Salahaddin University-Erbil/College of Science
Department of Computer Science & IT

Computer Graphics

Lecture 14

Tarza Hasan

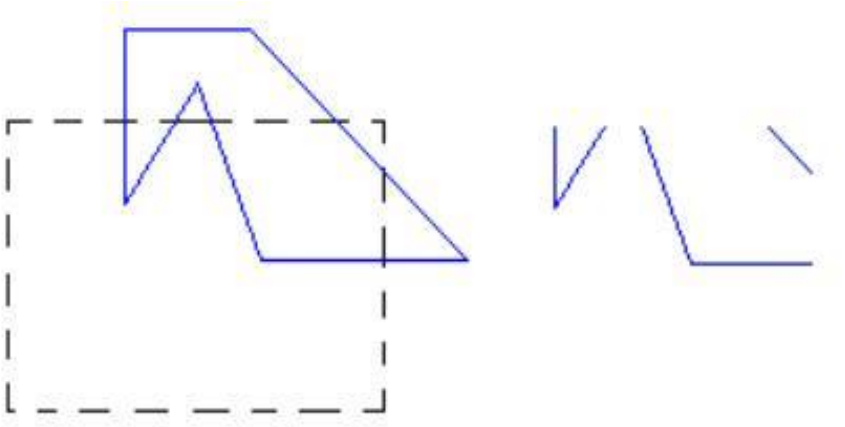
tarza.abdullah@su.edu.krd

Contents

- Clipping
  - **Polygon Clipping**
  - **Sutherland-Hodgeman Polygon Clipping Algorithm**

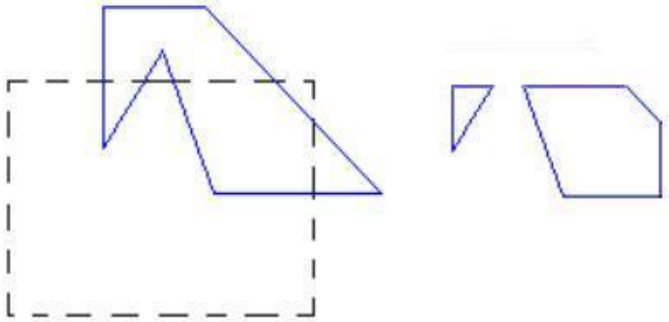
Polygon Clipping

To clip a **polygon**, we cannot directly apply a line-clipping method to the individual polygon edges because this approach would produce a series of **unconnected line segments** as shown in figure .



Polygon Clipping (Cont…)

The clipped polygons must be a bounded area after clipping as shown in figure.



- For polygon clipping, we require an algorithm that will generate one or more **closed areas** that are then scan converted for the appreciate area fill.
- The output of a polygon clipper should be a **sequence of vertices** that defines the clipped polygon boundaries.

Sutherland-Hodgman Polygon Clipping

Clip a polygon by processing the polygon boundary as a whole against each window edge.

- Processing all polygon vertices against each clip rectangle boundary in turn.

- Beginning with the initial set of polygon vertices, we could first clip the polygon against the **left** rectangle boundary to **produce a new sequence of vertices.**

- The new set of vertices could be successively passed to a **right** boundary clipper, a **bottom** boundary clipper, and a **top** boundary clipper, a right boundary clipper.

