Exp.No: 04 Vishalini.R

Date: 25-09-25 231701062

**Polygon Clipping**

**Aim:**

To write a program that clips a polygon to a specified rectangular clipping window using the **Sutherland–Hodgman Polygon Clipping Algorithm** and display the clipped polygon.

**Procedure:**

1. Start the program.
2. Input the coordinates of the polygon and the clipping window.
3. For each edge of the clipping window:
   * Take all vertices of the polygon as input list.
   * For every adjacent pair of vertices:
     + If the end vertex is inside the clipping edge:
       - If the start vertex is also inside, add end vertex to output list.
       - If the start vertex is outside, find intersection point with clipping edge and add intersection + end vertex to output list.
     + If the end vertex is outside but start is inside, find intersection point and add intersection point only.
   * Replace input list with output list for next edge.
4. After processing all edges, the output list contains the final clipped polygon vertices.
5. Display:
   * Draw original polygon.
   * Draw clipping window.
   * Draw clipped polygon in a different color.
6. Stop.

**Code:**

import matplotlib.pyplot as plt

def clip\_polygon(subject\_polygon, clip\_window):

def inside(p, edge):

x, y = p

x1, y1, x2, y2 = edge

return (x2 - x1) \* (y - y1) - (y2 - y1) \* (x - x1) >= 0

def compute\_intersection(p1, p2, edge):

x1, y1, x2, y2 = edge

px1, py1 = p1

px2, py2 = p2

A1 = py2 - py1

B1 = px1 - px2

C1 = A1 \* px1 + B1 \* py1

A2 = y2 - y1

B2 = x1 - x2

C2 = A2 \* x1 + B2 \* y1

det = A1 \* B2 - A2 \* B1

if det == 0:

return None

x = (B2 \* C1 - B1 \* C2) / det

y = (A1 \* C2 - A2 \* C1) / det

return x, y

output\_list = subject\_polygon

for i in range(len(clip\_window)):

input\_list = output\_list

output\_list = []

A = clip\_window[i]

B = clip\_window[(i + 1) % len(clip\_window)]

for j in range(len(input\_list)):

P = input\_list[j]

Q = input\_list[(j + 1) % len(input\_list)]

if inside(Q, (A[0], A[1], B[0], B[1])):

if not inside(P, (A[0], A[1], B[0], B[1])):

intersection = compute\_intersection(P, Q, (A[0], A[1], B[0], B[1]))

if intersection:

output\_list.append(intersection)

output\_list.append(Q)

elif inside(P, (A[0], A[1], B[0], B[1])):

intersection = compute\_intersection(P, Q, (A[0], A[1], B[0], B[1]))

if intersection:

output\_list.append(intersection)

return output\_list

# Example Input

subject\_polygon = [(50, 150), (200, 50), (350, 150), (350, 300), (250, 300)]

clip\_window = [(100, 100), (300, 100), (300, 250), (100, 250)]

# Clipping

clipped\_polygon = clip\_polygon(subject\_polygon, clip\_window)

# Display

plt.figure()

subject\_x, subject\_y = zip(\*subject\_polygon)

plt.fill(subject\_x, subject\_y, 'skyblue', alpha=0.3, label='Original Polygon')

clip\_x, clip\_y = zip(\*clip\_window)

plt.plot(list(clip\_x) + [clip\_x[0]], list(clip\_y) + [clip\_y[0]], 'k--', label='Clipping Window')

if clipped\_polygon:

clipped\_x, clipped\_y = zip(\*clipped\_polygon)

plt.fill(clipped\_x, clipped\_y, 'orange', alpha=0.6, label='Clipped Polygon')

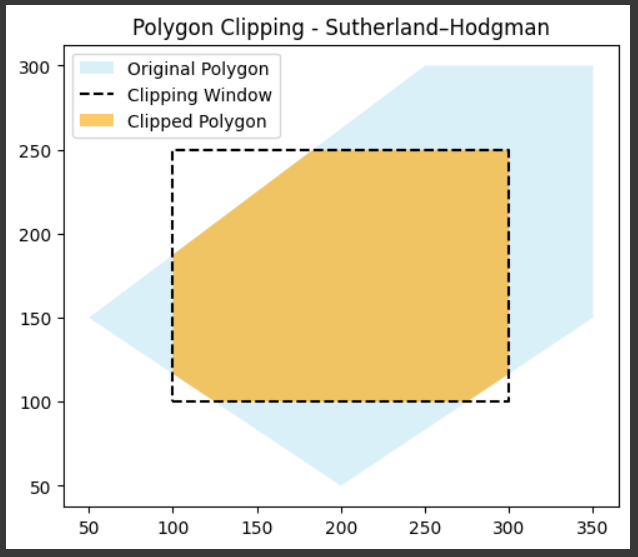
plt.gca().set\_aspect('equal')

plt.legend()

plt.title("Polygon Clipping - Sutherland–Hodgman")

plt.show()

**Output:**

****

**Result:**

The program successfully clips the given polygon to the specified rectangular window.  
 The clipped polygon is displayed graphically along with the original polygon and clipping window.