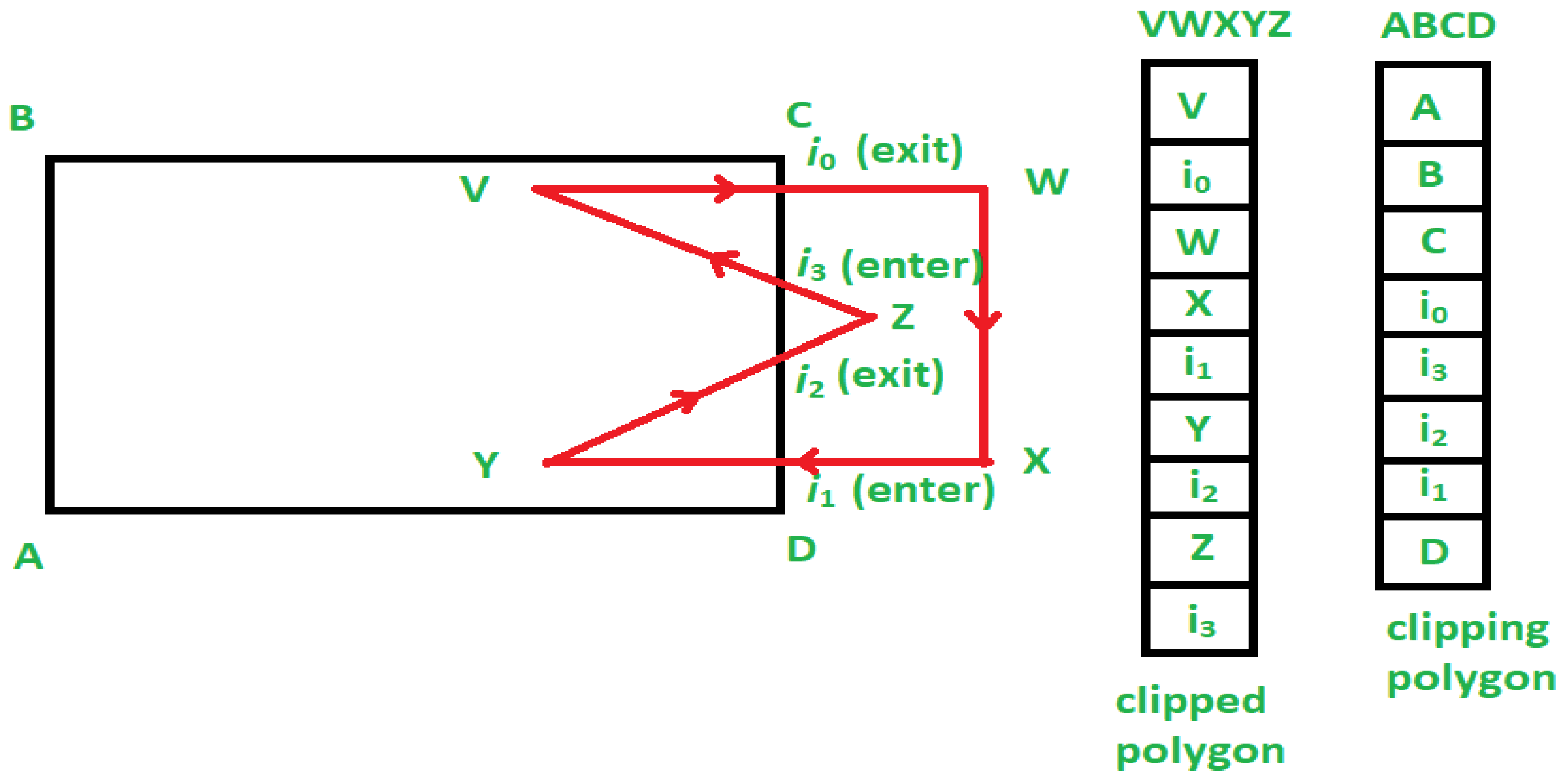



B
C
i_0 (exit)
W
V
i_3 (enter)
Z
i_2 (exit)
Y
X
i_1 (enter)
A
D

2. Making and filling of two lists

Now, we make two lists. One for the clipping polygon and one for the clipped polygon.

Now this is how we fill it:

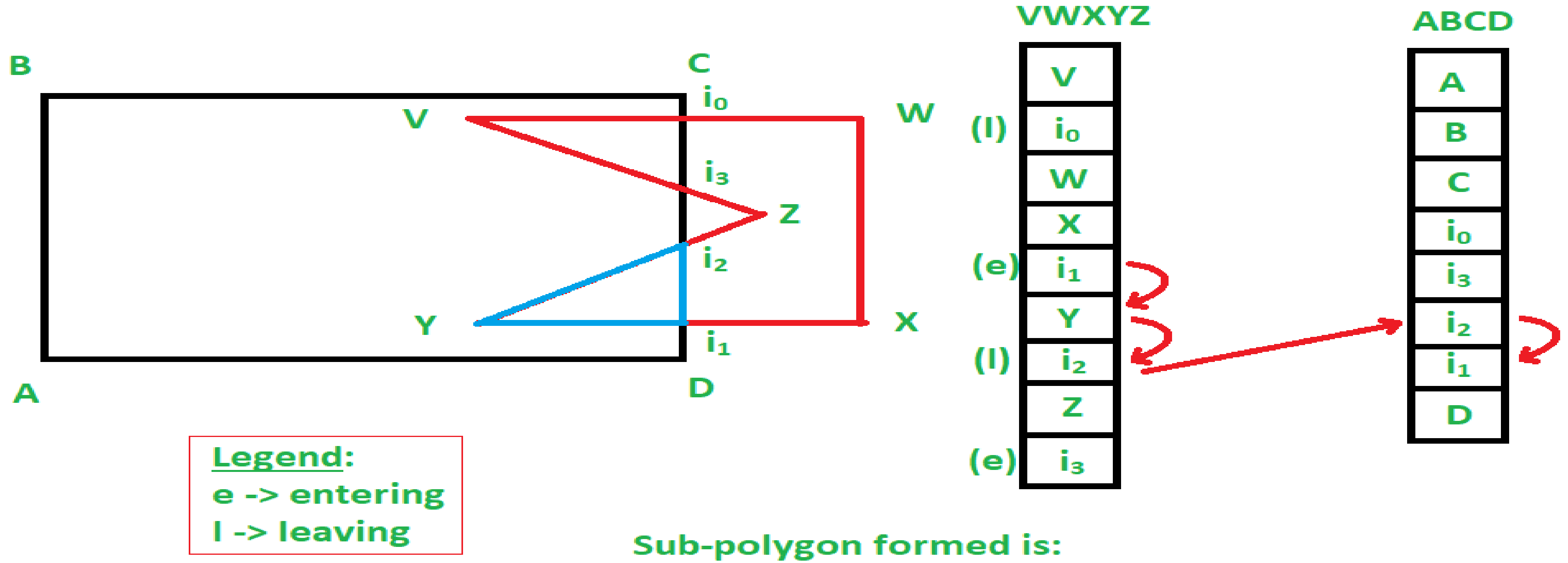


3. Running of the algorithm

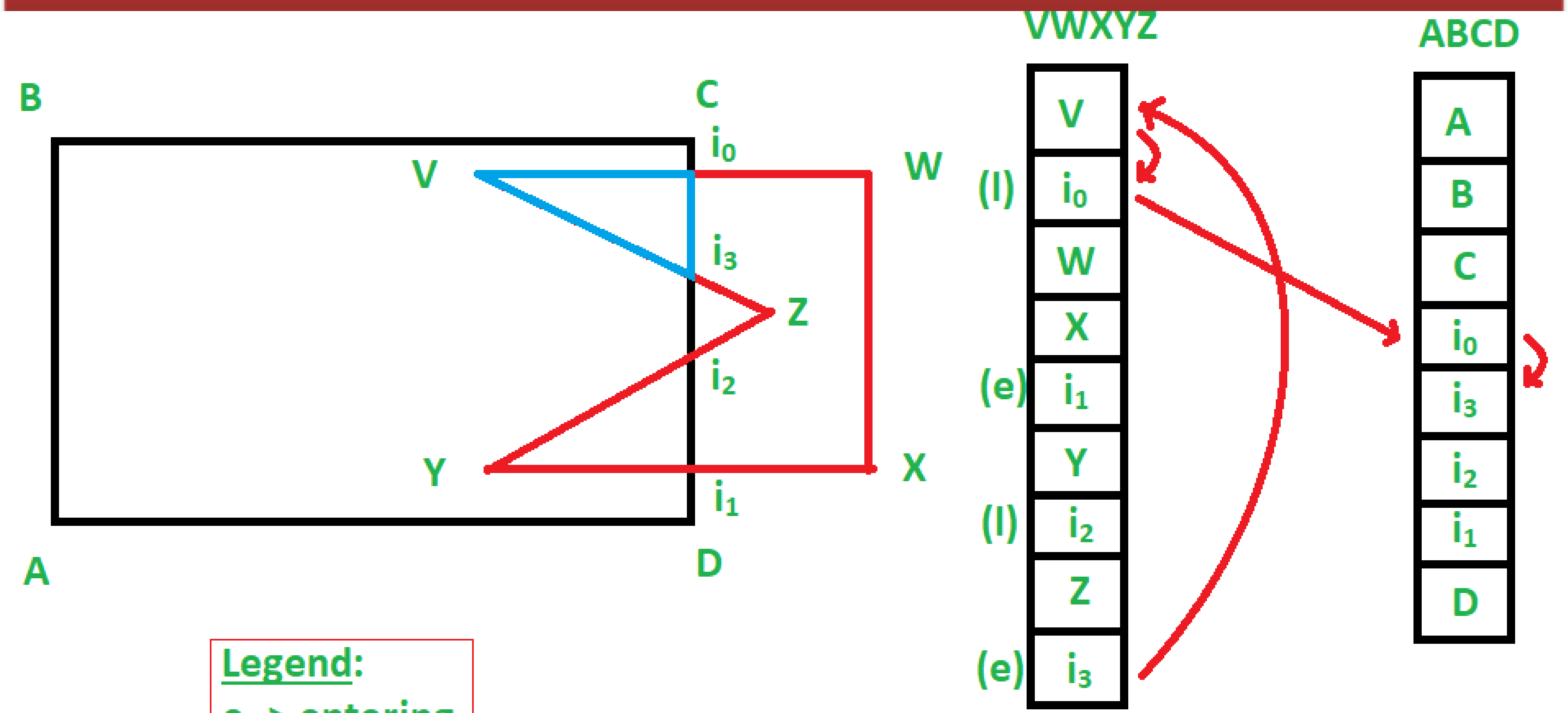
We start at the clipped polygon's list, i.e. VWXYZ.