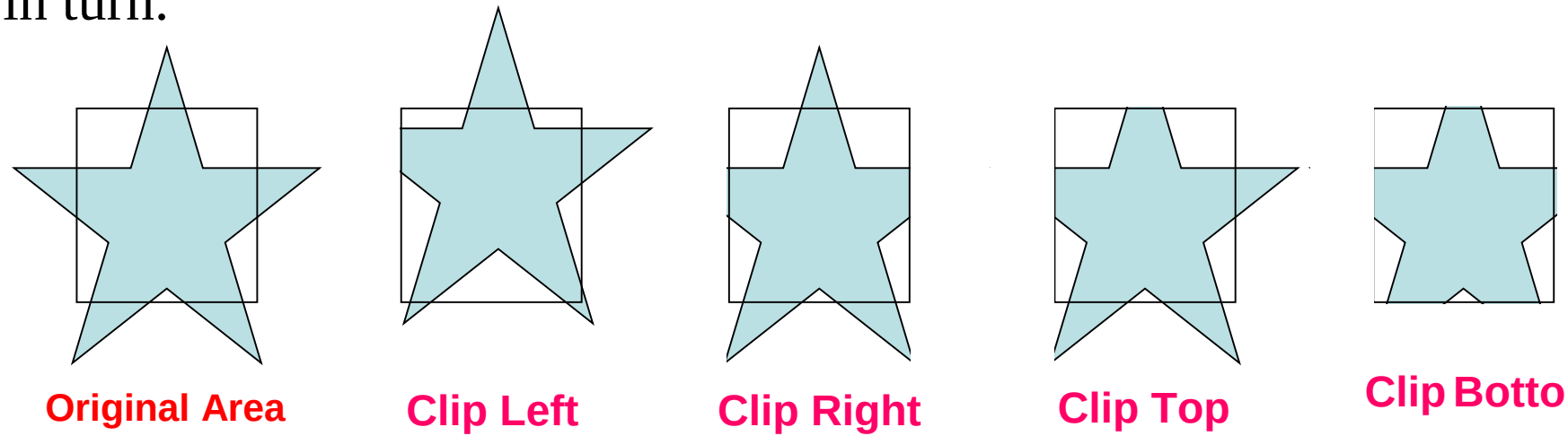
Sutherland-Hodgman Polygon Clipping (cont…)

A technique for clipping areas developed by **Sutherland & Hodgman.**

Basic Concept: Simplify via separation, clip the whole polygon against one edge

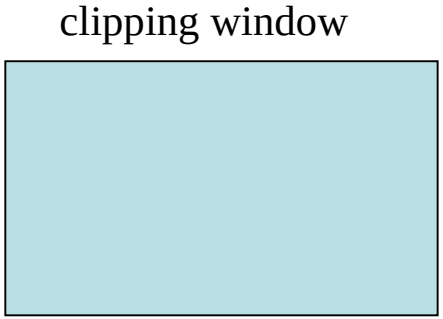
Repeat with output for other 3 edges

Put simply the **polygon** is clipped by comparing it **against** each boundary in turn.



Sutherland-Hodgman Polygon Clipping (cont…)

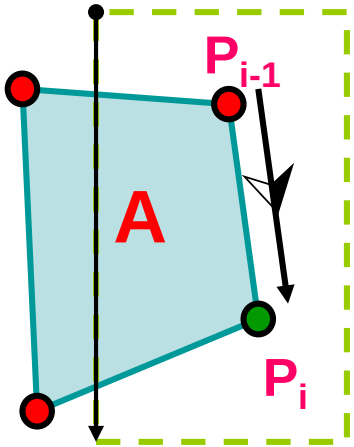
Let (P_1, P_2,…. P_N) be the **vertex list** of the Polygon to be clipped and E be the edge *clipping window*.



We **clip** each **edge** of the polygon in turn against each window **edge E**, forming **a new polygon** whose vertices are determined as follows:

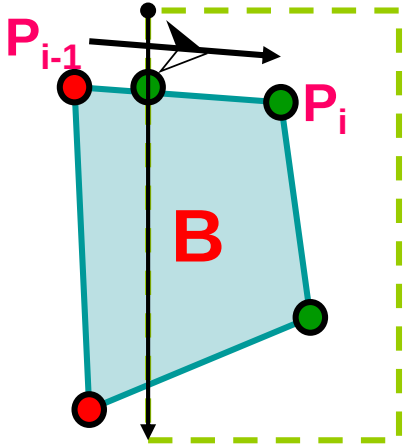
Sutherland-Hodgman Polygon Clipping (cont…)

Four cases:

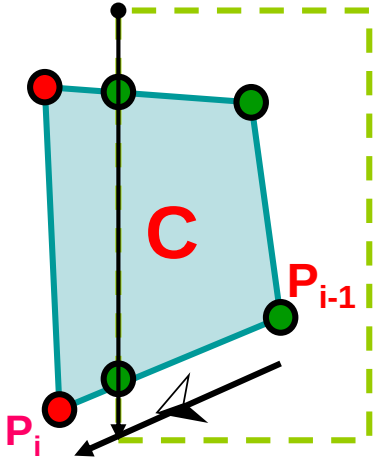


In → In

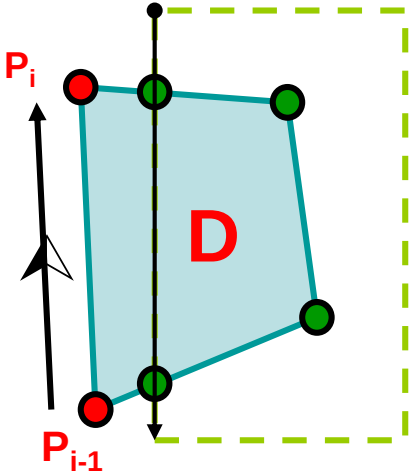
save ending vert
Inside
(1 output)



