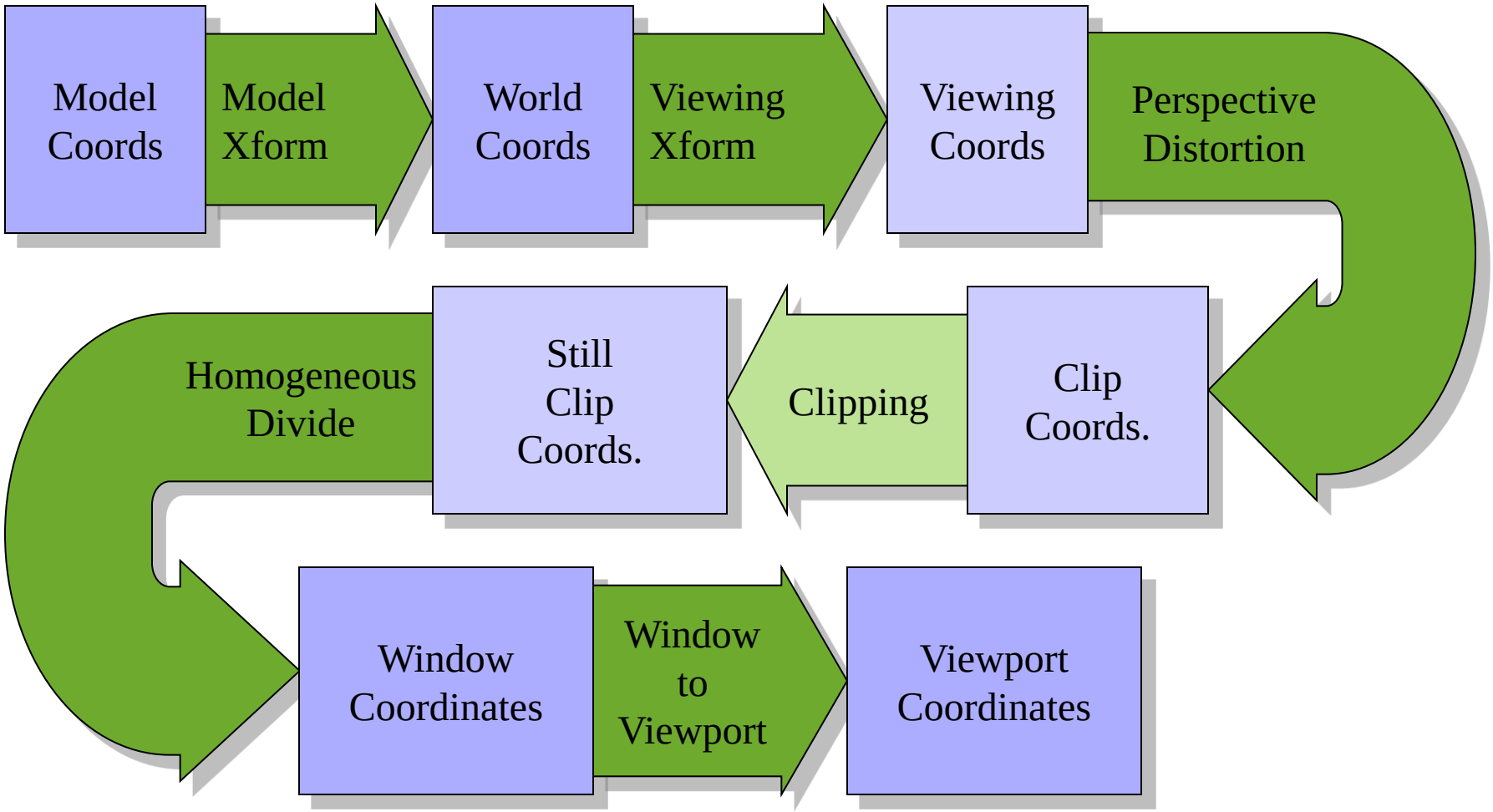
Clipping

CS418 Computer Graphics

John C. Hart

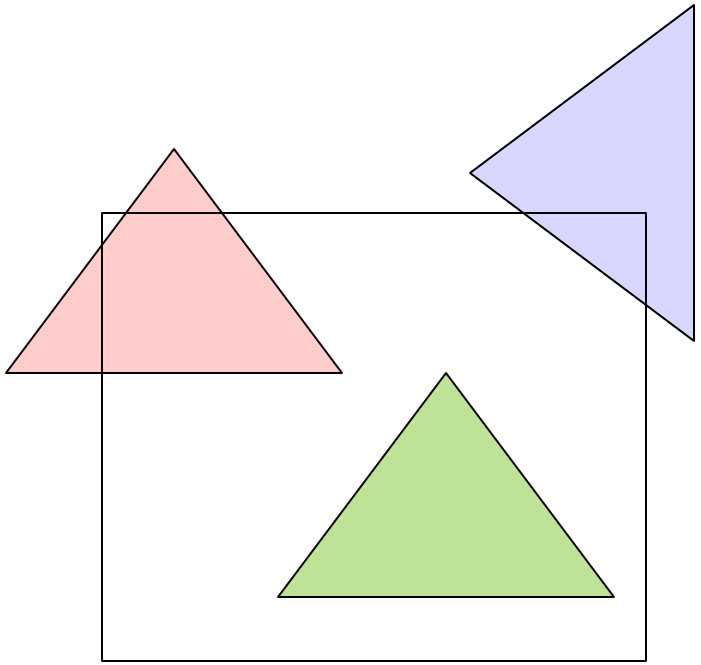
Graphics Pipeline

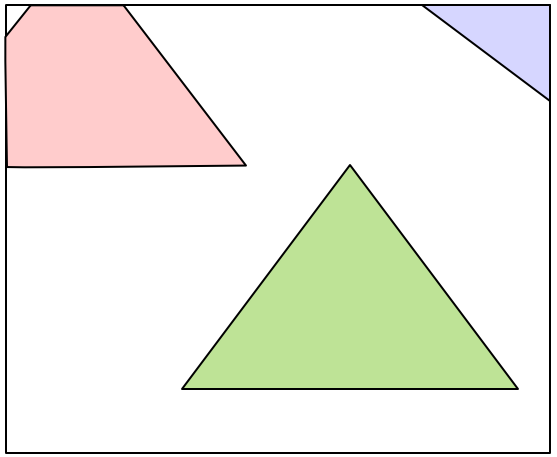


Why Clip?

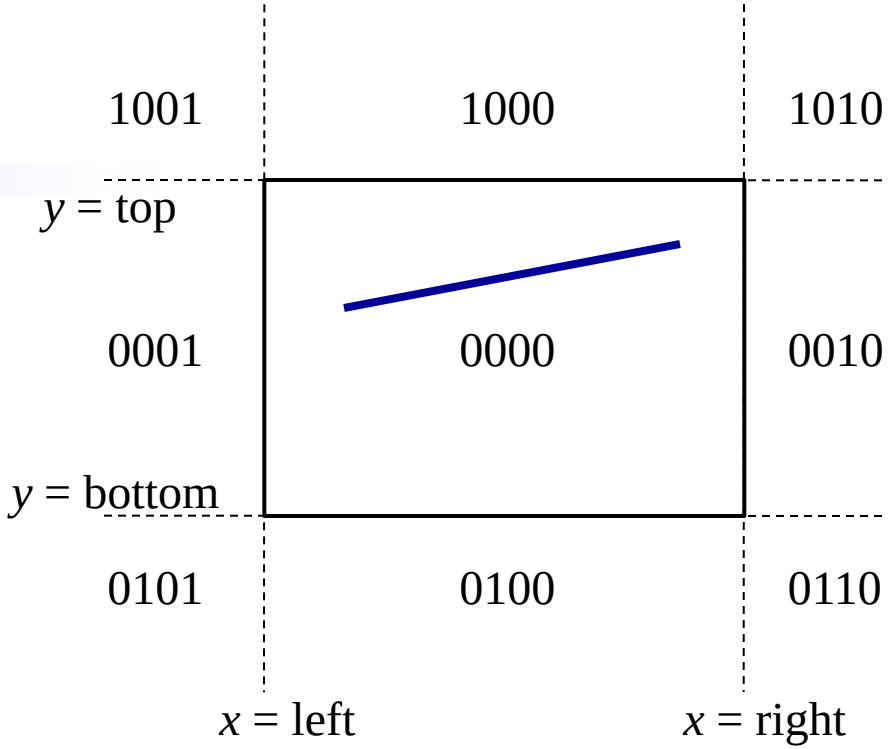
Why not just transform all triangles to the screen and just ignore pixels off the screen?

