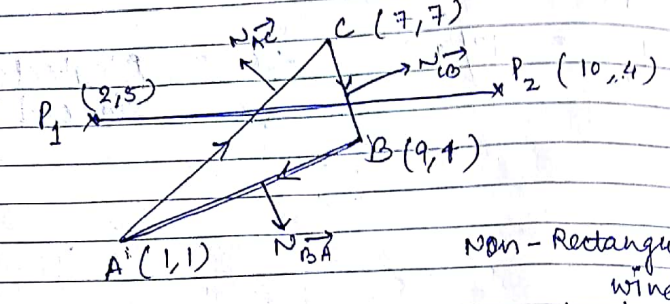
28/02/2018

Line Clipping

Cyrus – Beck :

Example 3 :



This window is convex as all the angles $< 180°$

Non-Rectangular window → Can be convex or concave

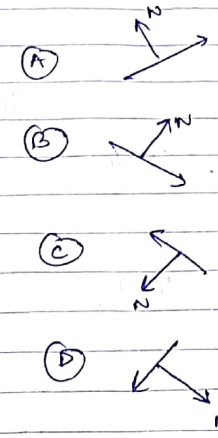
We can traverse the window either in ↷ or ↶ dir^n.

When ↷ → RHS is always towards the window, LH → outward

Edges	Normal (N_i)	P_{Ei}	$P_1 - P_{Ei}$	$N_i \cdot (P_1 - P_{Ei})$	$(P_2 - P_1) \cdot N_i$	t
$\vec{AC}$ $(6\hat{i}+6\hat{j})$	$-6\hat{i}+6\hat{j}$	$i+\hat{j}$	$\hat{i}+4\hat{j}$	18	−54	$1/3$ (PE_{ent})
$\vec{CB}$ $(2\hat{i}-3\hat{j})$	$3\hat{i}+2\hat{j}$	$7\hat{i}+7\hat{j}$	$-5\hat{i}-2\hat{j}$	−19	22	$19/22$ (PE_{ext}) (P_{Exit})
$\vec{BA}$ $(-8\hat{i}-3\hat{j})$	$3\hat{i}-8\hat{j}$	$9\hat{i}+4\hat{j}$	$-7\hat{i}+\hat{j}$	−29	32	$29/32$ (PE_{ent}) (P_{Exit})

$P_2 - P_1 = 8\hat{i} - \hat{j}$

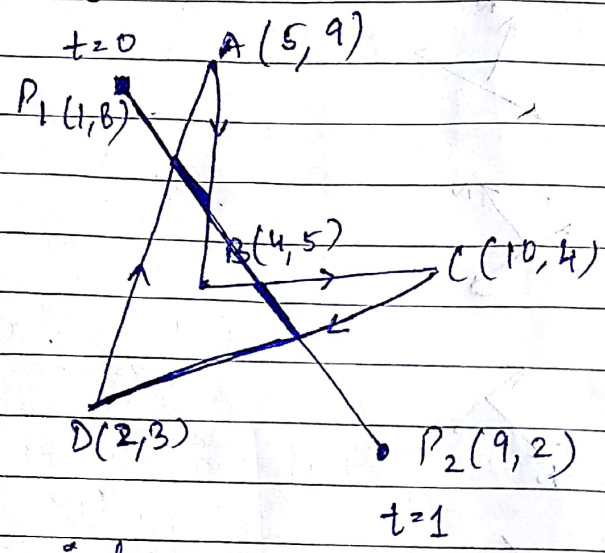
Edge	Normal
(A) + +	− +
(B) + −	+ +
(C) − +	− −
(D) − −	+ −



25/01/2018

Line Clipping

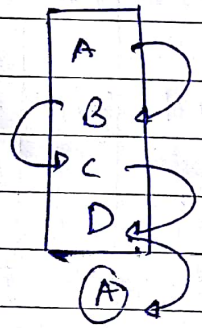
Non-Rectangular Window:



Clockwise

Concave window

arr[]



$P_1 = \hat{i} + 8\hat{j}$

$P_2 = 9\hat{i} + 2\hat{j}$

$P_2 - P_1 = 8\hat{i} - 6\hat{j}$

Edges	Normal (N_i)	P_{E_i}	$P_1 - P_{E_i}$	$N_i \cdot (P_1 - P_{E_i})$	$N_i \cdot (P_2 - P_1)$	t
$\overrightarrow{AB} = -\hat{i} - 4\hat{j}$	$4\hat{i} - \hat{j}$	$5\hat{i} + 9\hat{j}$	$-4\hat{i} - \hat{j}$	-15	38 (P_{Exit})	15/38
$\overrightarrow{BC} = 6\hat{i} - \hat{j}$	$\hat{i} + 6\hat{j}$	$4\hat{i} + 5\hat{j}$	$-3\hat{i} + 3\hat{j}$	15	-28 (P_{Ent})	15/28
$\overrightarrow{CD} = -8\hat{i} - \hat{j}$	$\hat{i} - 8\hat{j}$	$10\hat{i} + 4\hat{j}$	$-9\hat{i} + 4\hat{j}$	-41	56 (P_{Exit})	41/56
$\overrightarrow{DA} = 3\hat{i} + 6\hat{j}$	$-6\hat{i} + 3\hat{j}$	$2\hat{i} + 3\hat{j}$	$-\hat{i} + 5\hat{j}$	21	-.66 (P_{Ent})	21/66

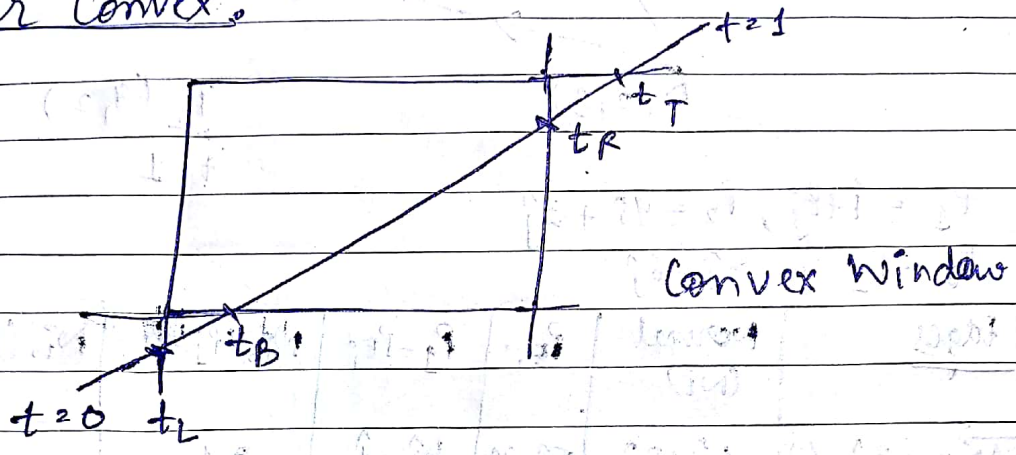
⟶ increasing order

$S : (0 \;\; 21/66 \;\; 15/38 \;\; 15/28 \;\; 41/56 \;\; 1)$ (t)

