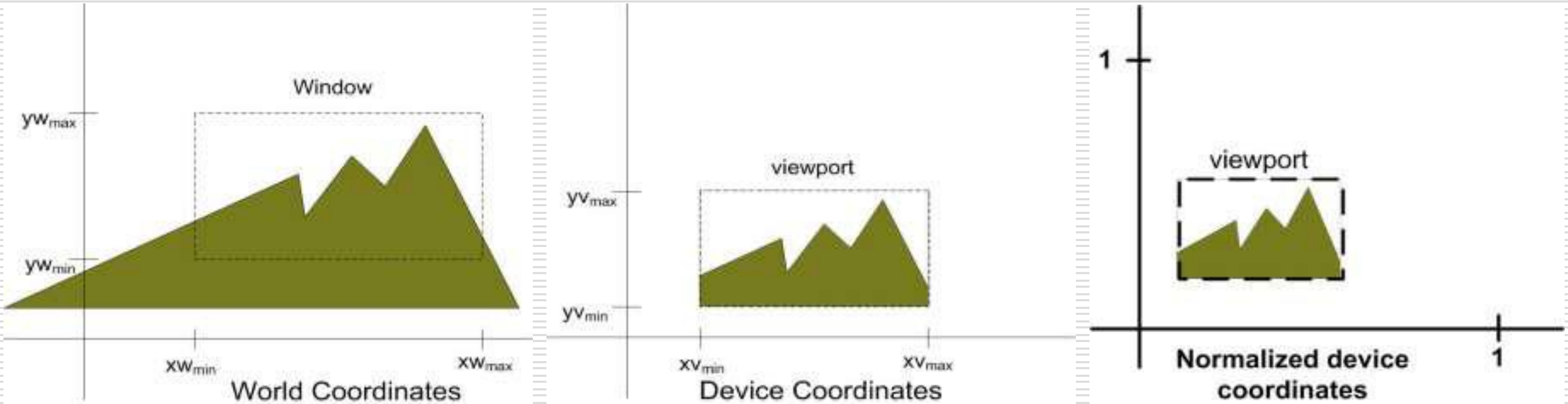
Computer Graphics (L10)

EG678EX

2-D Viewing

Viewing Transformation

- Transformation from world to viewport



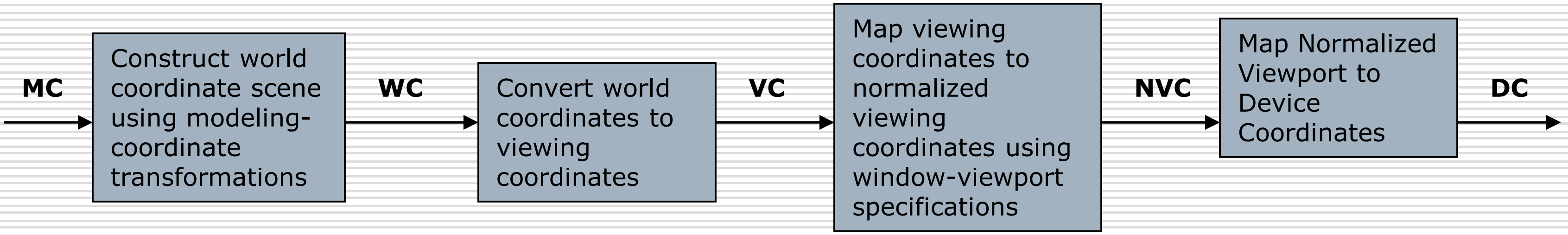
- For Fixed sized viewport, **zooming in** effect is attained if window size is decreased and **zooming out** effect if window size is increased.

