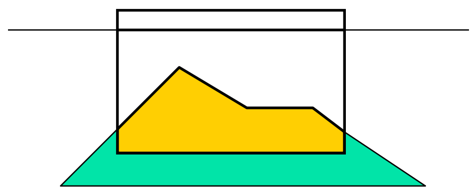
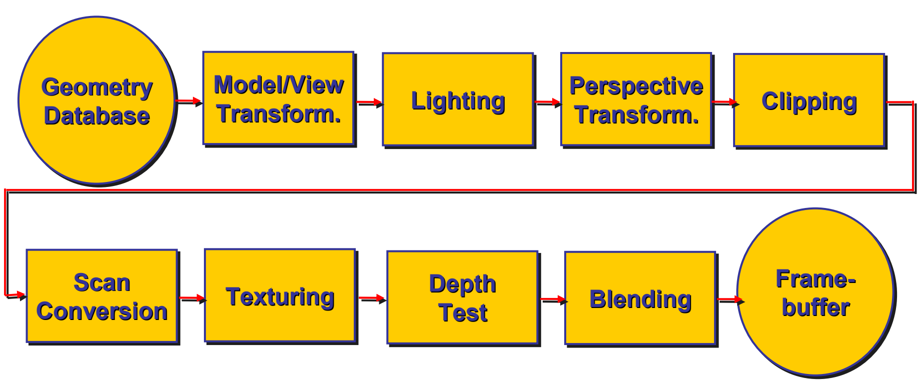
University of
British Columbia

Chapter 6

Clipping



The Rendering Pipeline



University of
British Columbia

Line/Polygon Clipping

Problem:
Given a set of 2*D* lines/polygons and a window, clip the lines/polygons to their regions that are *inside* the window.

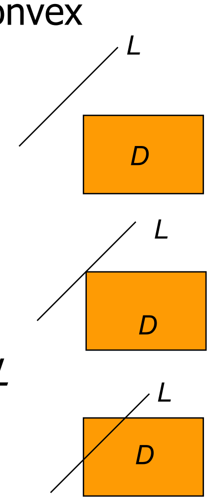
Objectives

- Efficiency
- Display in portion of screen (rectangular window)

University of British Columbia

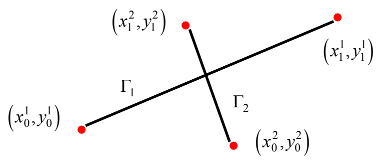
Analytic Solution

- *Intersection* of convex regions is convex
  - Why?
- *L* & *D* are *convex* - intersection is convex
  - single connected segment of *L*
- **Question:** Can boundary of two convex shapes intersect more than twice?
- Clipping - compute intersection of *L* with four boundary segments of window *D*



University of British Columbia

Line-Line Intersection



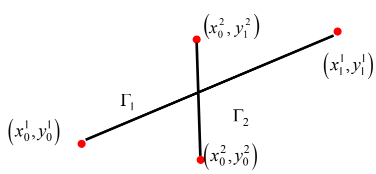
$$G_1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} \quad t \in [0,1] \qquad G_2 = \begin{cases} x^2(r) = x_0^2 + (x_1^2 - x_0^2)r \\ y^2(r) = y_0^2 + (y_1^2 - y_0^2)r \end{cases} \quad r \in [0,1]$$

Intersection: *x* & *y* values equal in both representations - two linear equations in two unknowns (*r*,*t*)

$$x_0^1 + (x_1^1 - x_0^1)t = x_0^2 + (x_1^2 - x_0^2)r$$
$$y_0^1 + (y_1^1 - y_0^1)t = y_0^2 + (y_1^2 - y_0^2)r$$

University of British Columbia

Intersection with vertical/horizontal lines



$$G_1 = \begin{cases} x^1(t) = x_0^1 + (x_1^1 - x_0^1)t \\ y^1(t) = y_0^1 + (y_1^1 - y_0^1)t \end{cases} t \in [0,1] \; G_2 = \begin{cases} x^2(r) = x_0^2 \\ y^2(r) = y_0^2 + (y_1^2 - y_0^2)r \end{cases} r \in [0,1]$$

Intersection: *x* & *y* values equal in both representations - two linear equations in two unknowns (*r*,*t*)

$$x_0^1 + (x_1^1 - x_0^1)t = x_0^2$$
$$t = \frac{x_0^2 - x_0^1}{x_1^1 - x_0^1}$$
$$y_0^1 + (y_1^1 - y_0^1)t = y_0^2 + (y_1^2 - y_0^2)r$$

University of British Columbia

Cohen-Sutherland Algorithm

Purpose:
Fast treatment of line segments that are trivially inside/outside window.