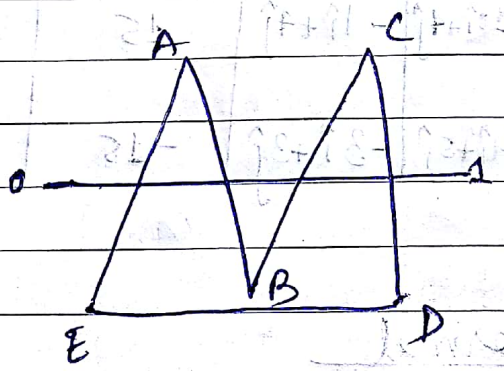
P_1 P_2

P_{Ent} P_{Exit} P_{Ent} P_{Exit}

For Convex:



$S : (0, P_{Ent}, P_{Ent}, P_{Exit}, P_{Exit}, 1)$ (t_L t_B t_R t_T)

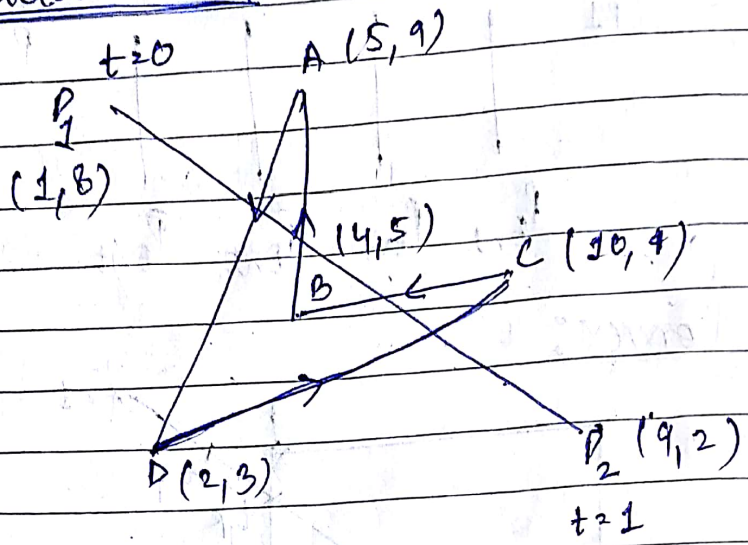


⟹ Ent, Exit, Ent, Exit

If such pairs (Ent, Exit) are more than 1 b/w t=0 & t=1. ⇒ then clipping in concave part.

⟶ ⇒ t<0 or t>1

Anticlockwise:



$P_1 = \hat{i} + 8\hat{j}$, $P_2 = 9\hat{i} + 2\hat{j}$

$P_2 - P_1 = 8\hat{i} - 6\hat{j}$

Edges	Normal (N_i)	P_{E_i}	$P_1 - P_{E_i}$	$N_i \cdot (P_1 - P_{E_i})$	$N_i \cdot (P_2 - P_1)$	t
$\overrightarrow{AD} = -3\hat{i} - 6\hat{j}$	$-6\hat{i} + 3\hat{j}$	$5\hat{i} + 9\hat{j}$	$-4\hat{i} - \hat{j}$	21	-66 (P_{Ent})	21/66
$\overrightarrow{DC} = 8\hat{i} + \hat{j}$	$\hat{i} - 8\hat{j}$	$2\hat{i} + 3\hat{j}$	$-\hat{i} + 5\hat{j}$	-41	56 (P_{Exit})	41/56
$\overrightarrow{CB} = -6\hat{i} + \hat{j}$	$\hat{i} + 6\hat{j}$	$10\hat{i} + 4\hat{j}$	$-9\hat{i} + 4\hat{j}$	15	-28 (P_{Ent})	15/28
$\overrightarrow{BA} = \hat{i} + 4\hat{j}$	$4\hat{i} - \hat{j}$	$4\hat{i} + 5\hat{j}$	$-3\hat{i} + 3\hat{j}$	-15	38 (P_{Exit})	15/38

Edge	Normal
+ +	+ −
+ −	− −
− +	+ +
− −	− +

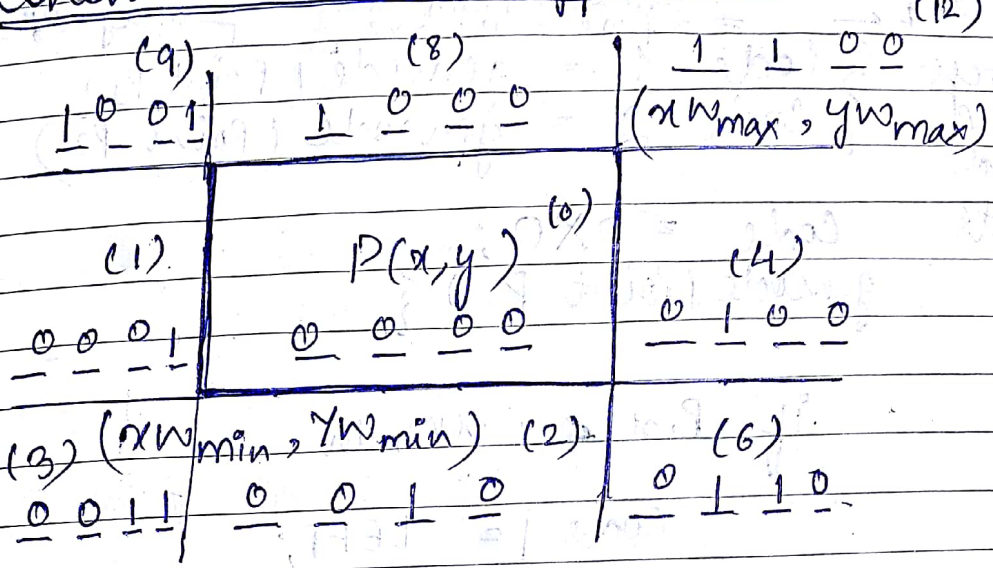
S: (0 21/66 15/38 15/28 41/56 1)

0 P_{Ent} P_{Exit} P_{Ent} P_{Exit}

29/01/2018

Line Clipping

Cohen - Sutherland Approach :



$$xw_{min} \leq x \leq xw_{max}$$
$$yw_{min} \leq x \leq yw_{max}$$

A rectangular windaw as 9 regions.
Point can be any where in these 9 regions.
Each region has a unique region code (address).