- Takes time to rasterize a triangle
- Very small number of triangles fall within the viewing frustum
- Output may not go directly to screen





Outcodes

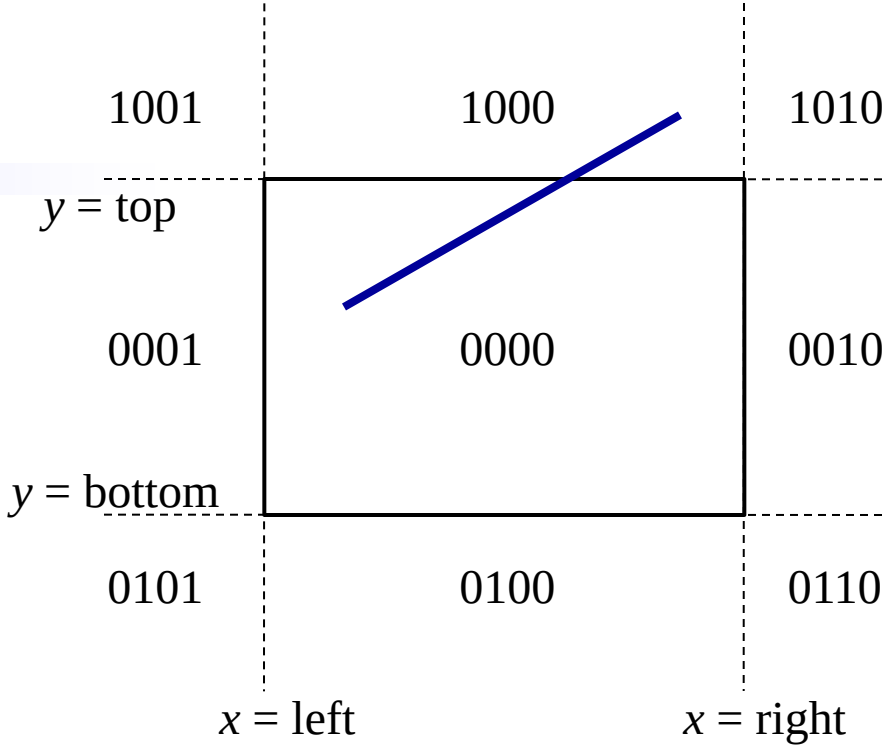


- Cohen-Sutherland
- Assign segment endpoints a bitcode

  $b_3 b_2 b_1 b_0$

  - $b_0 = x <$ left
  - $b_1 = x >$ right
  - $b_2 = y <$ bottom
  - $b_3 = y >$ top
- Let o_0 = outcode(x0,y0), o_1 = outcode(x1,y1)
  - $o_0 = o_1 = 0$: segment visible
  - $o_0 = 0$, $o_1 \neq 0$: segment must be clipped

Outcodes

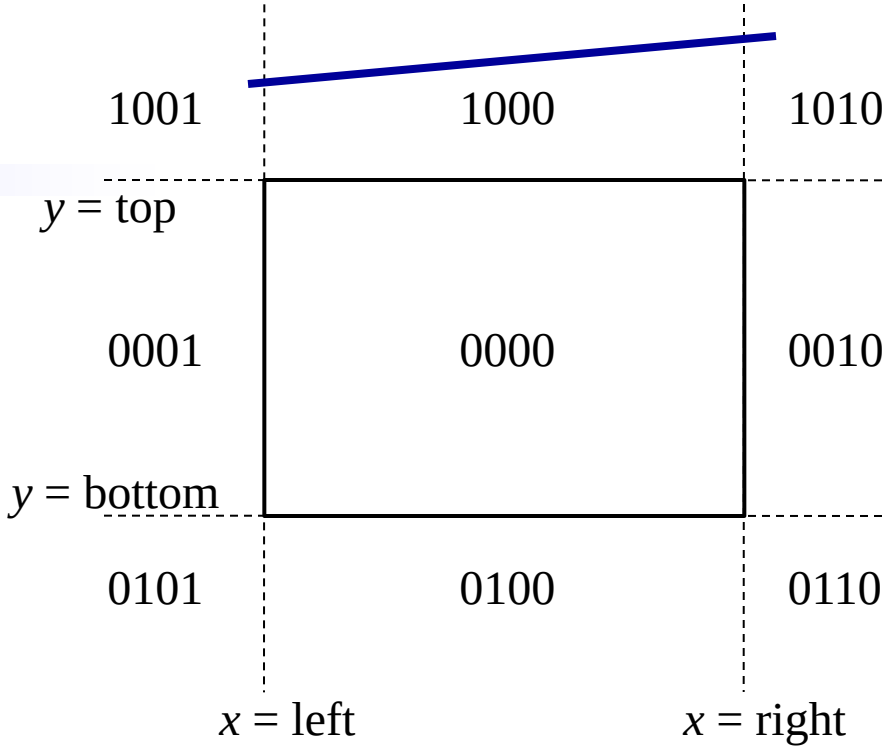


- Cohen-Sutherland
- Assign segment endpoints a bitcode

  $b_3 b_2 b_1 b_0$

  - $b_0 = x <$ left
  - $b_1 = x >$ right
  - $b_2 = y <$ bottom
  - $b_3 = y >$ top
- Let o_0 = outcode(x0,y0), o_1 = outcode(x1,y1)
  - $o_0 = o_1 = 0$: segment visible
  - $o_0 = 0$, $o_1 \neq 0$: segment must be clipped

Outcodes

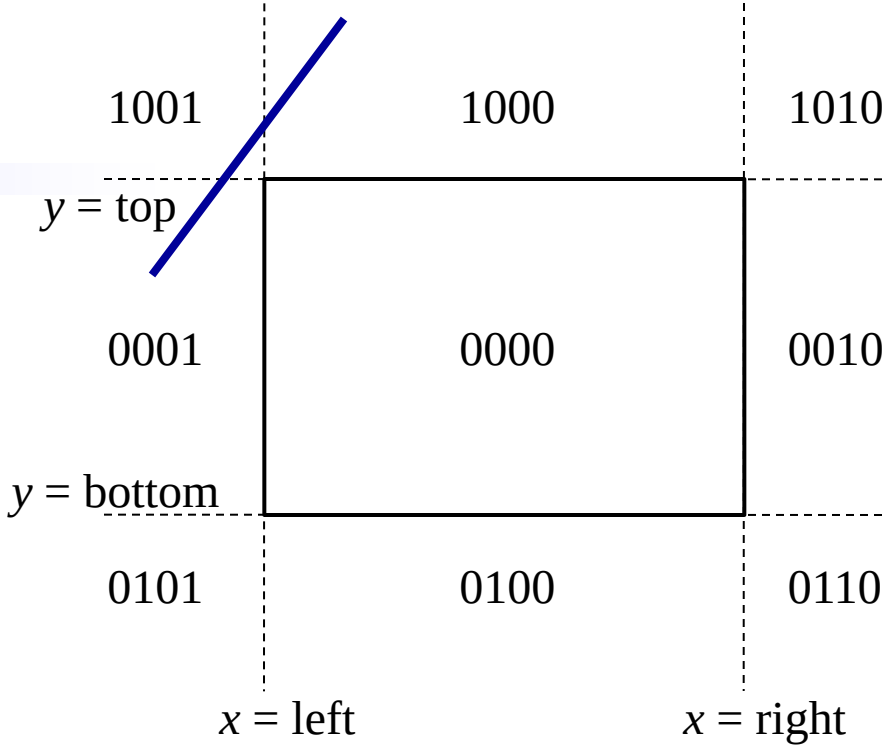


- Cohen-Sutherland
- Assign segment endpoints a bitcode

  $b_3 b_2 b_1 b_0$

  - $b_0 = x <$ left
  - $b_1 = x >$ right
  - $b_2 = y <$ bottom
  - $b_3 = y >$ top
- Let o_0 = outcode(x0,y0), o_1 = outcode(x1,y1)
  - $o_0 = o_1 = 0$: segment visible
  - $o_0 = 0$, $o_1 \neq 0$: segment must be clipped
  - o_0 & $o_1 \neq 0$: segment can be ignored