0101	0100	0110
0001	0000	0010
1001	1000	1010

P = (*x*,*y*) - point to be classified against window *D*

Idea: Assign to P a binary code consisting of a bit for each edge of *D*, using lookup table:

bit	1	0
1	$y < y_{\min}$	$y \ge y_{\min}$
2	$y > y_{\max}$	$y \le y_{\max}$
3	$x > x_{\max}$	$x \le x_{\max}$
4	$x < x_{\min}$	$x \ge x_{\min}$

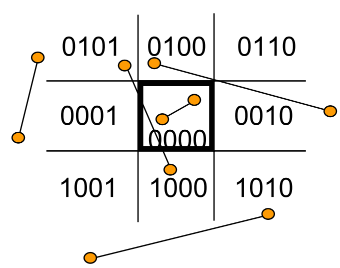
University of British Columbia

Cohen-Sutherland Algorithm (cont'd)

Given *L* from (x_0, y_0) to (x_1, y_1)
& rectangle D.

If bitwise **and** of the codes of (x_0, y_0) and (x_1, y_1) is not zero, or the bitwise **or** is zero,

then *L* can be trivially handled (it is either totally outside or totally inside D).

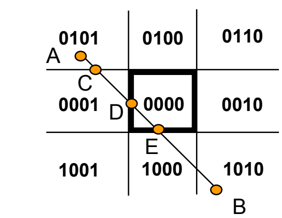


Why?

University of British Columbia

Cohen-Sutherland Algorithm (cont'd)

C-S-Clip($P_0 = (x_0, y_0), P_1 = (x_1, y_1), x_{\min}, x_{\max}, y_{\min}, y_{\max}$)
$C_0 \Leftarrow$ code (P_0); $C_1 \Leftarrow$ code (P_1);
if ((C_0 and C_1) != 0) then return;
if ((C_0 or C_1) == 0) then draw(P_0, P_1);
else if (OutsideWindow(P_0)) then
begin
Edge $\Leftarrow$ Window boundary of leftmost non-zero bit of C_0;
$P_2 \Leftarrow \overline{P_0, P_1} \cap Edge$;
C-S-Clip($P_2, P_1, x_{\min}, x_{\max}, y_{\min}, y_{\max}$);
end
else
Edge $\Leftarrow$ Window boundary of leftmost non-zero bit of C_1;
$P_2 \Leftarrow \overline{P_0, P_1} \cap Edge$;
C-S-Clip($P_0, P_2, x_{\min}, x_{\max}, y_{\min}, y_{\max}$);
end



AB → CB → DB → DE

bit	1	0
1	$y < y_{\min}$	$y \geq y_{\min}$
2	$y > y_{\max}$	$y \leq y_{\max}$
3	$x > x_{\max}$	$x \leq x_{\max}$
4	$x < x_{\min}$	$x \geq x_{\min}$

University of British Columbia

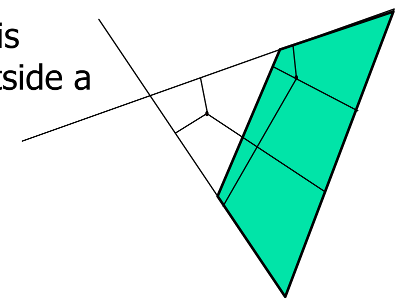
Triangle Clipping

- How does intersection of rectangle & triangle looks like?
- How to expand clipping to triangles?
  - Hint: it is convex

University of British Columbia

Other Geometric Problems

- Questions: How can these ideas be used to design an algorithm for checking if:
  - a point is inside a (convex) polygon ?
  - a (convex) polygon is inside/intersects/outside a (convex) polygon ?



University of British Columbia