Out → In
save new clip vert
and ending vert
Entering
(2 outputs)



In → Out
save new clip vert
Leaving
(1 output)



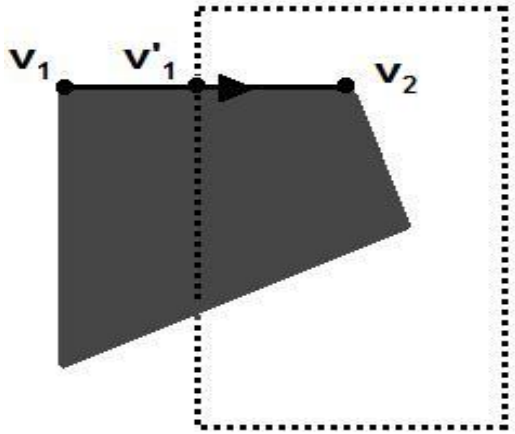
Out → Out
save nothing
Outside
(0 output)

Sutherland-Hodgman Polygon Clipping (cont…)

There are **four** possible **cases** when processing vertices in sequence around the polygon.

As **each pair** of **adjacent polygon vertices** is passed to a next window boundary clipper, we make the following tests:

1. **If** the **first vertex** is **outside** the window boundary and the **second vertex** is **inside**

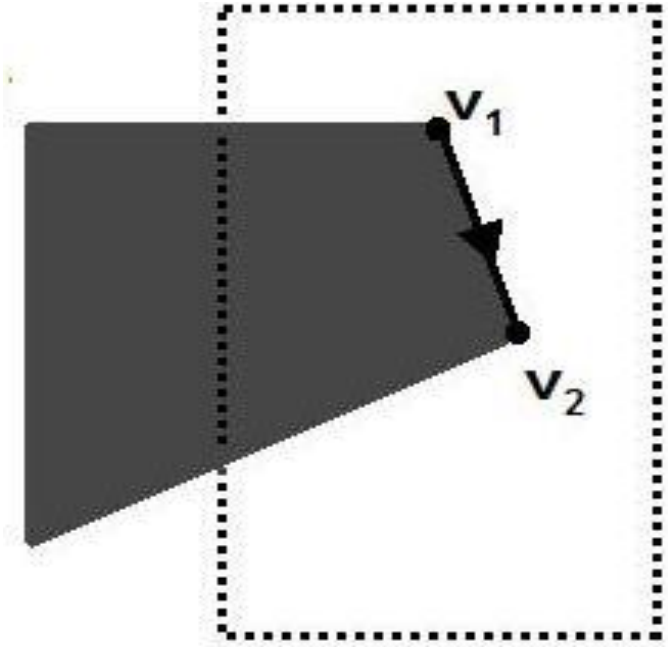


Then , both the **intersection point** of the polygon edge with the window boundary and the **second vertex** are added to the output vertex list.

Sutherland-Hodgman Polygon Clipping (cont…)

2. If both input vertices are **inside** the window boundary.

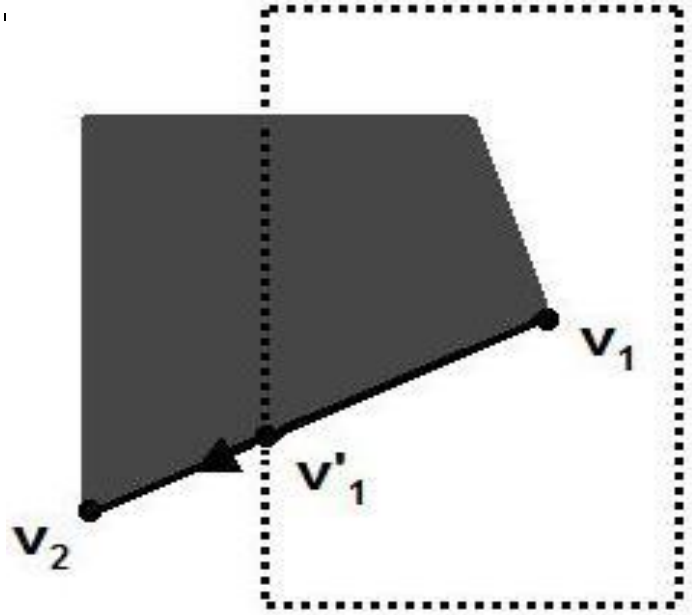
Then, only the **second vertex** is added to the output vertex list.