Outcodes

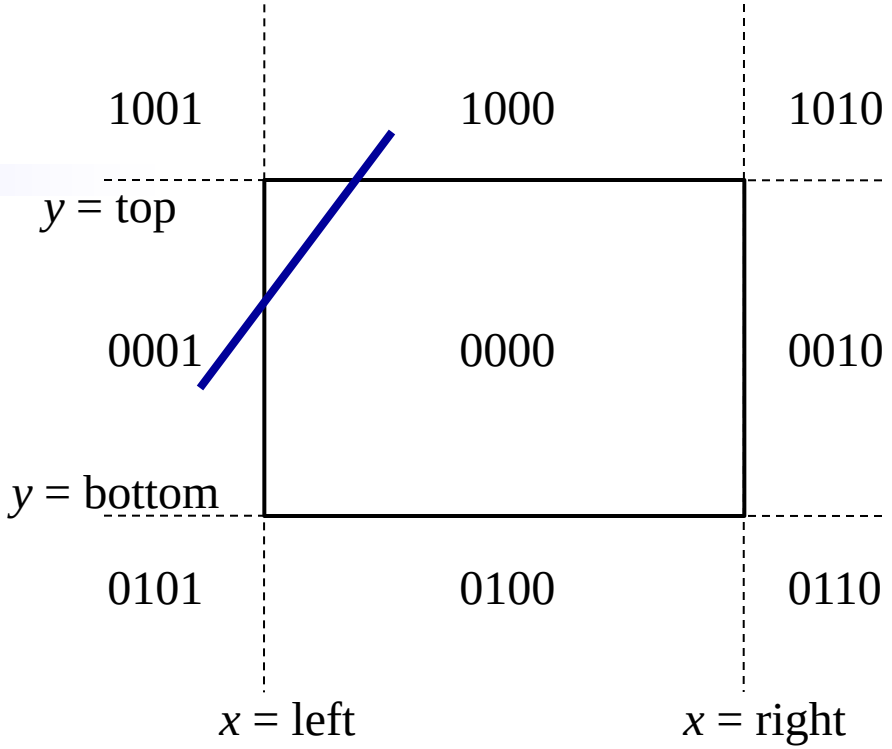


- Cohen-Sutherland
- Assign segment endpoints a bitcode

  $b_3 b_2 b_1 b_0$

  - $b_0 = x <$ left
  - $b_1 = x >$ right
  - $b_2 = y <$ bottom
  - $b_3 = y >$ top
- Let o_0 = outcode(x0,y0), o_1 = outcode(x1,y1)
  - $o_0 = o_1 = 0$: segment visible
  - $o_0 = 0$, $o_1 \neq 0$: segment must be clipped
  - o_0 & $o_1 \neq 0$: segment can be ignored
  - o_0 & $o_1 = 0$: segment might need clipping

Outcodes

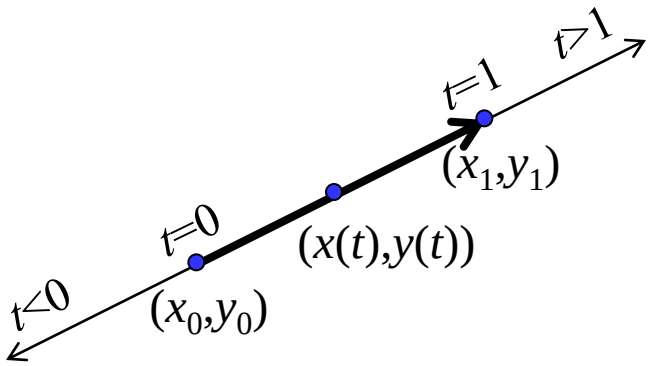


- Cohen-Sutherland
- Assign segment endpoints a bitcode

  $b_3 b_2 b_1 b_0$

  - $b_0 = x <$ left
  - $b_1 = x >$ right
  - $b_2 = y <$ bottom
  - $b_3 = y >$ top
- Let o_0 = outcode(x0,y0), o_1 = outcode(x1,y1)
  - $o_0 = o_1 = 0$: segment visible
  - $o_0 = 0$, $o_1 \neq 0$: segment must be clipped
  - o_0 & $o_1 \neq 0$: segment can be ignored
  - o_0 & $o_1 = 0$: segment might need clipping

Intersecting Lines

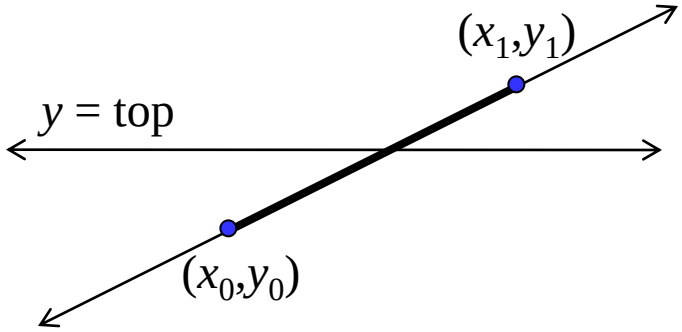


- Parametric representation of a line segment

$$x(t) = x_0 + t\,(x_1 - x_0)$$

$$y(t) = y_0 + t\,(y_1 - y_0)$$

Intersecting Lines



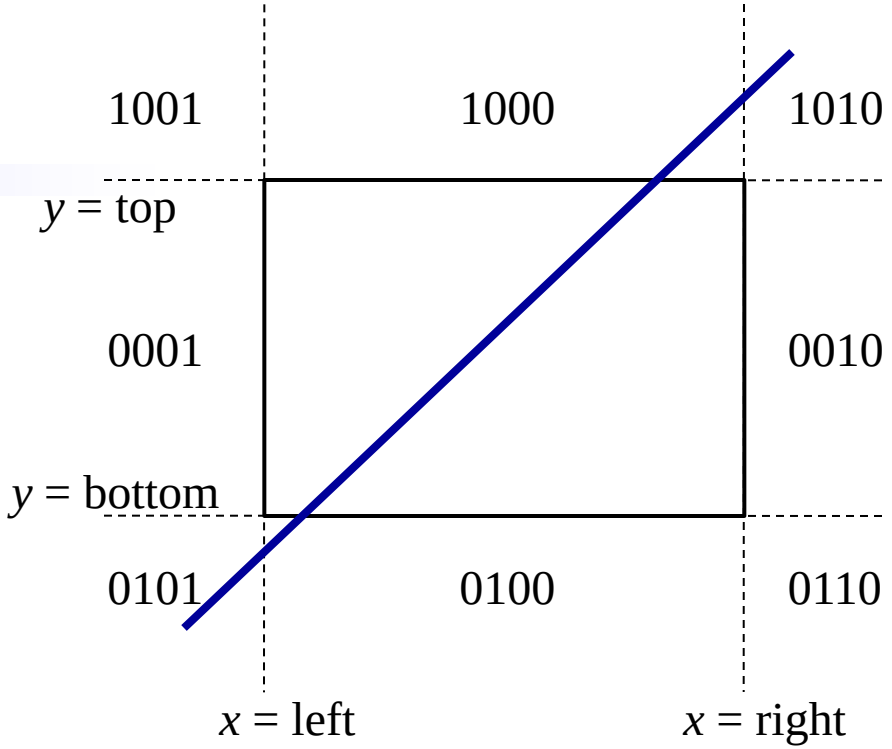
- Parametric representation of a line segment