Left side → LSB 1.
Bottom side → 2nd bit 1
Right side → 3rd bit 1.
Upper side → MSB 1

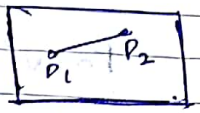
Input: $P_1(x_1, y_1)$, $P_2(x_2, y_2)$

Method:

```
#define LEFT 0X1
#define RIGHT 0X4
#define BOTTOM 0X2
#define TOP 0X8
do{                                          flag = 0; done = 0;
Address { Code 1 = gencode (Point P1)
assigned { Code 2 = gencode (Point P2)
to end
points   Code = 0X0;
         gencode (Point P) {

           if P.x <= xwmin
              Code |= LEFT;

           if P.x >= xwmax
              code |= RIGHT;

           if P.y <= ywmin
              code |= BOTTOM;

           if P.y >= ywmax
              code |= TOP;

         }
```

Case I :

Code (P_1) = 0000
Code (P_2) = 0000



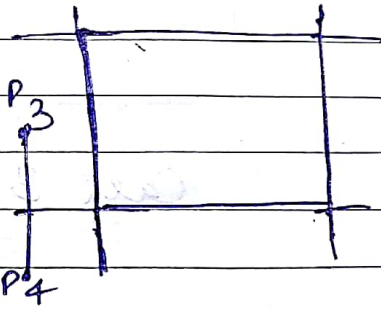
AND
OR

⇓

True ⇒ Line is inside, no clipping

Case II :

Code (P_3) = 0001
Code (P_4) = 0011



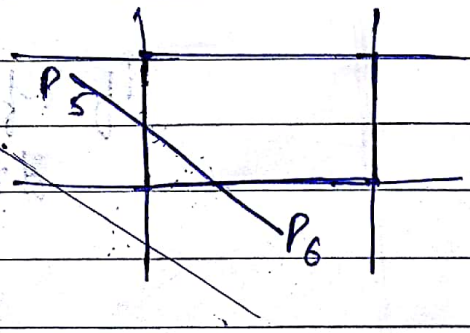
AND

⇓

NON-ZERO ⇒ Line is outside

Case III :

Code (P_5) = 0001
Code (P_6) = 0010



⇓

AND → 0
OR → NON-ZERO

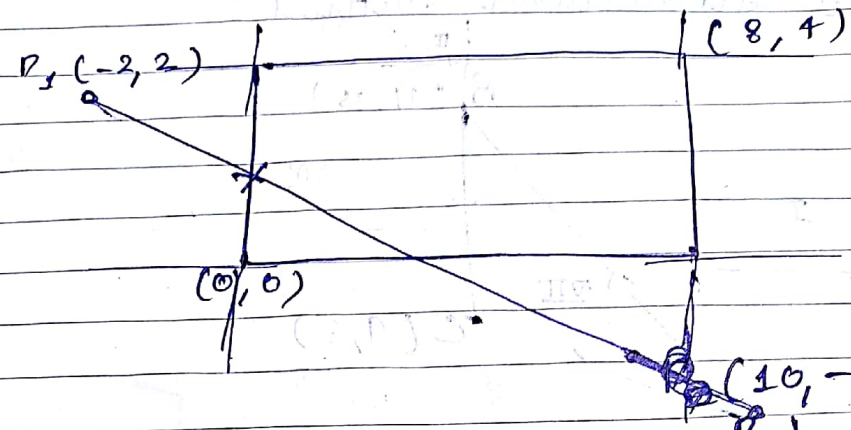
For case III:

* First clip P1 . then
* Keep checking iteratively whether after clipping line falls under case I or case II

Previous code:

```
case 1: if (code1 & code2 == 0) &&
                    (code1 | code2 == 0)
        {
            /* line is visible */
            ~~exit~~ flag = 1; done = 1;
            exit();
        }

case 2: if (code1 & code2 != 0)
        {
            /* line is not visible */
        exit(); done = 1;
        }
Case 3: if (code1 & code2 == 0) && (code1 | code2 != 0)
            find intersection();

} while (done != 1)

if (flag)  ---> // line is inside
    drawline (P1, P2)
```

EXAMPLE 1 :-



code (P_1) = 0001
code (P_2) = 0110

case (3
LEFT = 0X1;

clipped at left edge

if (LEFT & code P_1 == 1)
$x_I = x w_{min}$;
$y_I = y_1 + \frac{(y_2 - y_1)}{(x_2 - x_1)} * (x_I - x w_{min})$

$P_1 \leftarrow (x_I, y_I)$

flag	done
0	0

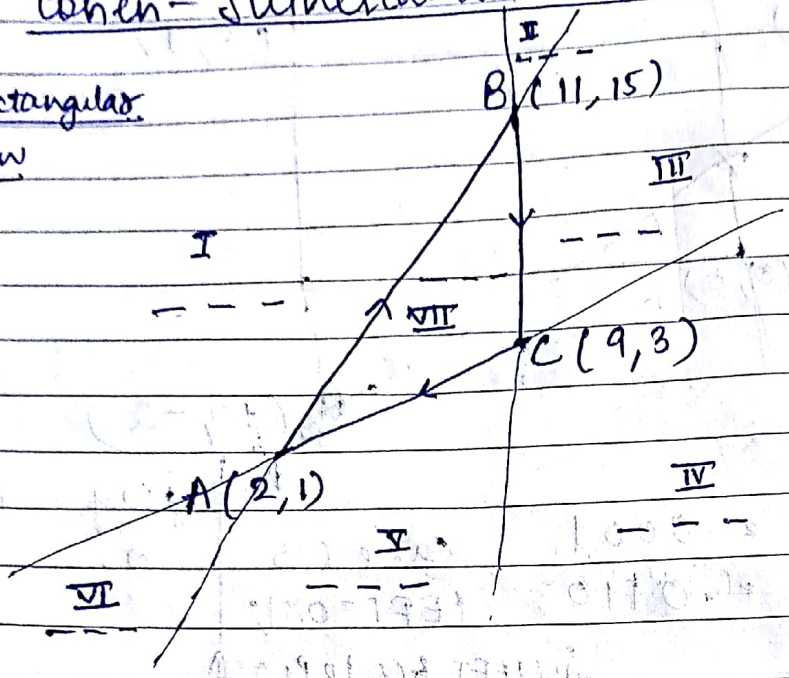
code (P1) = 0000
code (P_2) = 0110

30/01/2018

Line Clipping

Cohen-Sutherland Method:

Non-rectangular window



Inside Test

Equation	
AB $f(x,y) = 14x - 9y - 19$	$f(x,y)$: C(9, 3) > 0
BC $f(x,y) = 6x + y + 5$	$f(x,y)$: A(2, 1) > 0
CA $f(x,y) = 2x + 7y - 3$	$f(x,y)$: B(11, 15) > 0

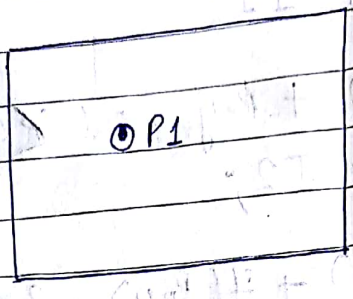
> 0 → Point is inside the window.

```
#define I1 1
#define I2 2
#define I3 4
int code = 0;
int gencode(Point P)
{
    if( 14(P.x) - 9(P.y) -19 <0)
        code |= I1;
    if( -6(P.x) + P.y +51 <0)
        code |= I2;
    if( -2(P.x) + 7(P.y) -3 <0)
        code |= I3;
}
```

01/02/2018

Line Clipping

Nicholl - Lee - Nicholl:



Computation time faster than other algorithms but code length is large

Step 1:
Locate one of the end point of a line.

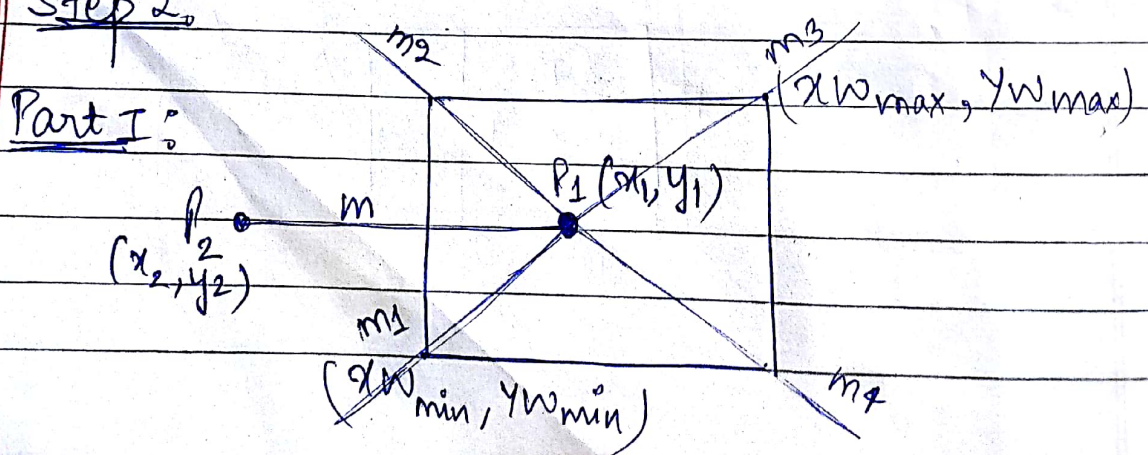
Part I:
End point is inside the rectangular window.

Part II:
End point is on the edge side.
(Left, Right, Top, Bottom)

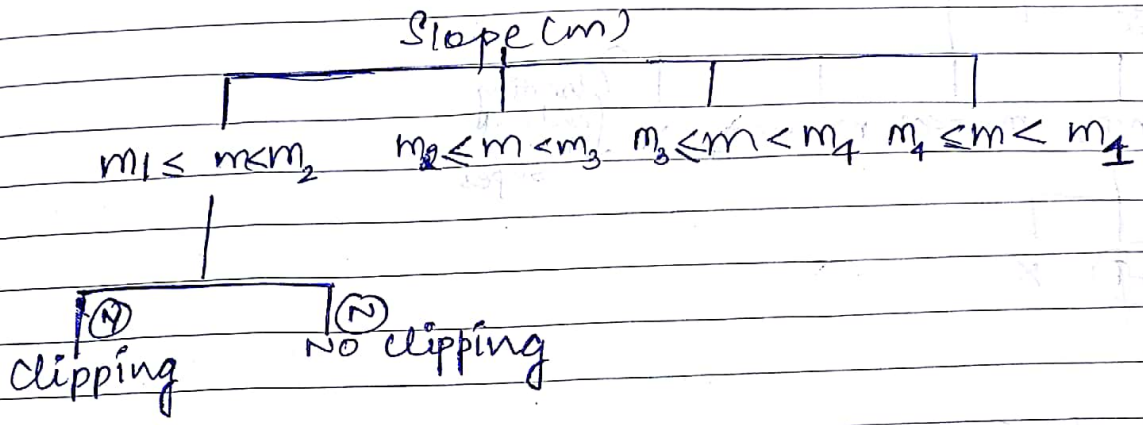
Part III:
End point is on the corner.