Sutherland-Hodgman Polygon Clipping (cont…)

3. **If** the **first vertex** is **inside** the window boundary and the **second vertex** is **outside**.

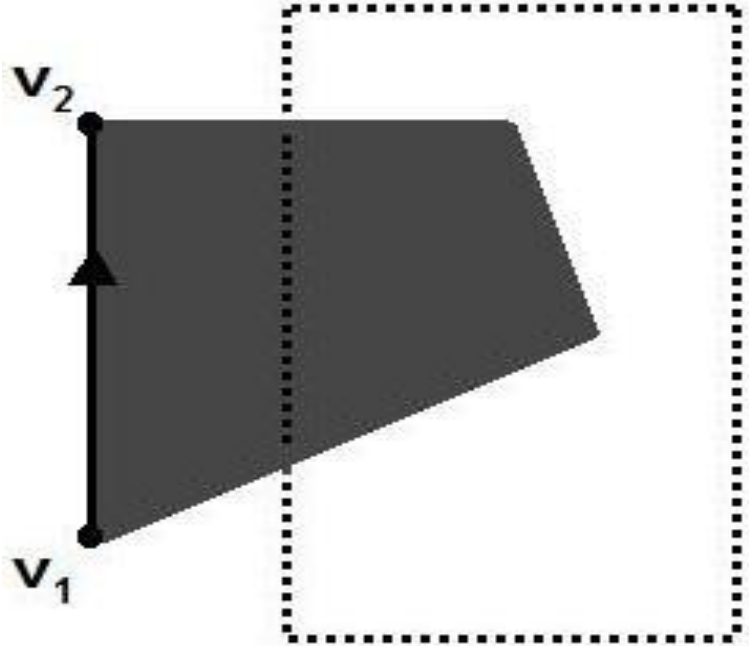
Then, only the edge intersection with the window boundary is added to the output vertex list.



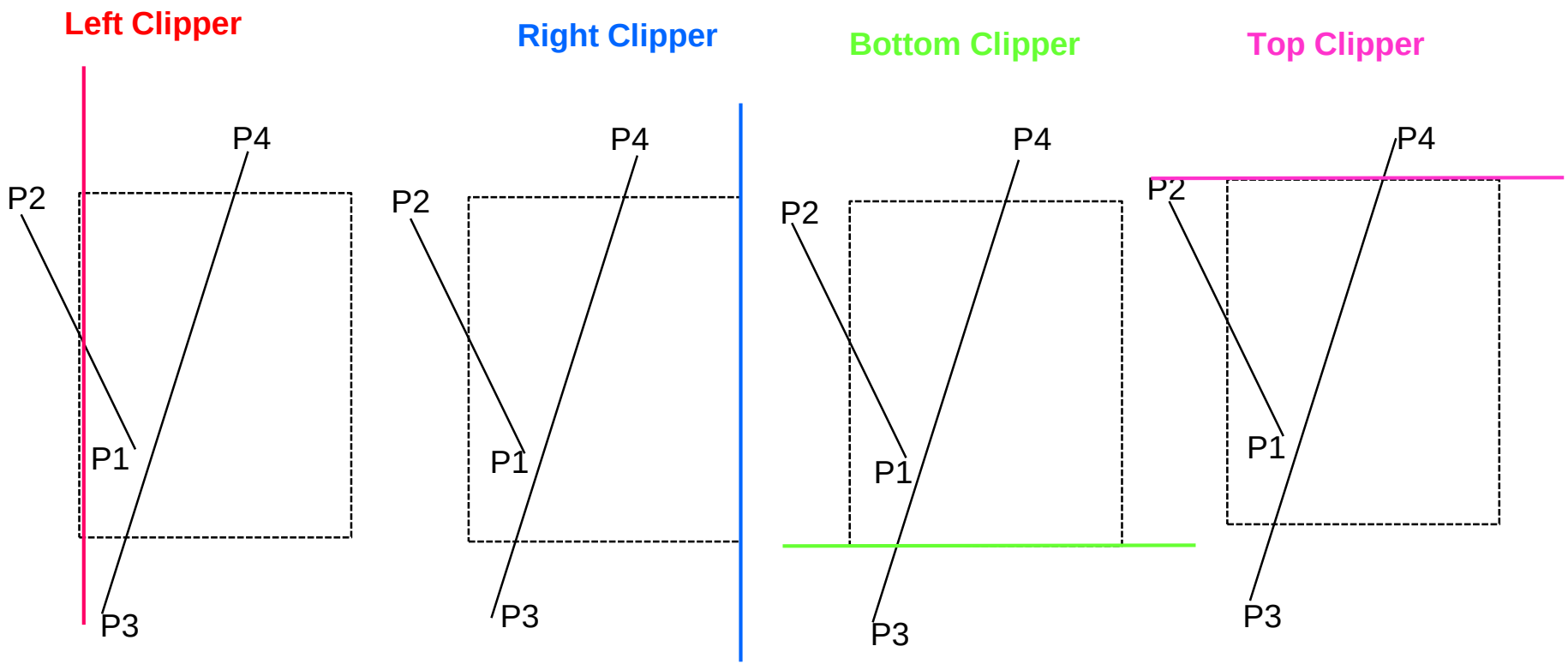
Sutherland-Hodgman Polygon Clipping (cont…)