$$x(t) = x_0 + t\,(x_1 - x_0)$$
$$y(t) = y_0 + t\,(y_1 - y_0)$$

- Plug in clipping window edge to find t

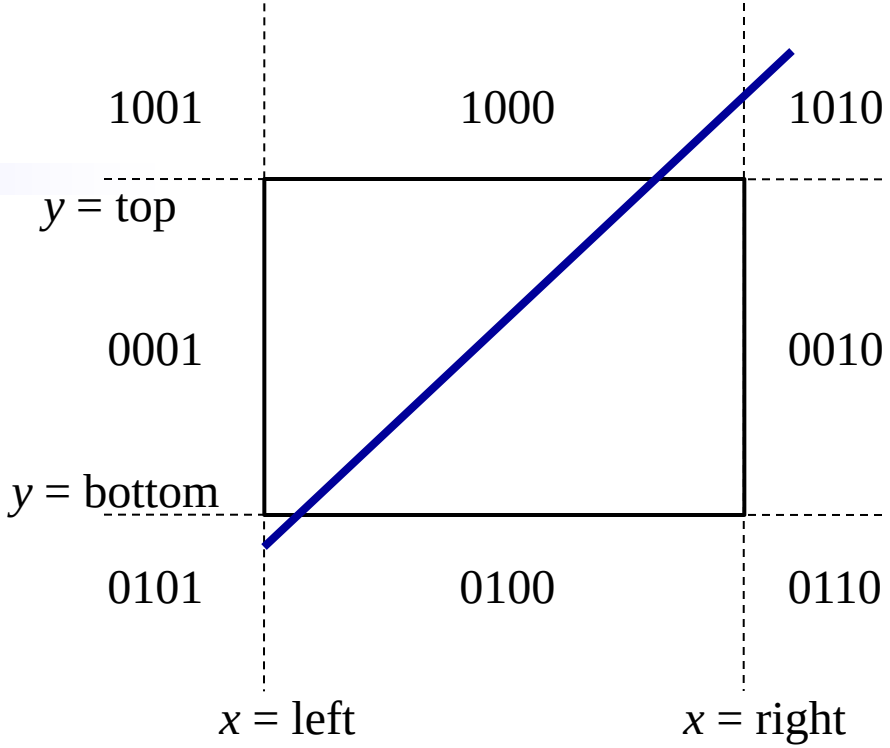
$$\text{top} = y_0 + t\,(y_1 - y_0)$$
$$t = (\text{top} - y_0)/(y_1 - y_0)$$

Serial Clipping



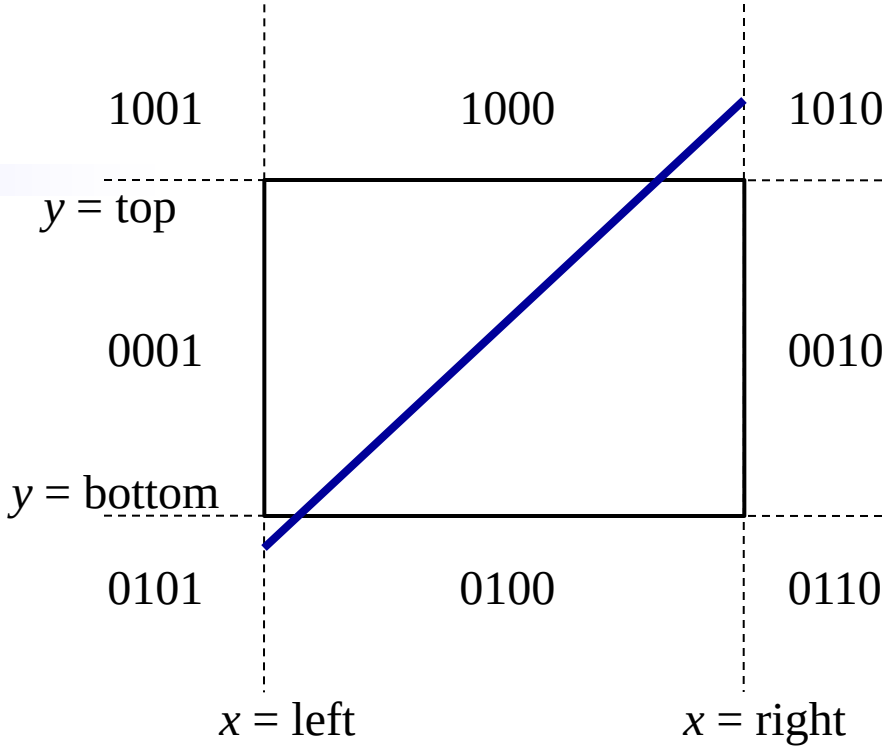
Serial Clipping

- First clip 0001
- Move (x_0,y_0) to (left,…)



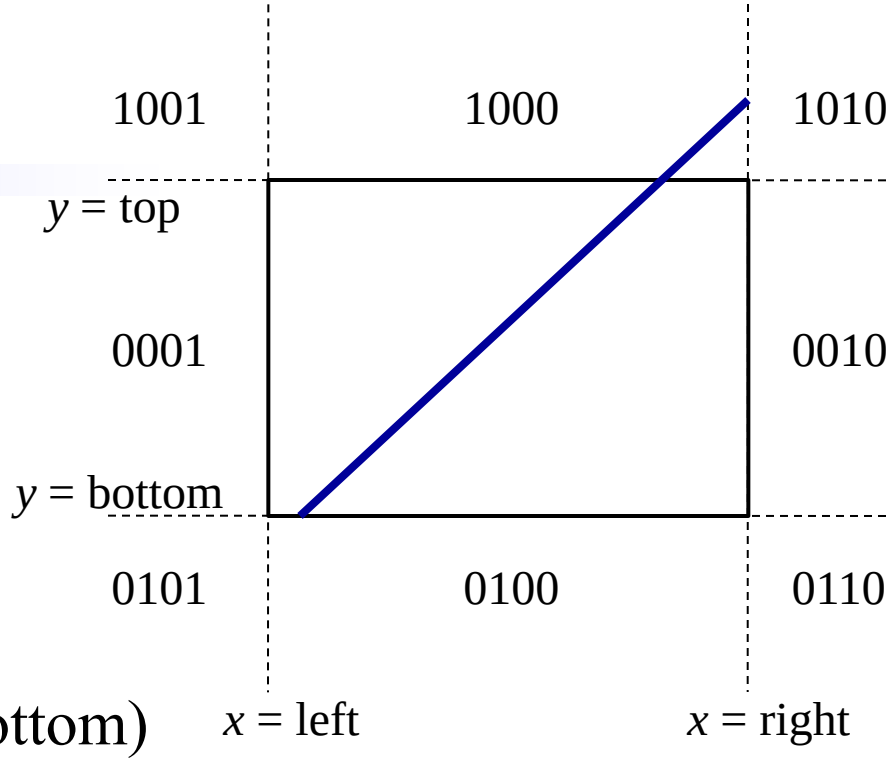
Serial Clipping

- First clip 0001
- Move (x_0,y_0) to (left,…)
- Then clip 0010
- Move (x_1,y_1) to (right,…)

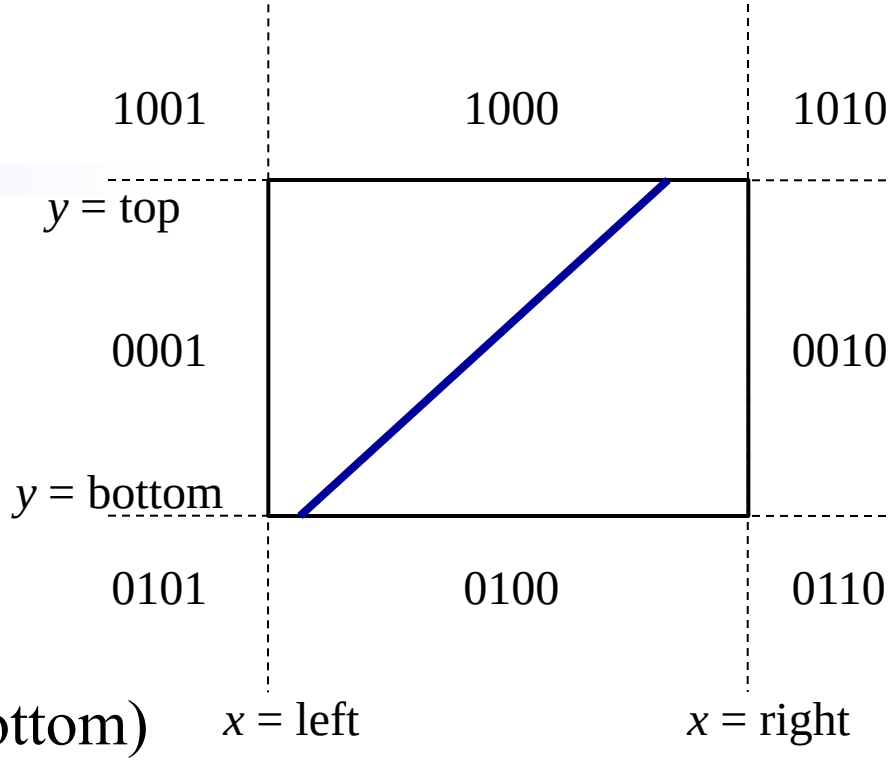


Serial Clipping

- First clip 0001
- Move (x_0,y_0) to (left,…)
- Then clip 0010
- Move (x_1,y_1) to (right,…)
- Then clip 0100
- Move (x_0,y_0) again, now to (…,bottom)