- Panning effect is attained by successively changing the position of viewing window

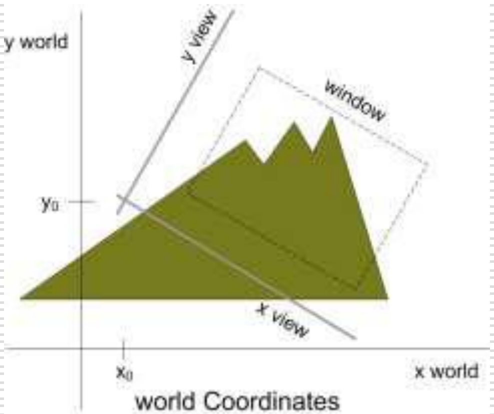
- The normalized device coordinates maps the world co-ordinate to the viewport of maximum size = unit square as shown in figure. The advantage is the mapping is display device independent

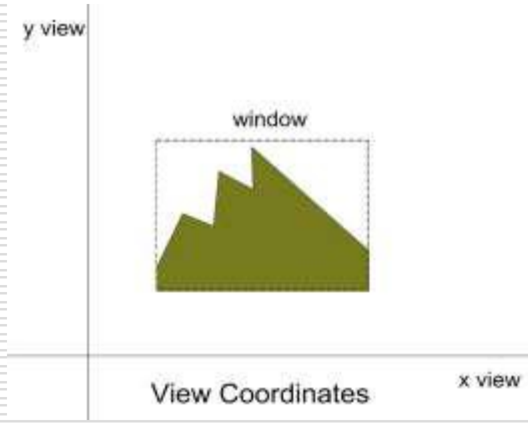
2-D Viewing pipeline

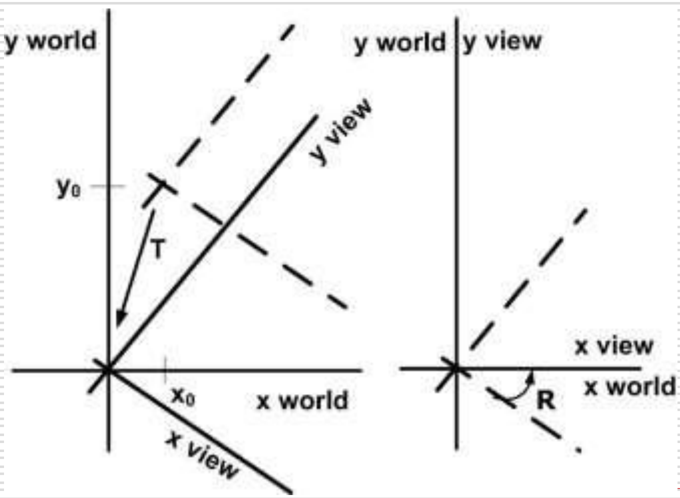
- Procedures for displaying views of a two-dimensional picture on an output device:
  - Specify which parts of the object to display (clipping window, or world window, or viewing window)
  - Where on the screen to display these parts (viewport).
- Clipping window is the selected section of a scene that is displayed on a display window.
- Viewport is the window where the object is viewed on the output device.



Viewing Reference Frame







Setting up viewing Reference Frame

The matrix is obtained in two steps:

1. Translate Viewing reference frame origin to world origin
2. Rotate Viewing reference frame to coincide with world coordinate axes