Step 2:

Part I:



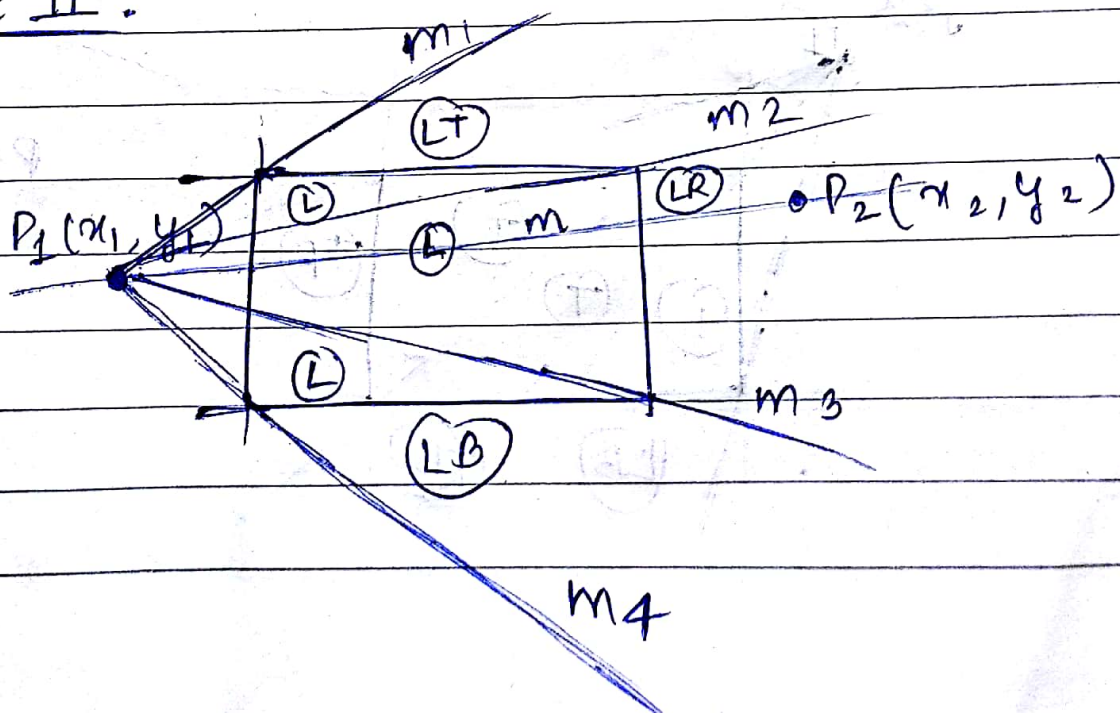
If line $(P_1 - P_2)$ is rotated, then its slope m' will between any of the slopes m_1, m_2, m_3, m_4.

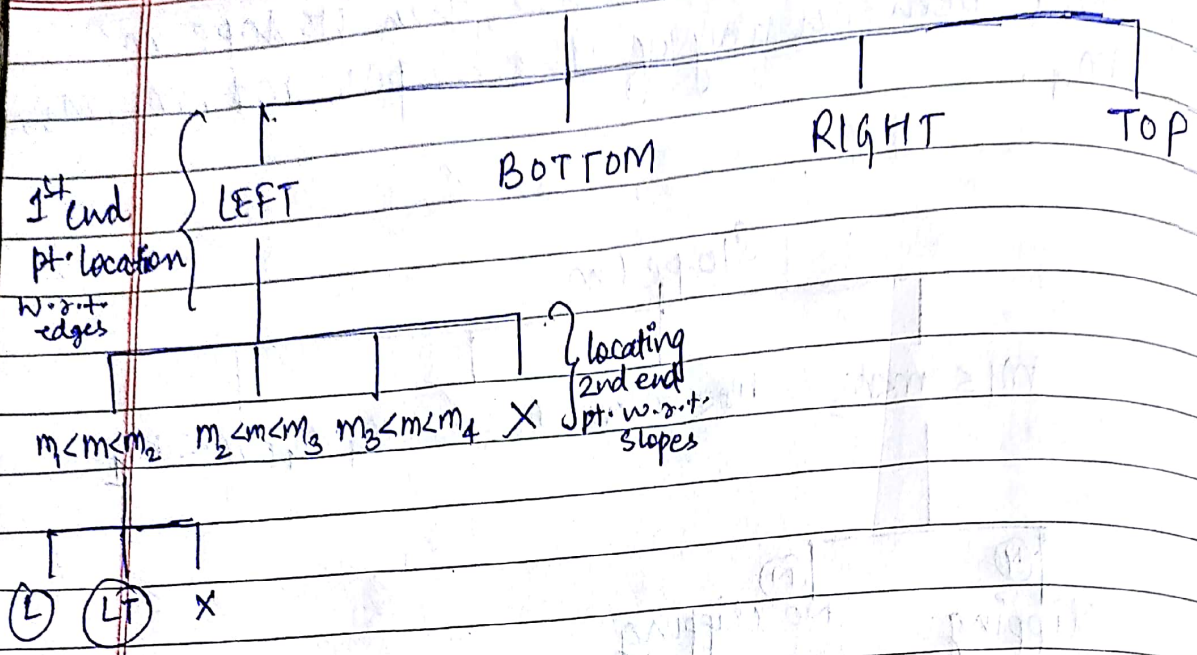



```
if( (m > m2) && (m < m1) && (P1.x1 > P2.x2))
{
    if (P2.x2 < xwmin)
        clipping

    ~~No cli~~
}
```

// Similarly for other cases.

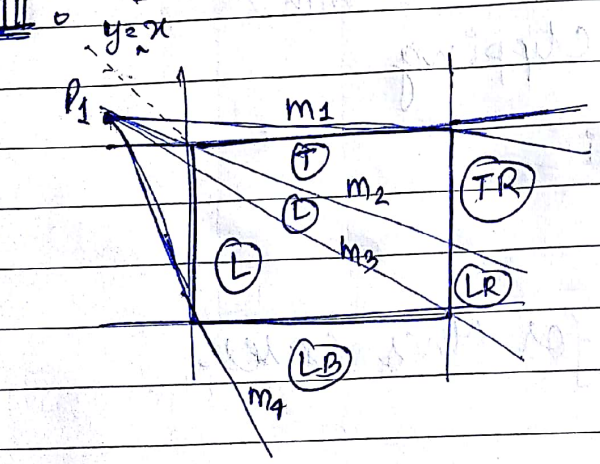
Part II:



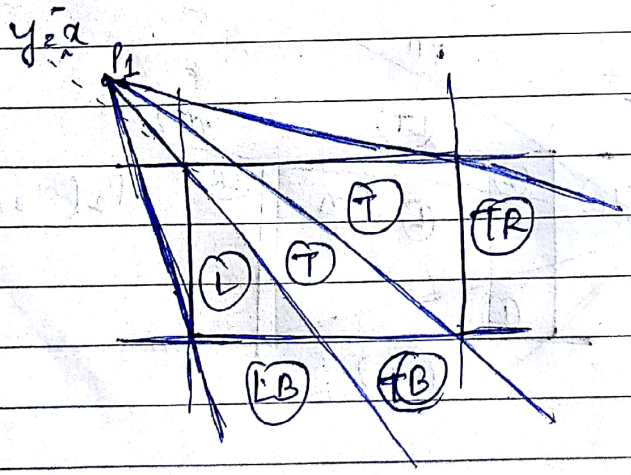


// Similarly for other cases.