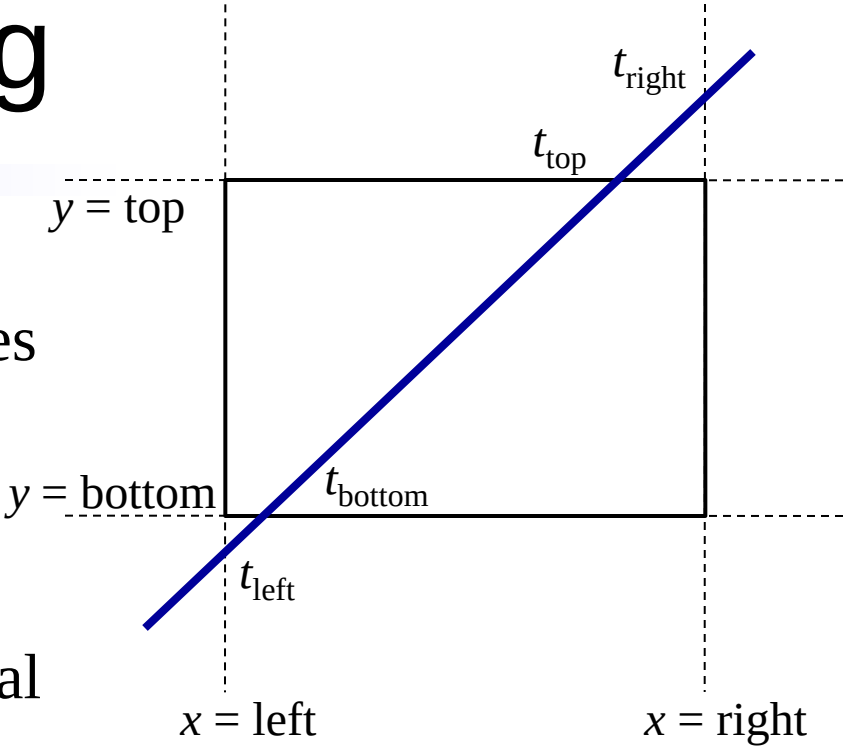
Serial Clipping



- First clip 0001
- Move (x_0,y_0) to (left,…)
- Then clip 0010
- Move (x_1,y_1) to (right,…)
- Then clip 0100
- Move (x_0,y_0) again, now to (…,bottom)
- Finally clip 1000
- Move (x_1,y_1) again, now to (…,top)

Parametric Clipping

- Cohen-Sutherland iterates through each edge intersection and computes them even if they are later clipped
- Liang-Barsky computes all at once postponing division to end
- Unless a line is horizontal or vertical it intersects all four clipping planes
- Compute all four intersections



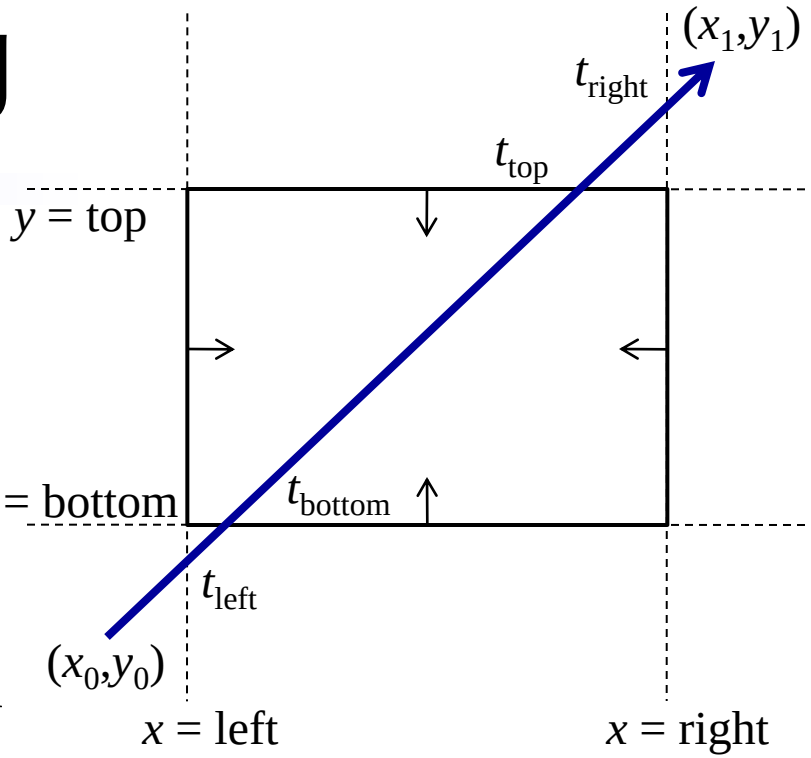
$$t_{left} = (left - x_0) / (x_1 - x_0)$$
$$t_{right} = (right - x_0) / (x_1 - x_0)$$
$$t_{bottom} = (bottom - y_0) / (y_1 - y_0)$$
$$t_{top} = (top - y_0) / (y_1 - y_0)$$

Parametric Clipping

- Cohen-Sutherland iterates through each edge intersection and computes them even if they are later clipped
- Liang-Barsky computes all at once postponing division to end
- Unless a line is horizontal or vertical it intersects all four clipping planes
- Compute oriented numerators and denominators of all four intersections
- Sign of denominator:

  positive ⇔ entering

  negative ⇔ leaving

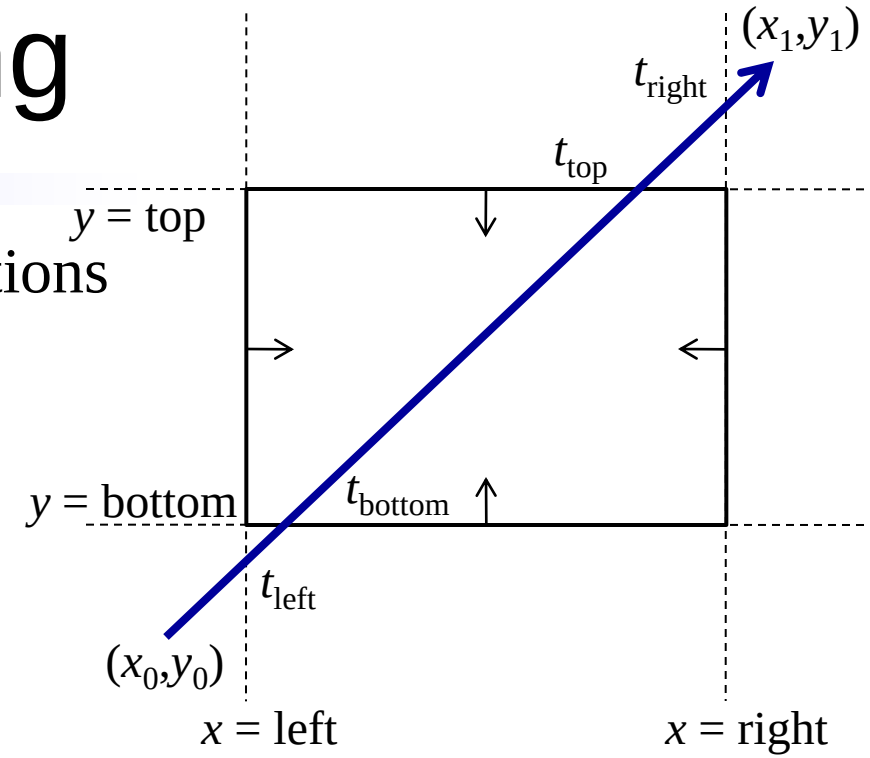


$t_{left} = (left - x_0) / (x_1 - x_0)$
$t_{right} = -(right - x_0) / -(x_1 - x_0)$
$t_{bottom} = (bottom - y_0) / (y_1 - y_0)$
$t_{top} = -(top - y_0) / -(y_1 - y_0)$