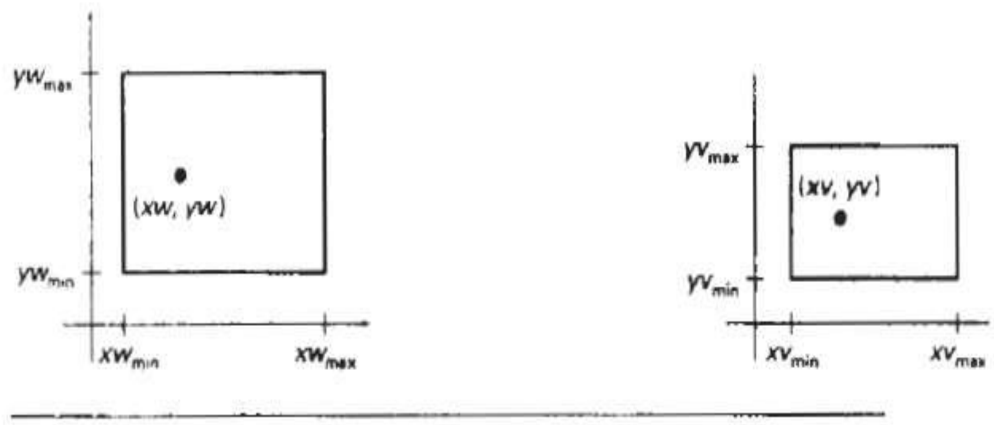
Thus we get the matrix as

$$\mathbf{M_{wc,vc} = R.T}$$

Windows To Viewport Co-ordinate Transformation

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}}$$



Solving these we get

$$xv = xv_{min} + (xw - xw_{min})sx$$

$$yv = yv_{min} + (yw - yw_{min})sy$$

Where scaling factors are

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$

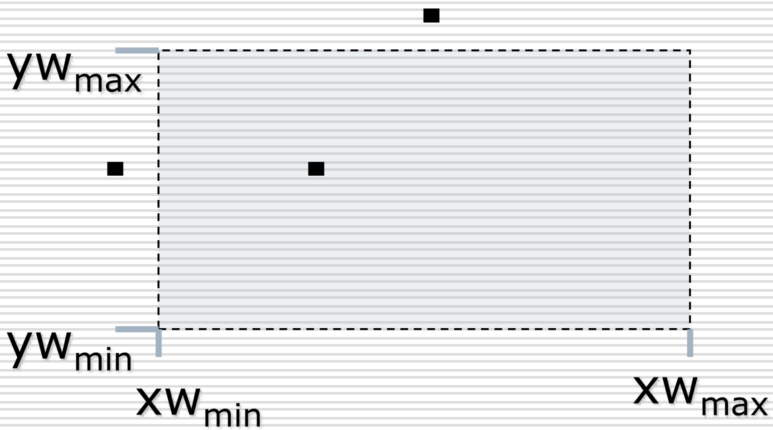
$$sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

If sx = sy, the proportion is maintained otherwise the scene is stretched

Point Clipping

$xw_{min} \le x \le xw_{max}$

$yw_{min} \le y \le yw_{max}$

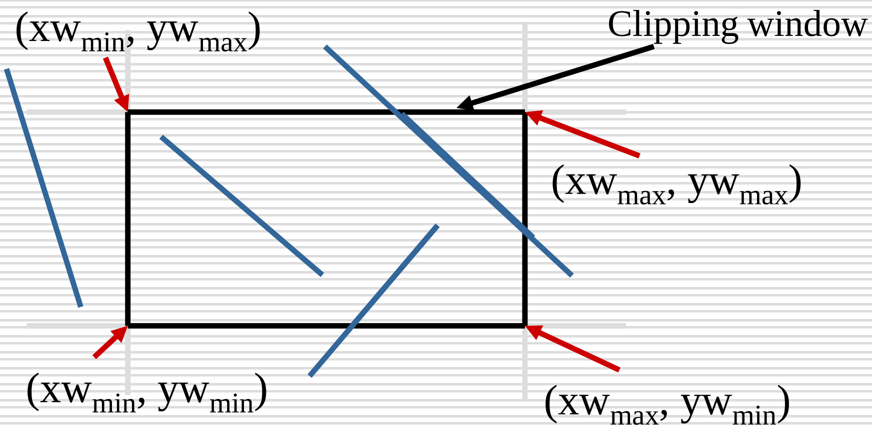


If all the four inequalities are satisfied for a point with co-ordinate (x,y), the point is accepted; i.e not clipped

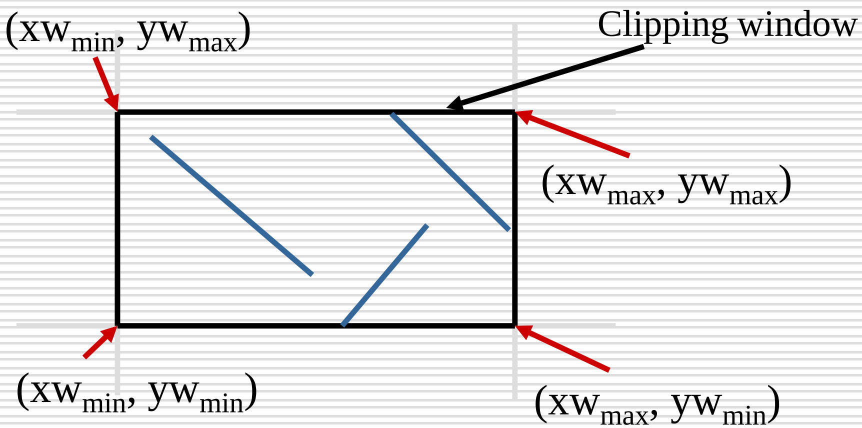
Line Clipping

[Note: most of the slides about line clipping are from some internet resource]