4. If both input vertices are **outside** the window boundary.

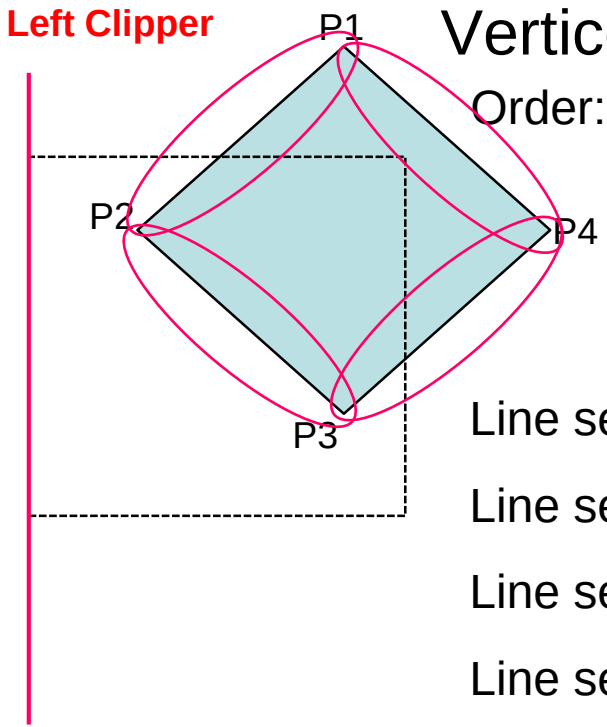
Then, nothing is added to the output vertex list.



Clippers and clipping stages



Sutherland-Hodgman Polygon Clipping (Example 1)



Vertices list is passed stage bye stage to clippers

Order: Left, Right, Bottom, Top

Line segment (**p1,p2**), both inside➔ save p2.

Line segment (**p2,p3**), both inside➔ save p3.

Line segment (**p3,p4**), both inside ➔save p4.

Line segment (**p4,p1**), both inside ➔save p1.

Initial vertices:

P1

P2

P3

P4

Left clipper vertices:

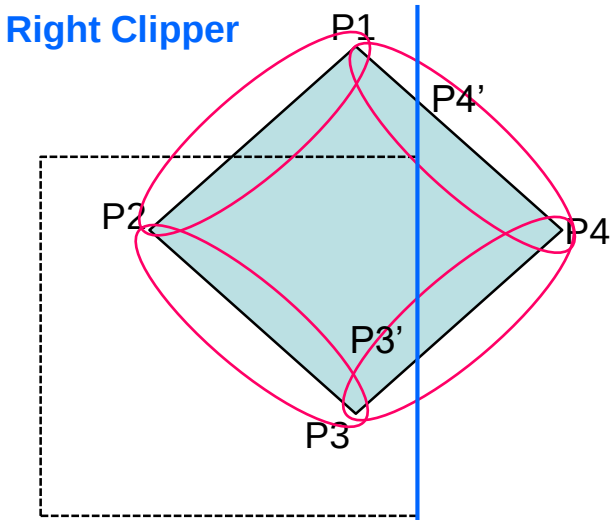
P2

P3

P4

P1

Sutherland-Hodgman Polygon Clipping (Example 1)