Parametric Clipping

- Compute ordered list of t-intersections

$$0 < t_{left} < t_{bottom} < t_{top} < t_{right} < 1$$



$$t_{left} = (left - x_0) / (x_1 - x_0)$$
$$t_{right} = -(right - x_0) / -(x_1 - x_0)$$
$$t_{bottom} = (bottom - y_0) / (y_1 - y_0)$$
$$t_{top} = -(top - y_0) / -(y_1 - y_0)$$

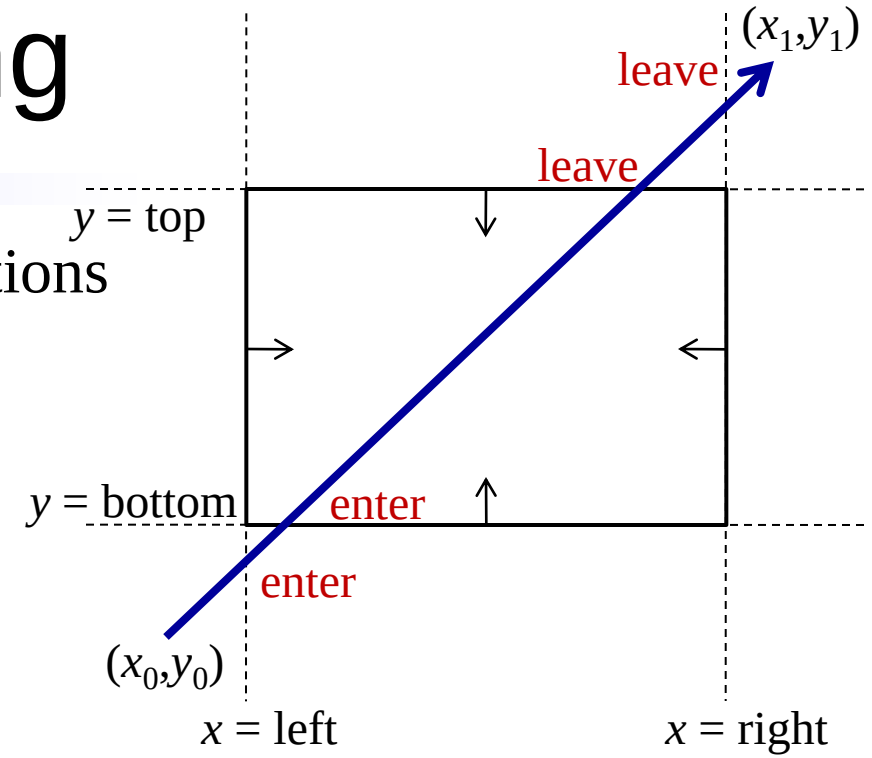
Parametric Clipping

- Compute ordered list of t-intersections

  $0 < t_{left} < t_{bottom} < t_{top} < t_{right} < 1$

- And corresponding orientations

  enter, enter, leave, leave



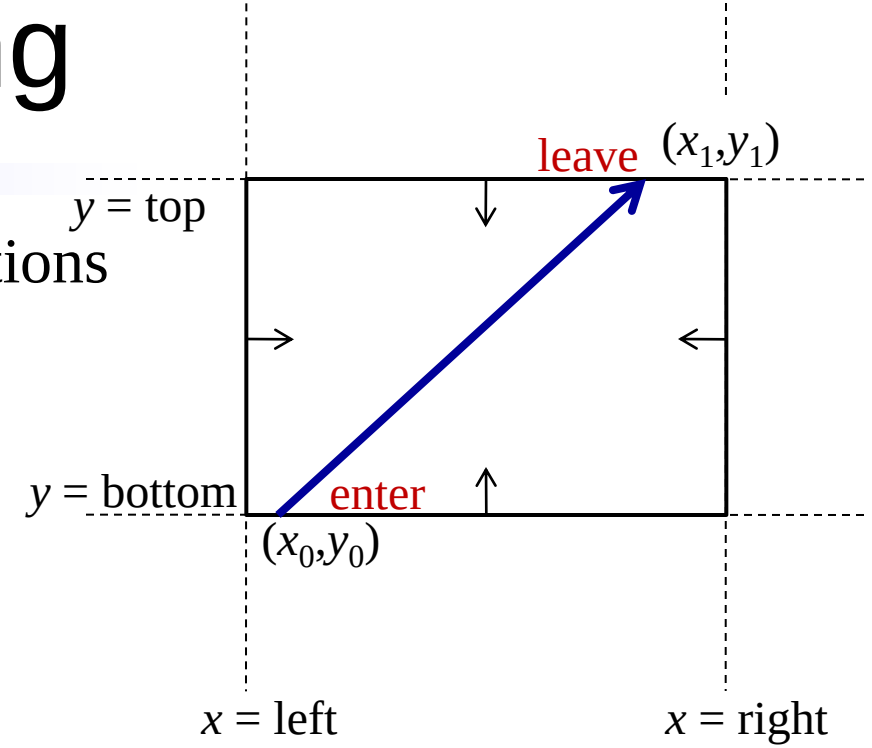
$t_{left} = (\text{left} - x_0) / (x_1 - x_0)$

$t_{right} = -(\text{right} - x_0) / -(x_1 - x_0)$

$t_{bottom} = (\text{bottom} - y_0) / (y_1 - y_0)$

$t_{top} = -(\text{top} - y_0) / -(y_1 - y_0)$

Parametric Clipping



- Compute ordered list of t-intersections

  $0 < t_{left} < t_{bottom} < t_{top} < t_{right} < 1$

- And corresponding orientations

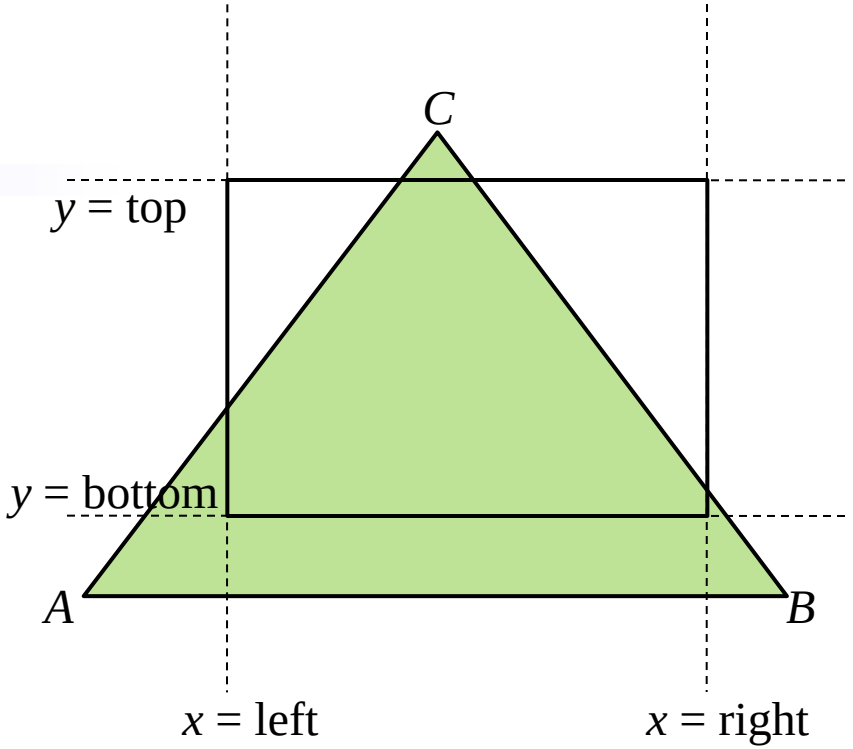
  enter, enter, leave, leave

- Keep only the <enter,leave> pair

$t_{left} = (left - x_0) / (x_1 - x_0)$
$t_{right} = -(right - x_0) / -(x_1 - x_0)$
$t_{bottom} = (bottom - y_0) / (y_1 - y_0)$
$t_{top} = -(top - y_0) / -(y_1 - y_0)$