Now, we find the first intersecting point that is entering. Hence we choose i_1.

From here we begin the making of the list of vertices (or vector) to make a clipped sub-polygon.

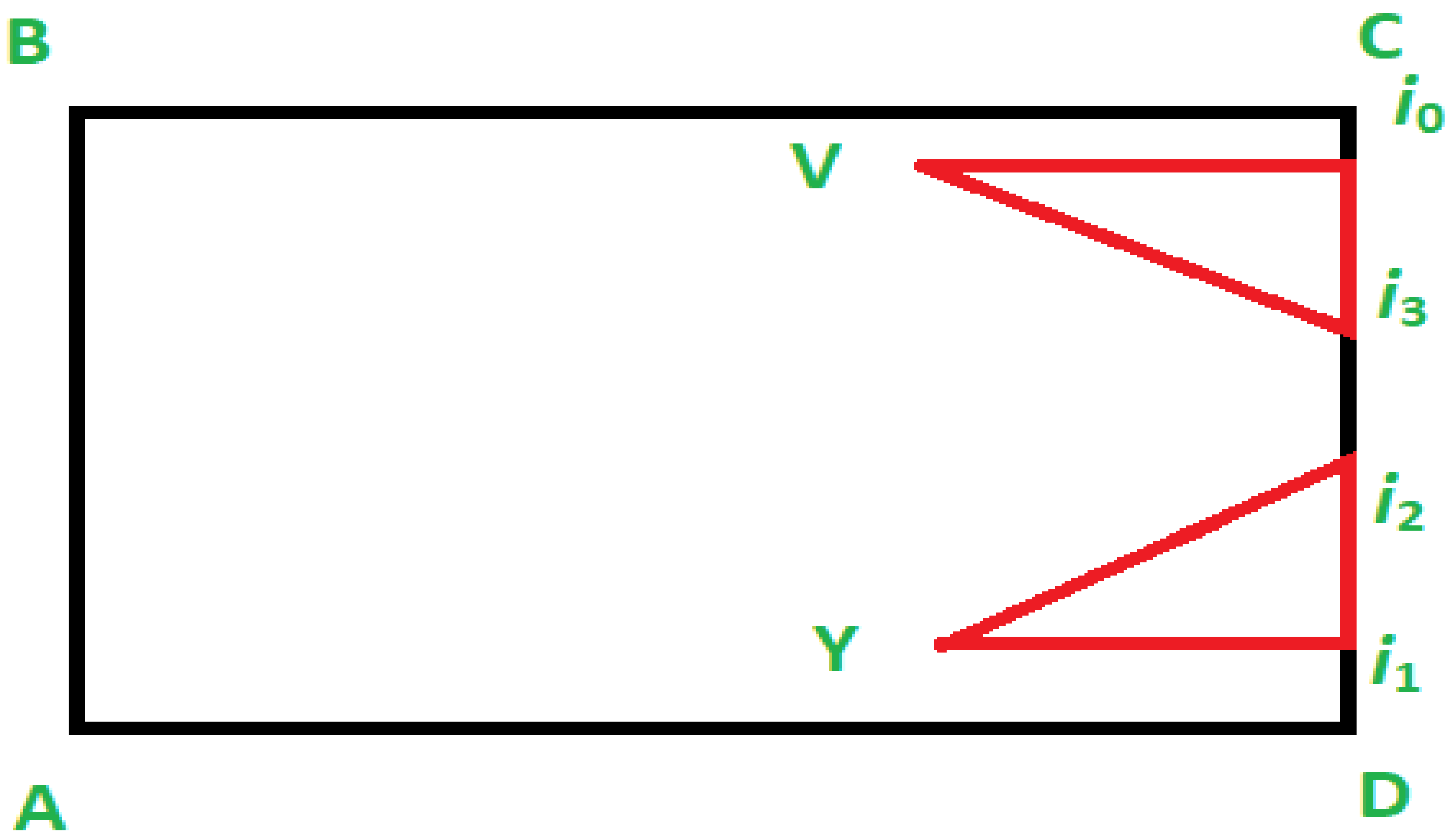


Sub-polygon formed is: i_1 Y i_2 which is a triangle and highlighted as blue in the figure



Legend:
e -> entering
l -> leaving

The resulting subpolygon here is i_3 V i_0


B
C
i_0
V
i_3
i_2
Y
i_1
A
D

Thank You