Line Clipping



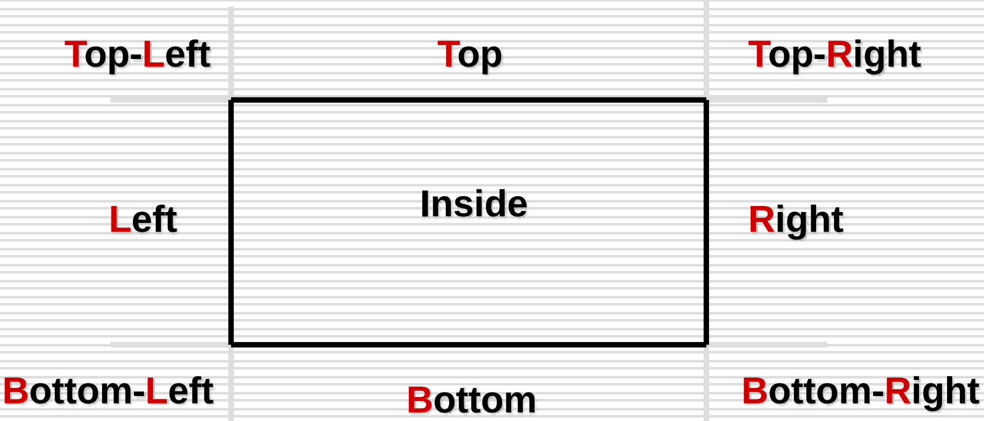
Before clipping

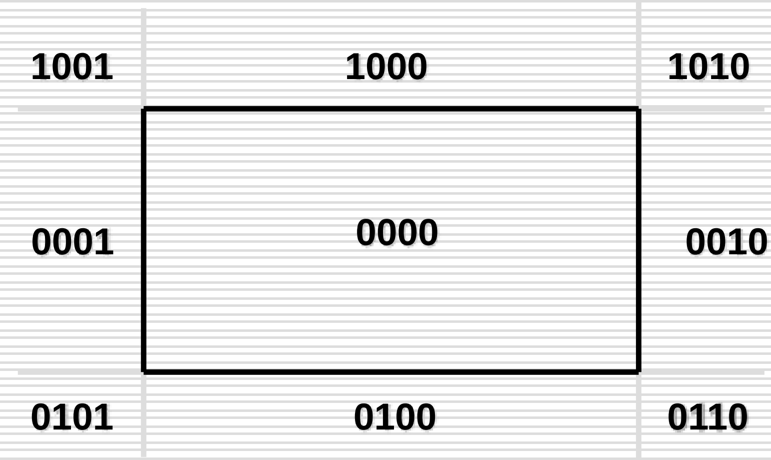


After clipping

What are the methods (algorithms) to perform clipping operations ?

1. Cohen-Sutherland line clipping





LRBT

Region codes	T	B	R	L
Bit position	1	2	3	4

Basic Idea

- label the Left, Right, Bottom and Top of clipping rectangle with region codes
- Test for Trivial Acceptance and Rejection (How?)
- If not trivially accepted or rejected successively clip out the portion of line outside the clip boundary and test whether it is trivially accepted or rejected

Cohen-Sutherland line clipping

Region coding

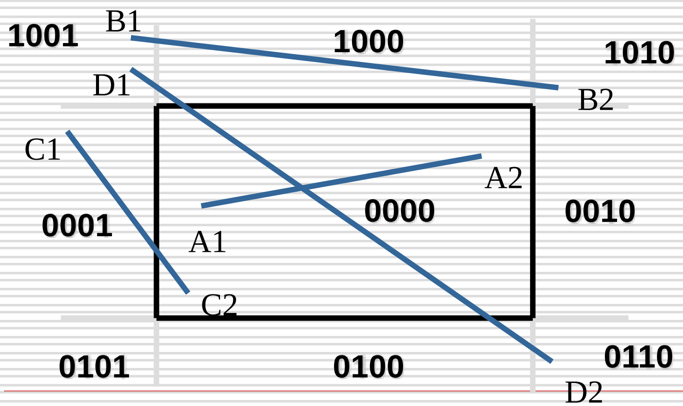
How would you decide which region an endpoint is in?

e.g

if $(x<xw_{min})$ && $(y>y_{max})$ → the point is at the **Top-Left**

Are there cases we can **trivially accept or reject**?