Order: Left, Right, Bottom, Top

Left clipper vertices:

P2

P3

P4

P1

Right clipper vertices:

P3

P3'

P4'

P1

P2

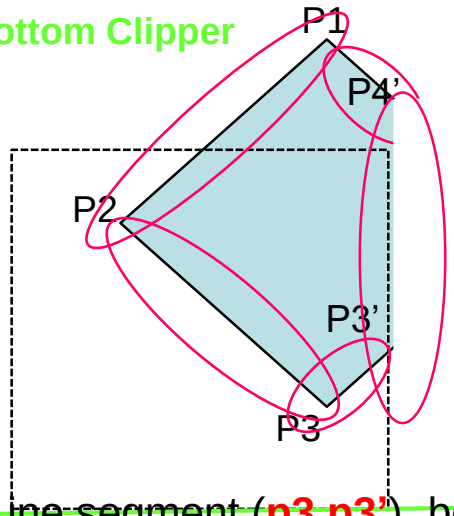
Line segment (**p2,p3**), both inside➔ save p3.

Line segment (**p3,p4**), inside, outside➔ save p3'.

Line segment (**p4,p1**), outside, inside ➔save p4', save p1 .

Line segment (**p1,p2**), both inside ➔save p2.

Sutherland-Hodgman Polygon Clipping (Example 1)



Order: Left, Right, Bottom, Top

Right clipper vertices:

P3

P3’

P4’

P1

P2

Bottom clipper vertices:

P3

P3’

P4’

P1

P2

Line segment (**p3,p3’**), both inside➔ save p3’.

Line segment (**p2,p3**), both inside➔ save p3.

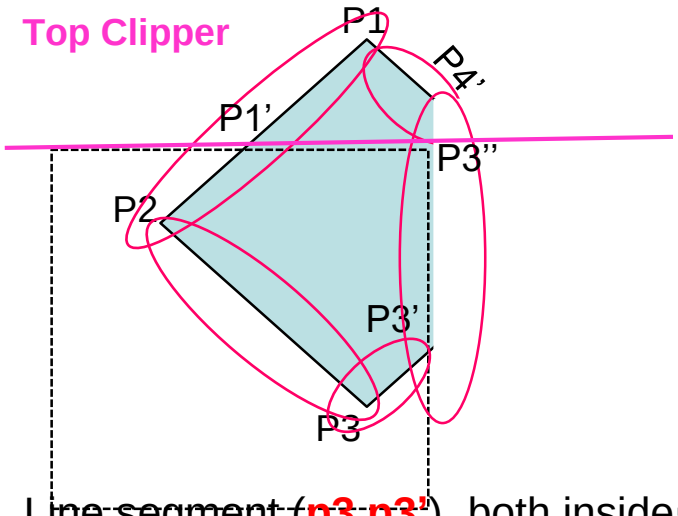
Line segment (**p3’,p4’**), both inside ➔save p4’.

Line segment (**p4’,p1**), both inside ➔save p1.

Line segment (**p1,p2**), both inside ➔save p2.

Sutherland-Hodgman Polygon Clipping (Example 1)

Top Clipper



Order: Left, Right, Bottom, Top

Line segment (~~**p3,p3''**~~), both inside➔ save p3'.

Line segment (**p3',p4'**), inside,outside➔ save p3''.

Line segment (**p1,p4'**), both outside ➔no vertex will be saved.

Line segment (**p1,p2**), outside,inside ➔save p1',save p2.

Line segment (**p2,p3**), inside, inside ➔save p3.

