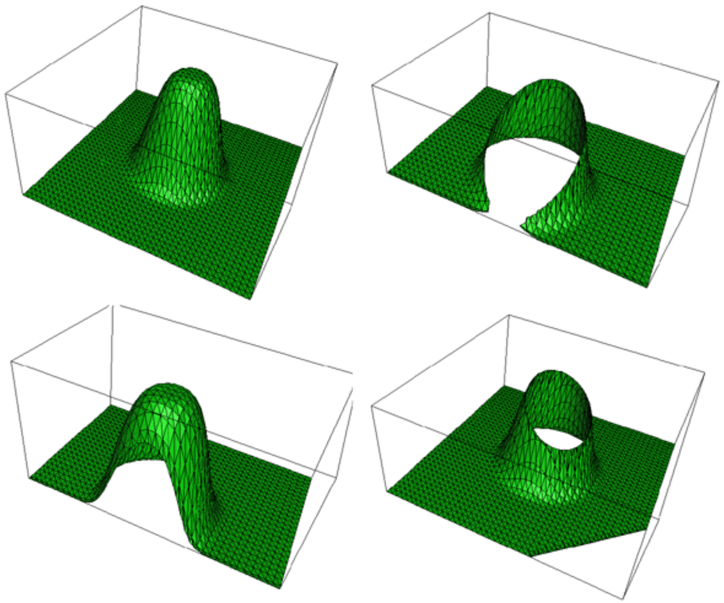
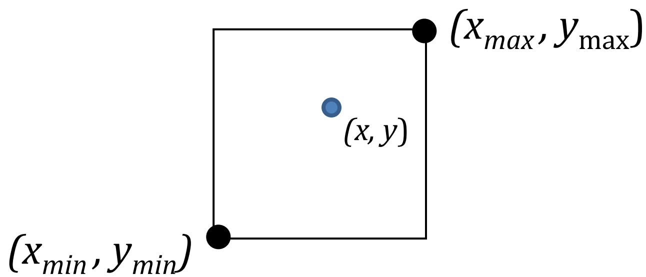


Clipping

Viewports and Algorithms

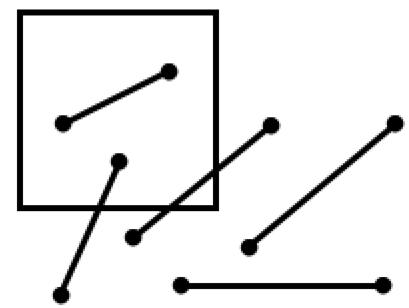
Line Clipping

- Clipping endpoints



  - $x_{min} < x < x_{max}$ **and** $y_{min} < y < y_{max}$ $\Rightarrow$ point inside

- Endpoint analysis for lines:
  - if both endpoints in , do "trivial acceptance"
  - if one endpoint inside, one outside, must clip
  - if both endpoints out, don't know

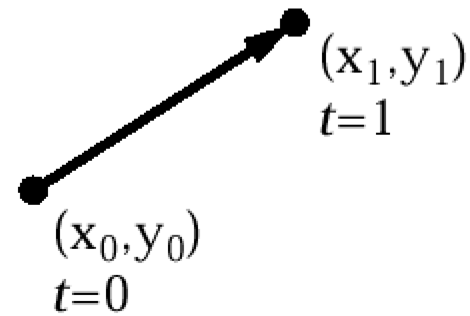


- Brute force clip: solve simultaneous equations using $y = mx + b$ for line and four clip edges
  - slope-intercept formula handles infinite lines only
  - doesn't handle vertical lines

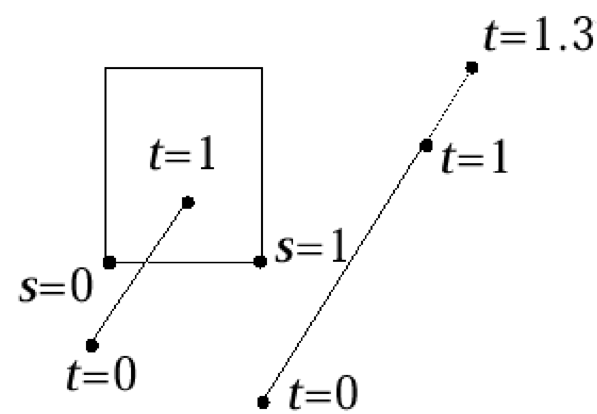
Parametric Line Formulation For Clipping