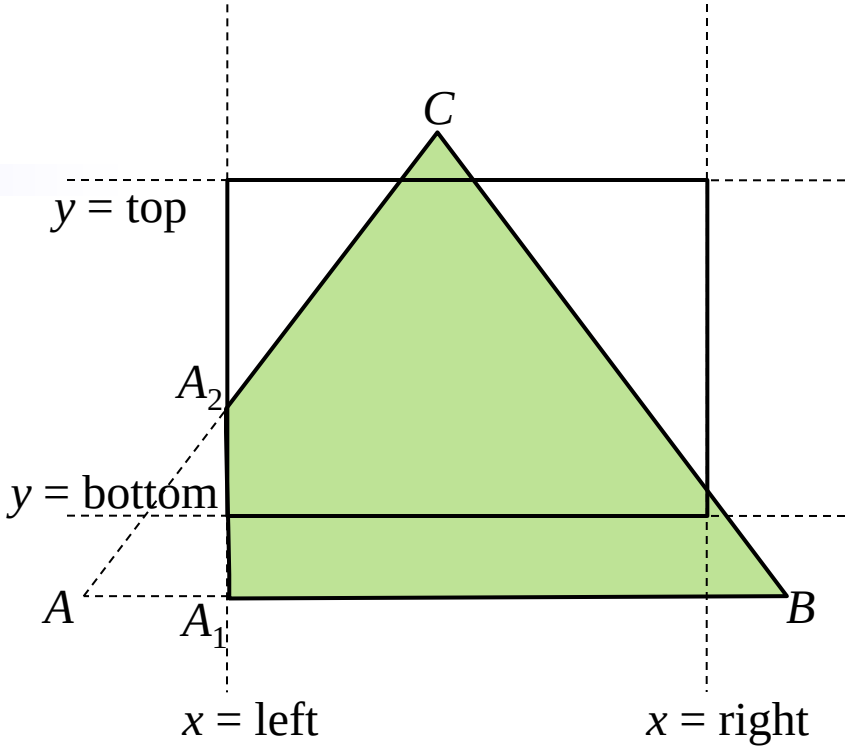
Polygon Clipping

- Sutherland-Hodgman
- Polygon *ABC*



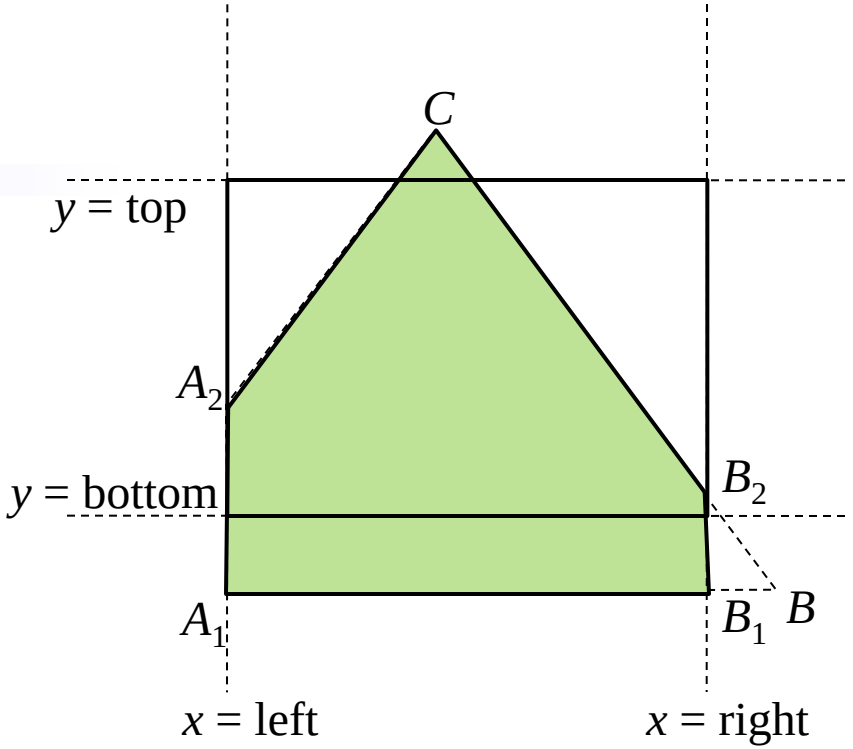
Polygon Clipping

- Sutherland-Hodgman
- Polygon *ABC*
- Clip left: A_1BCA_2



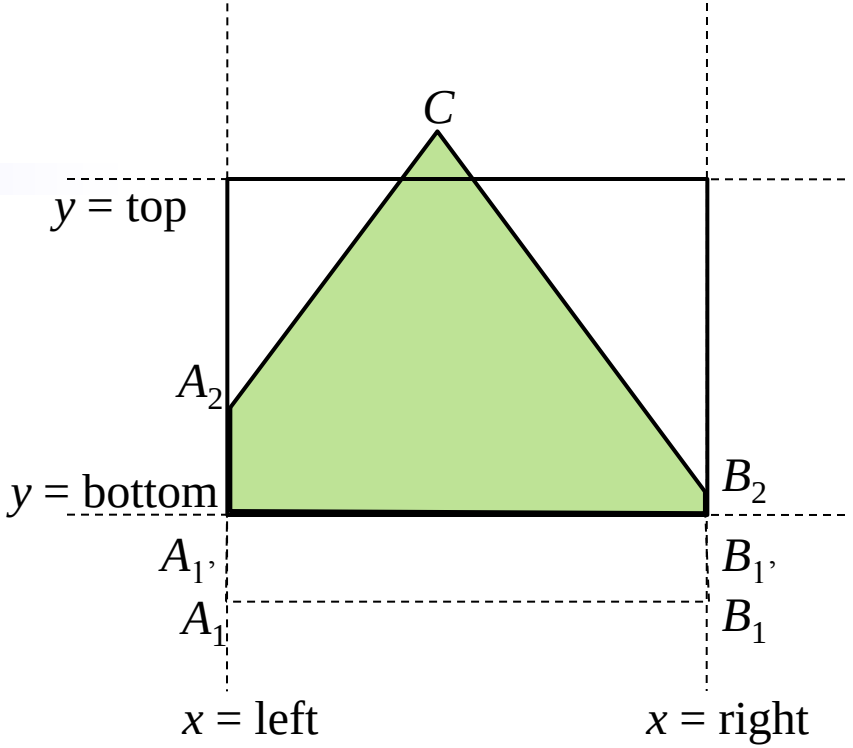
Polygon Clipping

- Sutherland-Hodgman
- Polygon *ABC*
- Clip left: A_1BCA_2
- Clip right: $A_1B_1B_2CA_2$