Bottom clipper vertices:

P3

P3'

P4'

P1

P2

Top clipper vertices:

P3

P3''

P4'

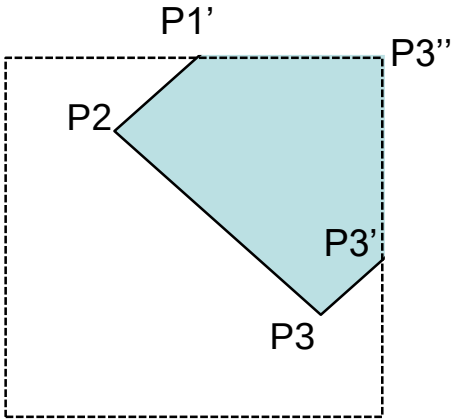
P1

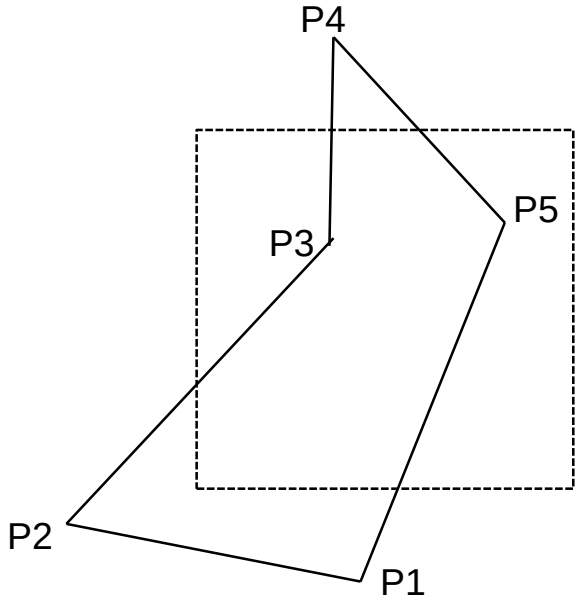
P2

p3

Sutherland-Hodgman Polygon Clipping (Example)

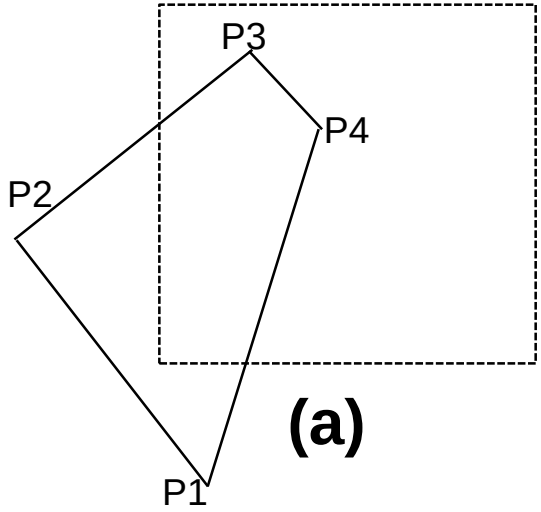
Polygon after clipping



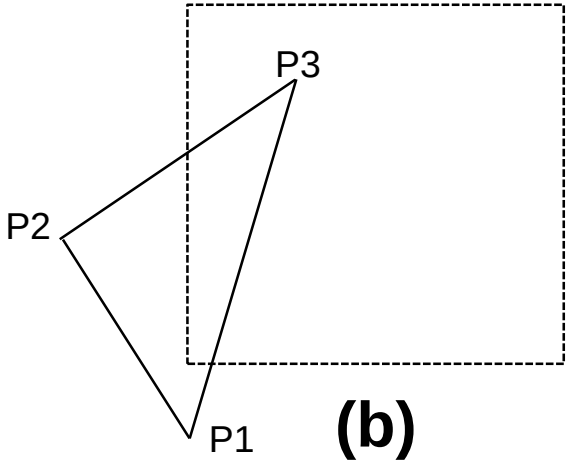


Activity

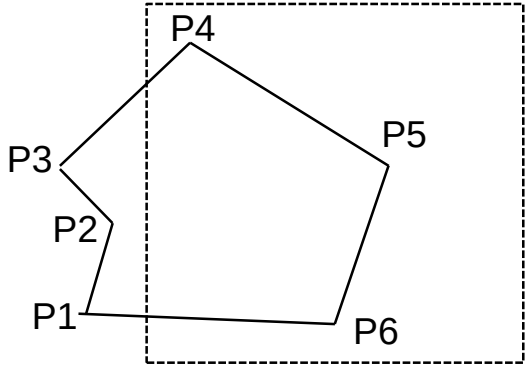
Clip the following figures with Sutherland-Hodgman polygon clipping algorithm



(a)



(b)



(c)