- Parametric form for line segment

$$X = x_0 + t(x_1 - x_0) \qquad 0 \leq t \leq 1$$
$$Y = y_0 + t(y_1 - y_0)$$
$$P(t) = P_0 + t(P_1 - P_0)$$

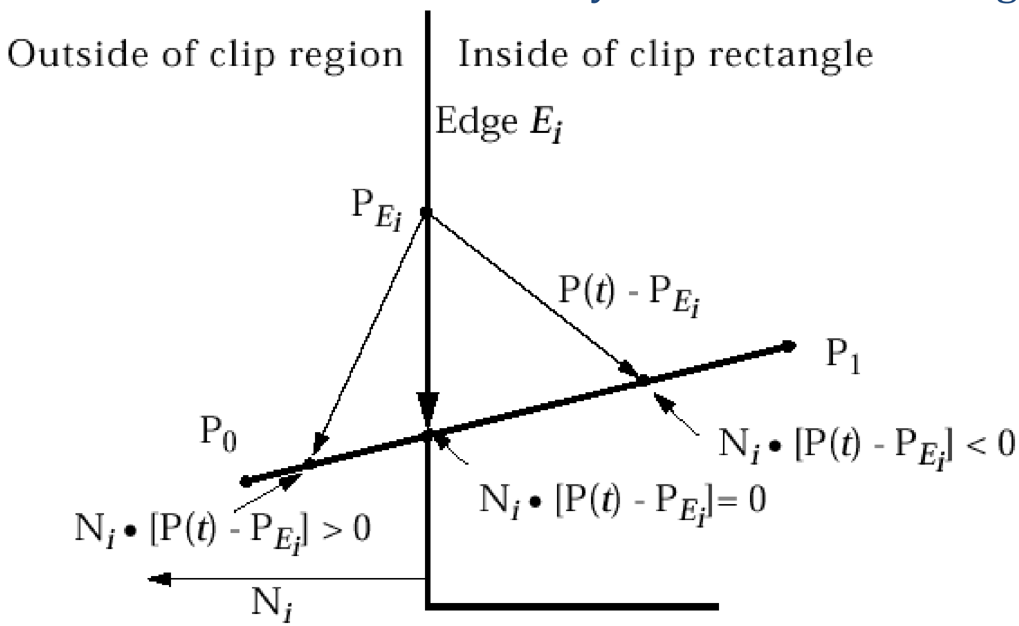


- Line in bounding area if parametric variables s_{edge} and t_{line} both in [0,1] for intersection point between an edge and a line
  - Slow, must intersect lines with all edges



Cyrus-Beck/Liang-Barsky Parametric Line Clipping (1/3)

- Use parametric line formulation: $P(t) = P_0 + (P_1 - P_0)t$
- Determine where line intersects the infinite line formed by each clip rectangle edge
  - solve for t multiple times depending on the number of clip edges crossed
  - decide which of these intersections actually occur on the rectangle



- For P_{E_i}: use any point on edge E_i

Cyrus-Beck/Liang-Barsky Parametric Line Clipping (2/3)

- Now solve for the value of *t* at the intersection of P_0 P_1 with the edge E_i:

$$N_i \bullet [P(t) - P_{E_i}] = 0$$

- First, substitute for P(t):

$$N_i \bullet [P_0 + (P_1 - P_0)t - P_{E_i}] = 0$$

- Next, group terms and distribute dot product:

$$N_i \bullet [P_0 - P_{E_i}] + N_i \bullet [P_1 - P_0]t = 0$$

- Let D be the vector from P_0 to $P_1 = (P_1 - P_0)$, and solve for t:

$$t = \frac{N_i \bullet [P_0 - P_{E_i}]}{-N_i \bullet D}$$

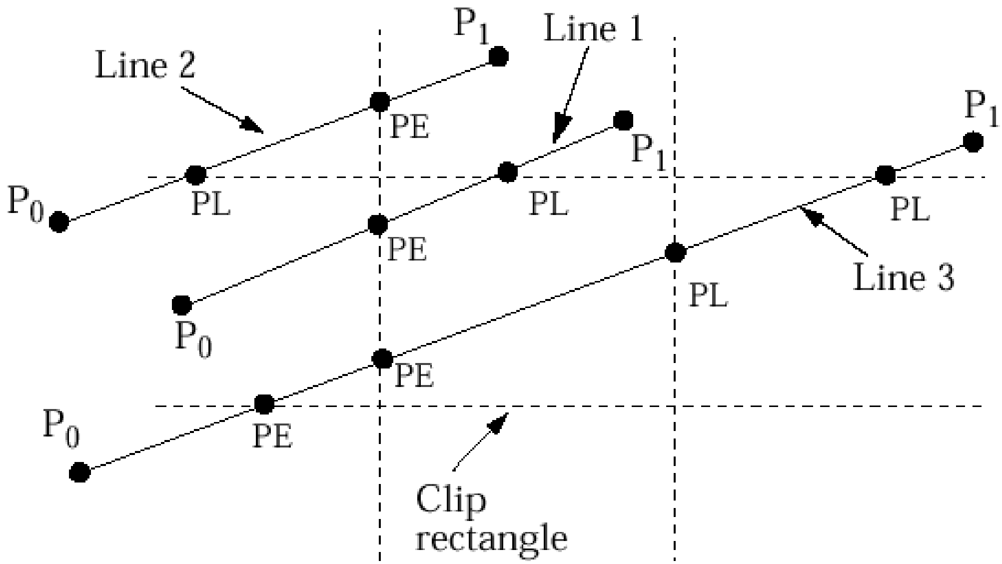
  - note that this gives a valid value of *t* only if the denominator of the expression is nonzero.

- For this to be true, it must be the case that:
  - $N_i \neq 0$ (that is, the normal should not be 0; this could occur only as a mistake)
  - $D \neq 0$ (that is, $P_1 \neq P_0$)
  - $N_i \bullet D \neq 0$ (edge E_i and line D are not parallel; if they are, no intersection).

- The algorithm checks these conditions.

Cyrus-Beck/Liang-Barsky Parametric Line Clipping (3/3)

- Eliminate t's outside [0,1] on the line
- Which remaining t's produce interior intersections?
- Can't just take the innermost t values!



- Move from P_0 to P_1; for a given edge, just before crossing:
  - if $N_i \bullet D < 0 \Rightarrow$ Potentially Entering (PE); if $N_i \bullet D > 0 \Rightarrow$ Potentially Leaving (PL)
- Pick inner PE/PL pair: t_E for P_{PE} with max t, t_L for P_{PL} with min t, and $t_E > 0, t_L < 1$
- If $t_L < t_E$, no intersection

Cyrus-Beck/ Liang-Barsky Line Clipping Algorithm

```
Pre-calculate N_i and select P_E_i for each edge;
for each line segment to be clipped
 if P_1 = P_0 then line is degenerate so clip as a point;
 else
  begin
   t_E = 0; t_L = 1;
   for each candidate intersection with a  clip edge
    if Ni • D ≠ 0 then {Ignore edges parallel to line}
     begin
      calculate t; {of line and clip edge intersection}
      use sign of N_i • D to categorize as PE or PL;
      if PE then t_E = max(t_E,t);
      if PL then t_L = min(t_L,t);
     end
   if t_E > t_L then return  nil
   else return P(t_E) and P(t_L) as true clip intersections
 end
```

Parametric Line Clipping for Upright Clip Rectangle (1/2)

- $D = P_1 - P_0 = (x_1 - x_0, y_1 - y_0)$
- Leave P_{E_i} as an arbitrary point on clip edge: it's a free variable and drops out

Calculations for Parametric Line Clipping Algorithm

Clip Edge$_i$	Normal N_i	P_{E_i}	P_0-P_{E_i}	$t = \frac{N_i \bullet (P_0 - P_{E_i})}{-N_i \bullet D}$
left: $x = x_{min}$	(-1,0)	(x_{min}, y)	$(x_0 - x_{min}, y_0 - y)$	$\frac{-(x_0 - x_{min})}{(x_1 - x_0)}$
right: $x = x_{max}$	(1,0)	(x_{max}, y)	$(x_0 - x_{max}, y_0 - y)$	$\frac{-(x_0 - x_{max})}{(x_1 - x_0)}$
bottom: $y = y_{min}$	(0,-1)	(x, y_{min})	$(x_0 - x, y_0 - y_{min})$	$\frac{-(y_0 - y_{min})}{(y_1 - y_0)}$
top: $y = y_{max}$	(0,1)	(x, y_{max})	$(x_0 - x, y_0 - y_{max})$	$\frac{-(y_0 - y_{max})}{(y_1 - y_0)}$

Parametric Line Clipping for Upright Clip Rectangle (2/2)

- Examine *t*:
  - numerator is just the directed distance to an edge; sign corresponds to OC
  - denominator is just the horizontal or vertical projection of the line, ***dx*** or ***dy***; sign determines PE or PL for a given edge
  - ratio is constant of proportionality: “how far over” from P_0 to P_1 intersection is relative to ***dx*** or ***dy***