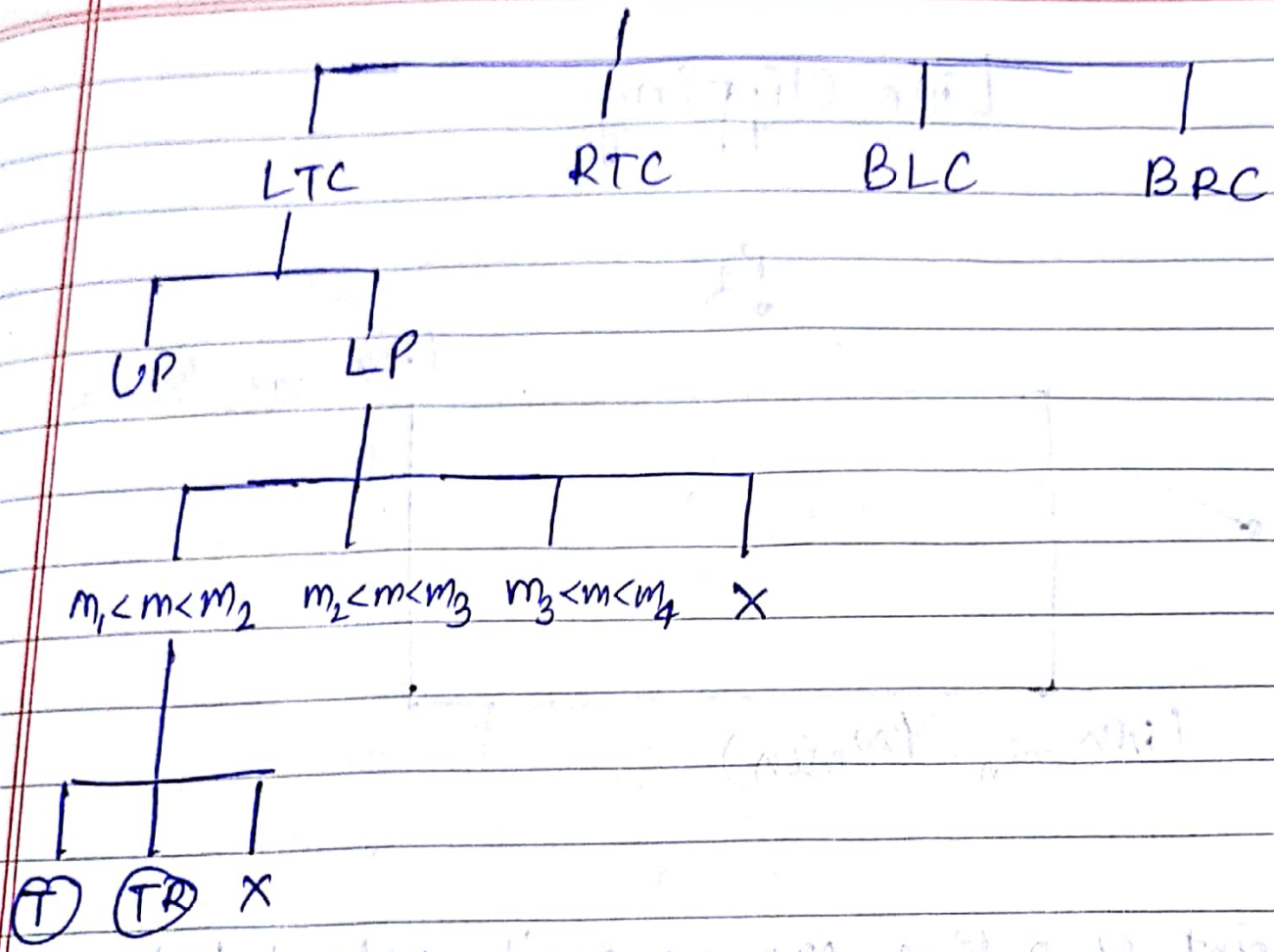
Part III :



P_1 in lower part of $y = x$



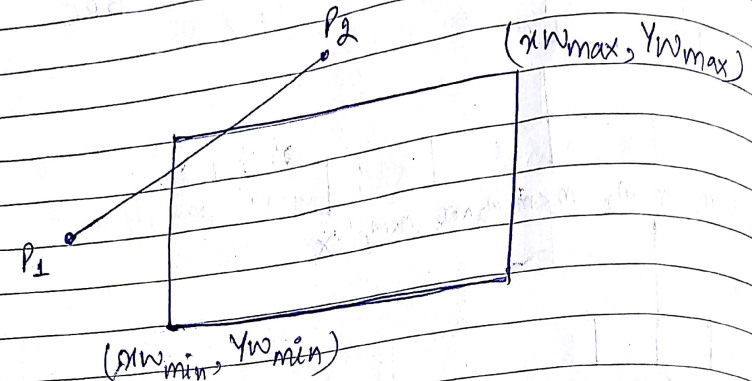
P_1 above $y = x$.



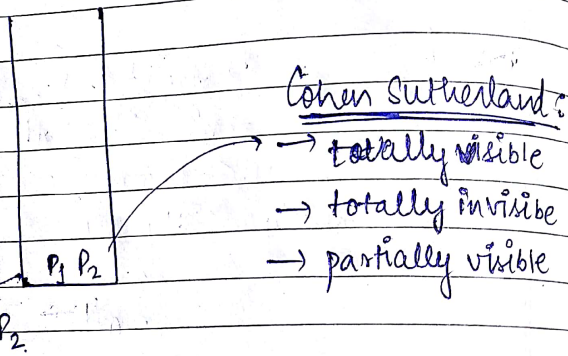
// Similarly for other cases.