How would you test for those?



- A1=0000,A2=0000
  both (0000) → accept trivially

- B1=1001,B2=1010
  both NOT (0000)
  AND Operation
  B1 → 1001
  B2 → <u>1010</u>
  Result 1000

  Result = NOT (0000) → reject trivially

Cohen-Sutherland line clipping

Algorithm Steps:

1. Assign a region code for each endpoints.
2. If both endpoints have a region code 0000 ---→ trivially accept these line.
3. Else, perform the logical AND operation for both region codes.
   3.1 **if** the result is **NOT** 0000 → trivially reject the line.
   3.2 **else** (i.e. result = 0000, need clipping)
      3.2.1. Choose an endpoint of the line that is outside the window.
      3.2.2. Find the intersection point at the window boundary (base on region code).
      3.2.3. Replace endpoint with the intersection point and update the region code.
      3.2.4. Repeat step 2 until we find a clipped line either trivially accepted or trivially rejected.
4. Repeat step 1 for other lines.

Cohen-Sutherland line clipping

Intersection calculations:

Intersection with vertical boundary

$y = y_1 + m(x-x_1)$

Where

$x = xw_{min}$ or xw_{max}

Intersection with horizontal boundary

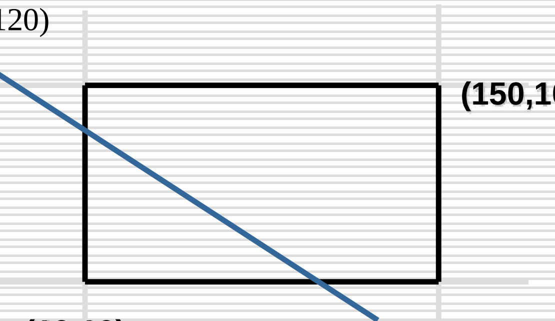
$x = x_1 + (y-y_1)/m$

Where

$y = yw_{min}$ or yw_{max}

Cohen-Sutherland line clipping

- Example:



1. P1=1001, P2=0100
2. (both 0000) – No
3. AND Operation

P1 → 1001

P2 → 0<u>100</u>

Result 0000

3.1 (not 0000) – no

3.2 (0000) yes

3.2.1 choose P2

3.2.2 intersection with BOTTOM boundary

m = (5-120)/(130-0) = -0.8846

x = x1 + (y –y1)/m where y = 10;

x = 130 + (10-5)/ -0.8846 = 124.35 = 124

P2' = (124, 10)

3.2.3 update region code P2' = 0000

3.2. 4 repeat step 2

2. Liang-Barsky Line Clipping

- Based on parametric equation of a line:

  $x = x_1 + u.\Delta x$

  $y = y_1 + u.\Delta y$ $\qquad 0 \le u \le 1$

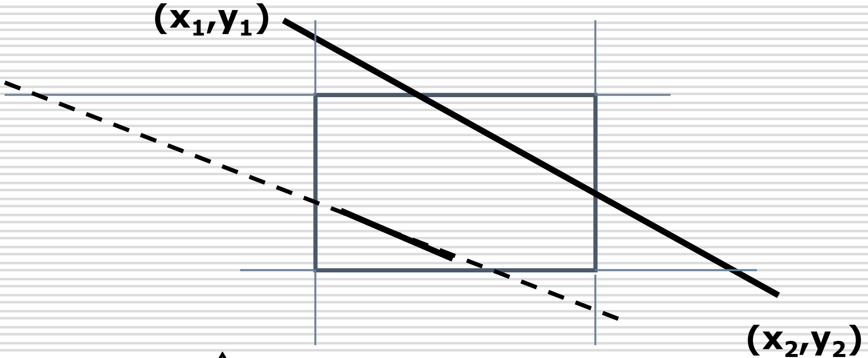
- Similarly, by adopting expressions for point clipping, the clipping window is represented by:

  $xw_{min} \le x_1 + u.\Delta x \le xw_{max}$

  $yw_{min} \le y_1 + u.\Delta y \le yw_{max}$

… or,

  $u.\ p_k \le q_k \qquad k = 1, 2, 3, 4$



- where:

k = 1 (is the line inside left boundary ?) $\quad p_1 = -\Delta x\ ,\ q_1 = x_1 - xw_{min}$

k = 2 (is the line inside right boundary ?) $\quad p_2 = \Delta x\ ,\ q_2 = xw_{max} - x_1$

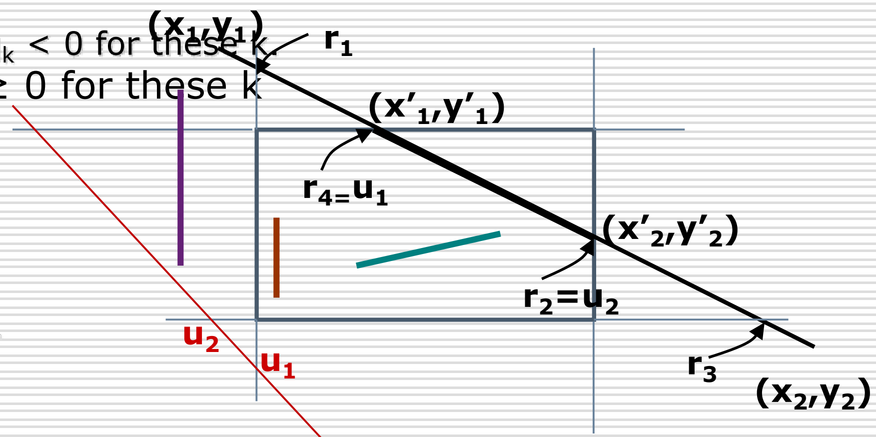
k = 3 (is the line inside bottom boundary ?) $\quad p_3 = -\Delta y\ ,\ q_3 = y_1 - yw_{min}$

k = 4 (is the line inside top boundary ?) $\quad p_4 = \Delta y\ ,\ q_4 = yw_{max} - y_1$

$P_k<0$ infinite extension of the line proceeds from outside to inside the infinitely extended boundary
$P_k>0$ infinite extension of the line proceeds from inside to outside of the infinitely extended boundary

Liang-Barsky Line Clipping

- Trivial rejection
  - Reject line with $p_k = 0$ for some k and one $q_k < 0$ for these k.
- For line with $p_k = 0$ for some k and all $q_k \geq 0$ for these k
  - Line is parallel to one of clip boundary
  - Some portion of line is inside



- For intersection with boundaries the parameters are supposed to be r_k given by

$$r_k = q_k / p_k$$

- Clipped line will be:

$x_1' = x_1 + u_1 . \triangle x;$ $\quad u_1 \geq 0$
$y_1' = y_1 + u_1 . \triangle y;$

$x_2' = x_1 + u_2 . \triangle x;$ $\quad u_2 \leq 1$
$y_2' = y_1 + u_2 . \triangle y;$

u_1 (For intersection with the boundaries to which line enters the boundary) = maximum value between 0 and r (for $p_k < 0$),

u_2 (For intersection with the boundaries to which line leaves the boundary) = minimum value between r and 1 (for $p_k > 0$),