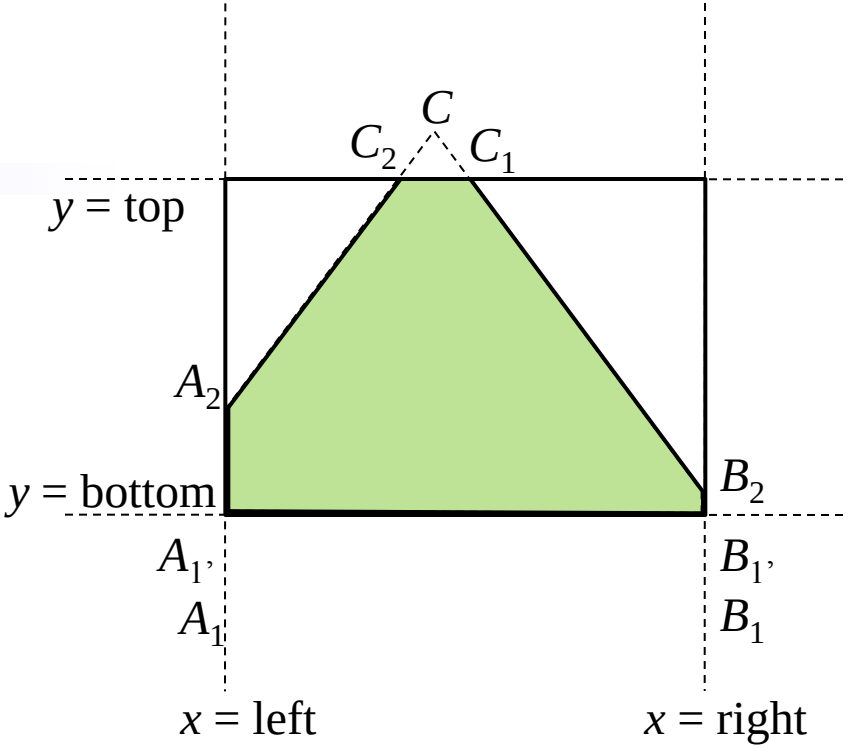
Polygon Clipping

- Sutherland-Hodgman
- Polygon *ABC*
- Clip left: A_1BCA_2
- Clip right: $A_1B_1B_2CA_2$
- Clip bottom: $A_1{}'B_1{}'B_2CA_2$



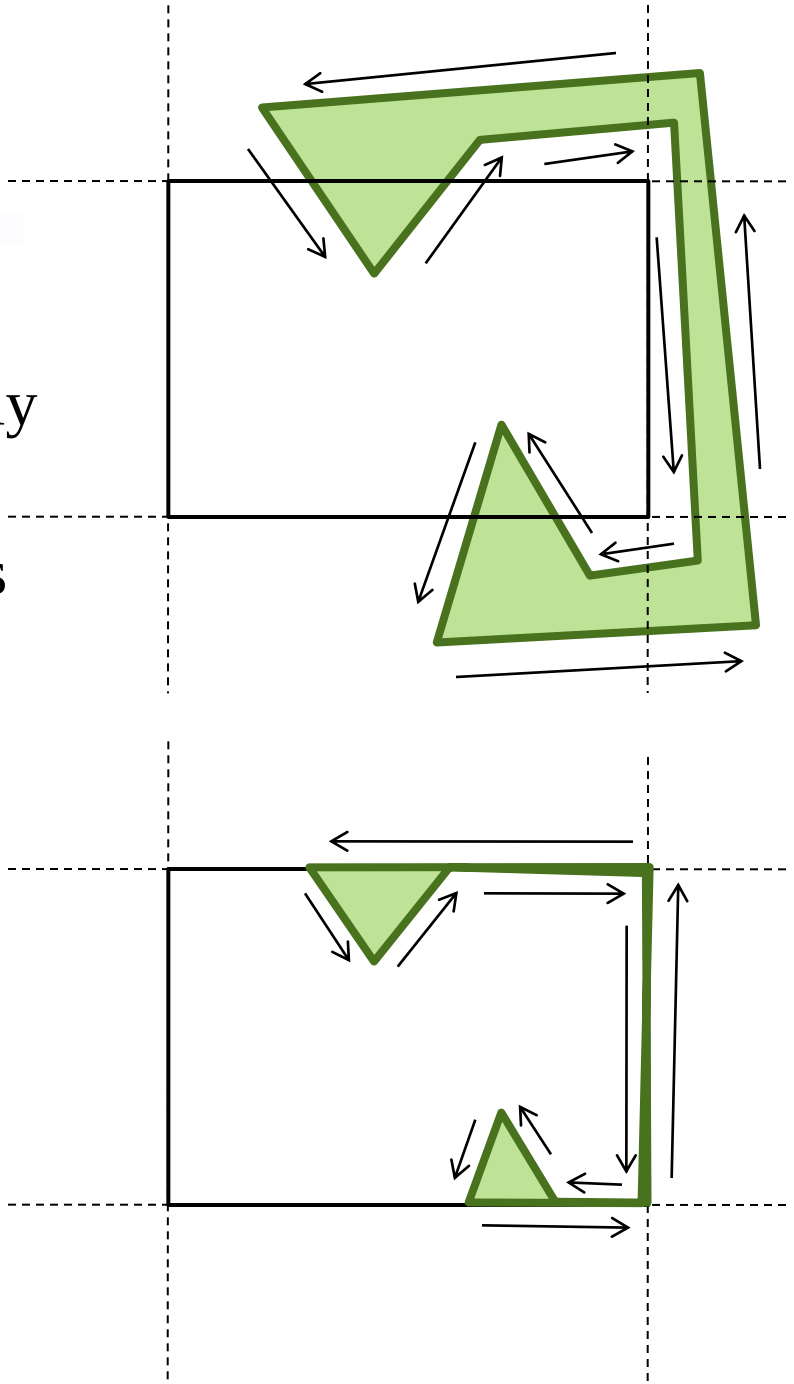
Polygon Clipping

- Sutherland-Hodgman
- Polygon *ABC*
- Clip left: A_1BCA_2
- Clip right: $A_1B_1B_2CA_2$
- Clip bottom: $A_1B_1{'}B_2{'}CA_2$
- Clip top: $A_1B_1{'}B_2{'}C_1C_2A_2$

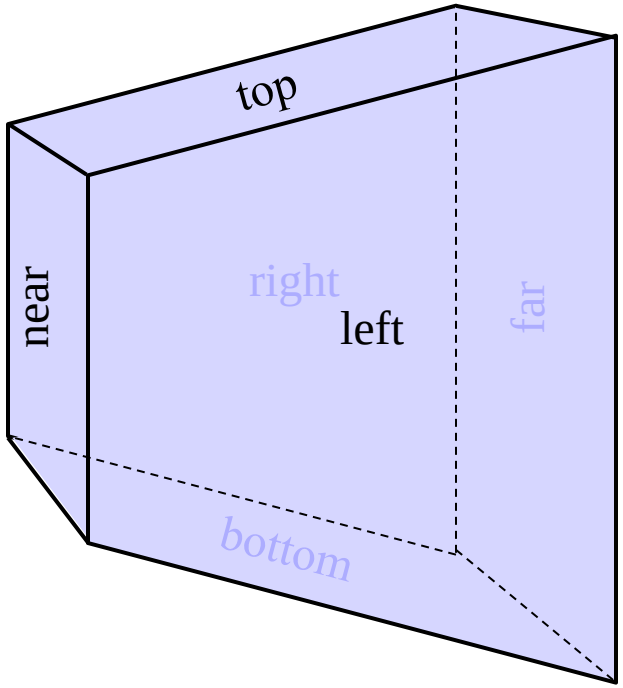


Concave Clipping

- Sutherland-Hodgman
- Clip segments even if they are trivially rejectible (rejectionable?)
- Outputs a single polygon that appears as multiple polygons
- Reversed edges don’t get filled

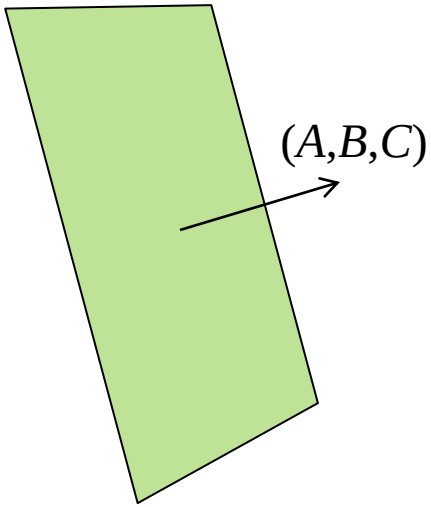


Clipping in 3-D (4-D)



- Need to keep depth (z-coordinate) of geometry for visible surface detection
- Generalize oriented screen edge to oriented clipping plane $C = (A,B,C,D)$
- Then *any* homogeneous point $P = (x,y,z,w)^{\mathrm{T}}$ classified as
  - "on" if $C\,P = 0$
  - "in" if $C\,P > 0$
  - "out" if $C\,P < 0$

$$\begin{bmatrix} A & B & C & D \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} = 0$$



$$Ax + By + Cz + D = 0$$
$$\Updownarrow$$
$$wAx + wBy + wCz + wD = 0$$

Graphics Pipeline