05/02/2018

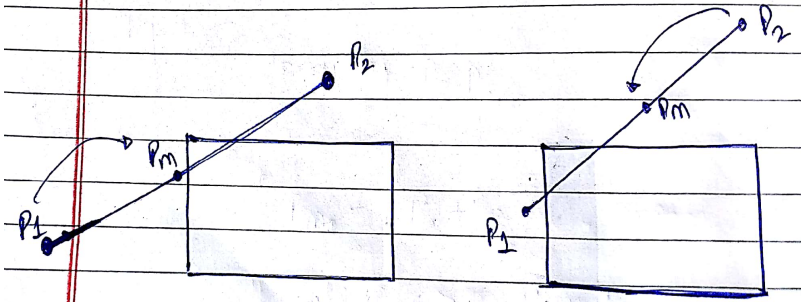
Line Clipping

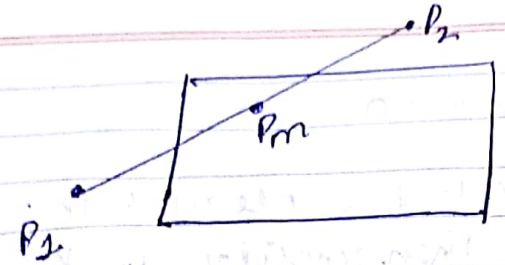


mid point of a line can be easily calculated.



Cohen Sutherland:
- → ~~totally~~ visible
- → totally invisibe
- → partially visible





Input: $P_1(x_1, y_1)$, $P_2(x_2, y_2)$

xw_{min}, xy_{min}, xw_{max}, yw_{max}.

code 1 = gencode (P_1);
code 2 = gencode (P_2);

Step ①

if code1 & code2 both = 0
then line P_1P_2

Step ② if code 1 & code 2 <> 0
/* line invisible */

Step ③ $P_m = (P_1 + P_2)/2$;

code m = gencode (P_m);

Step ④

if code m <> 0
then
if code 1 & code m <> 0 then
$P_1 \leftarrow P_m$; goto step ①

else if code 2 & code m <> 0 then
$P_2 \leftarrow P_m$, goto step ①

Step ⑤

If code m == 0

if code 1 , code m' both = 0
then consider $P_m P_2$

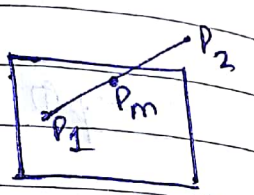
~~else if code~~
else if code 2 ; code m both = 0
then consider $P_1 P_m$.

Step ⑥

consider $P_1 P_m$

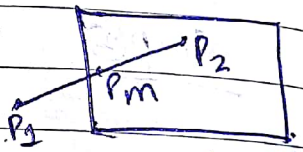
do { $P_{m_1} = (P_1 + P_m)/2$;
code m_1 = gencode(P_{m_1});
if code m_1 <> 0 then $P_1 = P_m$, else
$P_m = P_{m_1}$

}



Step ⑥

while ($P_m.x$ <> xw_{min} &&
$P_m.x$ <> xw_{max} &&
~~$P_m.y$ <> yw_{min} &&~~
$P_m.y$ <> yw_{max})



Step ⑦

$P_1 \leftarrow P_m$.

Step ⑦

consider $P_m P_2$

do {

$P_{m_2} = (P_2 + P_m)/2$; code m_2 = gencode(P_{m_2})

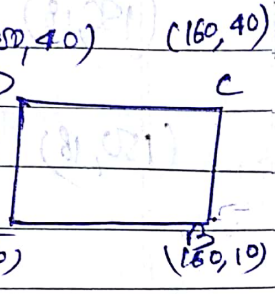
if code m_2 <> 0 then $P_2 = P_{m_2}$, else
$P_m = P_{m_2}$

}

while (same condition as step (6))

$P_2 \leftarrow P_m$

EXAMPLE :



$P_1(120, 5)$, $P_2(180, 30)$

Window ABCD

A(100, 10) , B(160, 10), C(160, 40), D(100, 40)

P_1	Code 1	P_2	Code 2	P_m	Code m.	Remarks
(120,5)	0010	(180,30)	0100	150,18	0000	Consider P_1P_m & consider P_mP_2.

consider P_1P_m

(120,5)	0010	(150,18)	0000	(135,12)	0000	
(120,5)	0010	(135,12)	0000	(128,9)	0010	
(128,9)	0010	(135,12)	0000	(132,11)	0000	
(128,9)	0010	(132,11)	0000	(130,10)	0001	

P_m

(128,9) 0010 (130,10)

(22)

Consider $P_m P_2$

~~(135,12) 0000 150,~~

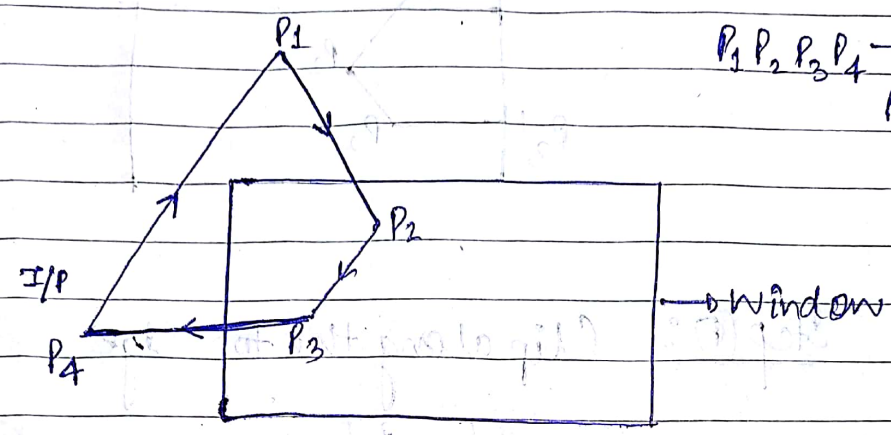
P_1 code 1	P_2 code 2	P_m Codem	Remarks.
(150,18) 0000	(180,30) 0100	(165,24) 0100	
(150,18) 0000	(165,24) 0100		

Polygon Clipping → Window → Rectangular / Non-Rectangular
Polygon → Convex / Concave.