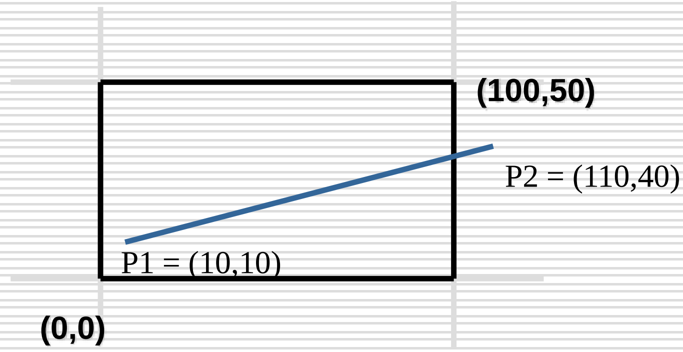
Liang-Barsky Algorithm Steps

1. If $p_k = 0$ for some k then the line is parallel to a clipping boundary. Now test q_k :
   if one $q_k < 0$ for these k then line is outside
   if all $q_k \geq 0$ for these k then some portion of line is inside
2. For all $p_k < 0$ (i.e. line proceeds from outside to inside the boundary) calculate $u = \max(0, \{r_k : r_k = q_k / p_k\})$ to determine intersection point with the possibly extended clipping boundary k and obtain a new starting point for the line at u_1.
3. For all $p_k > 0$ (i.e. line proceeds from inside to outside the boundary) calculate $u_2 = \min(1, \{r_k : r_k = q_k/p_k\})$ to determine intersection point with extended clipping boundary k and obtain a new end point at u_2.
4. If $u_1 > u_2$ then discard the line
5. The line is now between $[u_1, u_2]$

Liang-Barsky Algorithm Steps

- Example:

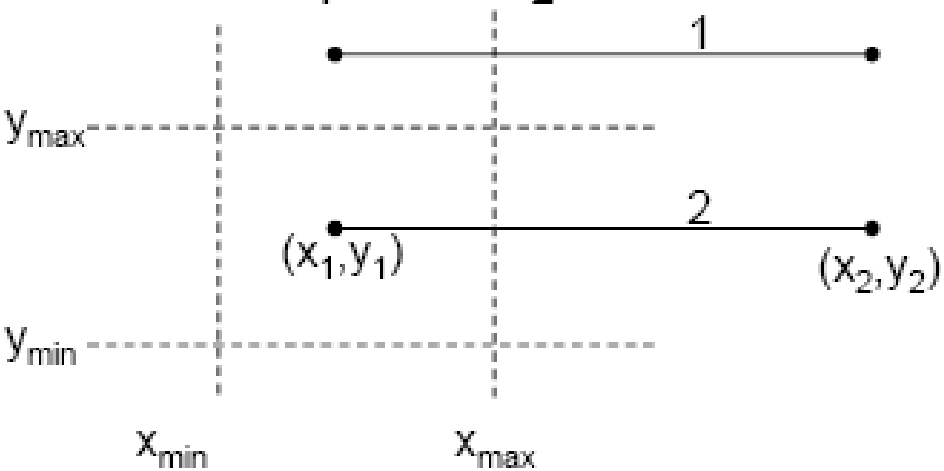
k	p_k	q_k	r_k
1	$-\triangle x$ $= -(110-10)$ $= -100$ i.e $p_k<0$	$x_1 - xw_{min}$ $= 10-0 = 10$	$r_1=10/(-100)$ $=-1/10$ u_1
2	$\triangle x$ $=110-10=100$ i.e $p_k>0$	$xw_{max}- x_1$ $= 100 - 10 = 90$	$r_2=90/100$ $=9/10$ u_2
3	$-\triangle y$ $= -(40-10)$ $=-30$ i.e $p_k<0$	$y_1 - yw_{min}$ $= 10-0 = 10$	$r_3=10/(-30)$ $=-1/3$ u_1
4	$\triangle y$ $= 40-10=30$ i.e $p_k>0$	$yw_{max} - y_1$ $= 50 - 10 = 40$	$r_4=40/30$ $=4/3$ u_2



We take
$u_1 = 0$
And
$u_2 = 0.9$

Line Clipping: Liang-Barsky

- Example 2: Consider horizontal lines with $\Delta y = 0$, $p_3 = p_4 = 0$.
- For line 1, $q_4 < 0$ and will be discarded.
- For line 2, $p_1 < 0$, $p_2 > 0$, $q_3 > 0$ and $q_4 > 0$. We proceed to calculate u_1 and u_2.



Line Clipping: Liang-Barsky

- $p_1 < 0$, note that $q_1 > 0$ hence $q_1/p_1 < 0$, and $u_1 = \max\{0, q_1/p_1\} = 0$
- $p_2 > 0$, calculate q_2/p_2 and $u_2 = \min(q_2/p_2, 1) = q_2/p_2$.
- $u_1 < u_2$ and the line is between $[u_1, u_2]$