10/2/2018

CLIPPING

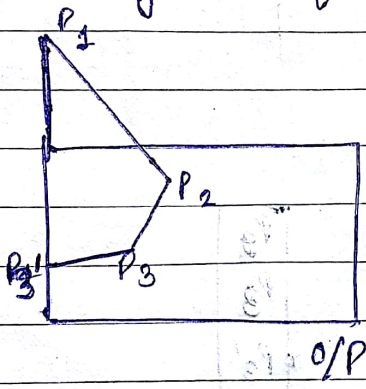
Polygon Clipping



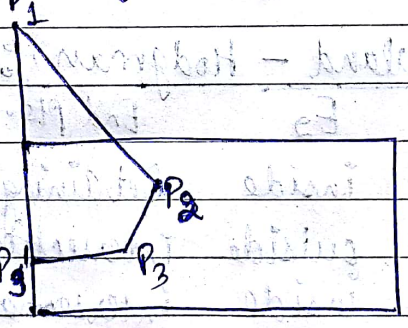
$P_1 P_2 P_3 P_4$ → Subject Polygon

Clipping against rectangular window

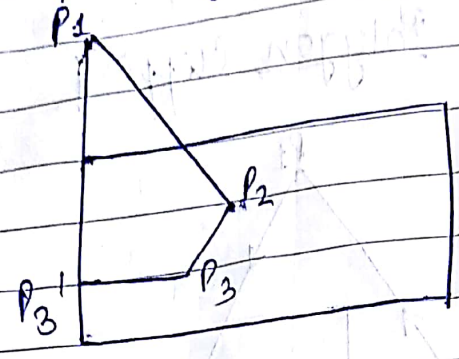
Step ① : Clip along the left edge



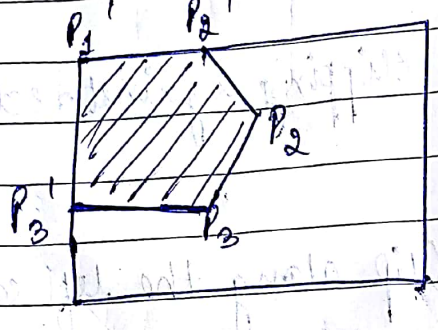
Step ② : Clip along the bottom edge

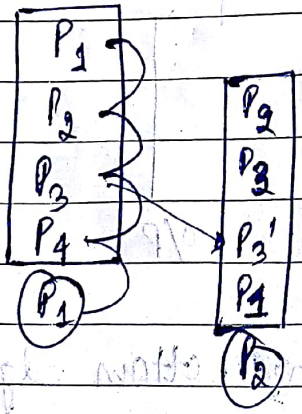


Step ③: Clip along the right edge



Step ④: Clip along the top edge



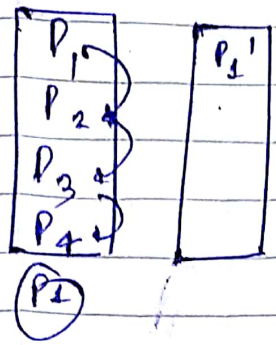


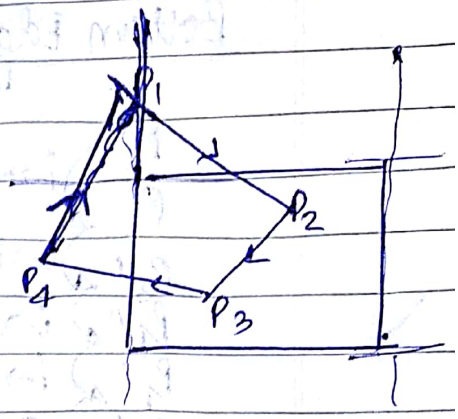
Rules (Sutherland – Hodgman):

E_1	E_2	End pts. of edge (E_1 & E_2)
inside	inside	Retaining the 2nd End Pt. (E_2)
inside	outside	intersection pt. is computed & stored
outside	inside	intersection pt & 2nd end pt to be stored
outside	outside	NIL

EXAMPLE:

i/p | After Processing

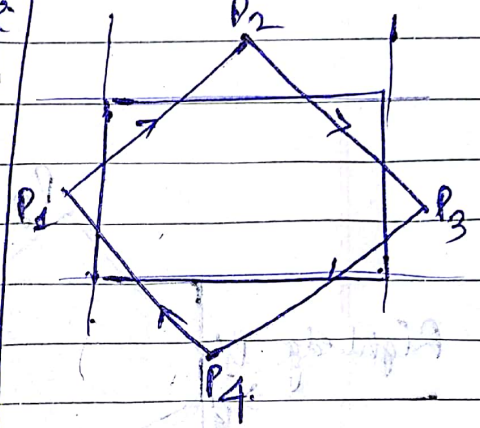




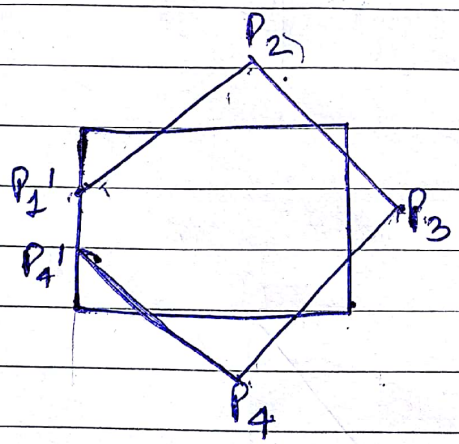
N.o.t. left edge

EXAMPLE:

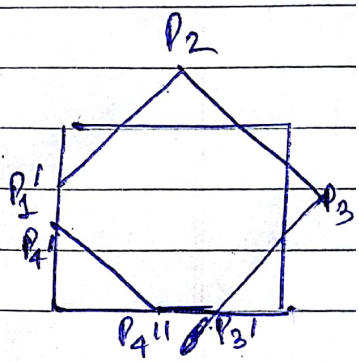
i/p	Left edge	Bottom edge
P1	P1'	P2
P2	P2	P3
P3	P3	P3'
P4	P4	P4''
P1	P4'	P4'
	P1'	P1'
		P2



After clipping at left edge:



Bottom edge.



Bottom Edge	Right Edge	Top Edge
P_2 $i \to 0$		
P_3 $0 \to i$	P_2'	
P_3' $i \to i$	P_3''	
P_4'' $i \to i$	P_3'	
P_4' $i \to i$	P_4''	
P_1' $i \to i$	P_4'	
P_2 $i \to i$	P_1'	
	P_2	
	P_2'