

CLIPPING ON A RASTER DISPLAY

- Clipping:
 - Remove points outside a region of interest.
 - Discard (parts of) primitives outside of window
- Point clipping:
 - Remove points outside window.
 - A point is either entirely inside the region or not.
- Line clipping:
 - Remove portion of line segment outside window.
- Polygon clipping:
 - Remove portion of polygon outside window

2/8/2000

CS 4/57101 Lecture 7

1

Approaches to Clipping

- Analytically
 - best for floating-point packages
 - scan convert the clipped primitives
- During scan conversion - *scissoring*
 - good for filled or thick primitives
 - only extrema need be clipped
 - good if primitives not much larger than clip rectangle
 - not too many extra pixels generated
- As part of copy-pixel operation
 - useful when likely to have window pan over large canvas

2/8/2000

CS 4/57101 Lecture 7

2

Role of Analytical Clipping

- Floating-point packages
 - clip then scan convert
- Integer packages
 - pre-clip and scan convert or
 - clip during scan conversion
- Common strategy
 - clip lines and polygons analytically
 - clip other primitives during scan conversion

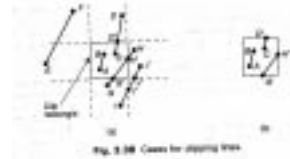
2/8/2000

CS 4/57101 Lecture 7

3

Clipping Lines against Rectangles

- Lines are always clipped to at most one line segment
- Lines on rectangle border consider inside (displayed)



Clipping Endpoints

- inside $x_{\min} \leq x \leq x_{\max} \wedge y_{\min} \leq y \leq y_{\max}$

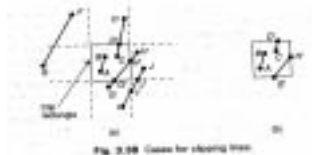
2/8/2000

CS 4/57101 Lecture 7

4

Clipping Rules

- Consider only end-points
- If both inside rectangle - *trivially accepted*
- One inside, one outside - need to compute intersection
- Both outside - may or may not intersect with clip rectangle



2/8/2000

CS 4/57101 Lecture 7

5

Computing Intersections

- Brute force
 - intersect each lines with 4 edges of clip rectangle
 - need to solve for 2 intersection points per edge
 - each involves solving 2 simultaneous equations
- Could use slope-intercept formula
 - really for infinite lines - does not handle vertical
- Use parametric form instead
 - $x = x_0 + t(x_1 - x_0)$, $y = y_0 + t(y_1 - y_0)$, t in $[0,1]$
- solve for t_{edge} and t_{line}
 - if both lie in $[0,1]$ then real intersection
- Still involves much calculation

2/8/2000

CS 4/57101 Lecture 7

6

Cohen-Sutherland Algorithm

- Performs tests to avoid calculations
 - check for *trivial acceptance*
 - do *region tests* e.g. if both endpoints lie to left of rectangle *trivially rejected* etc.
 - if neither divide line segment in two at clip edge
 - trivially reject one
 - continue comparing against each clip edge
- Efficient if clip rectangle large (almost all inside) or small (almost all outside) *picking window*

2/8/2000

CS 4/57101 Lecture 7

7

Outcodes

- Divide plane of clip rectangle into 9 regions



- Assign 4 bit code to each - each bit (1 or 0) indicates position wrt outside half-plane of clip edge
- Can calculate efficiently as sign bit of $(y_{max}-y)$, $(y-y_{min})$, $(x_{max}-x)$, $(x-x_{min})$

2/8/2000

CS 4/57101 Lecture 7

8

Cohen-Sutherland Procedures

- Both outcodes 0000, line inside, *trivially accept*
- Both in outside plane of same edge, *trivially reject*
 - if logical **and** of outcode not 0 *trivially reject*
- If neither, subdivide at edge
 - throw away outside segment
- Convention : go top to bottom, right to left
- Outcode property: if bit set, line crosses edge
 - makes it easy to divide segment at edge

2/8/2000

CS 4/57101 Lecture 7

9

Cohen-Sutherland Procedures

- Both outcodes 0000, line inside, *trivially accept*
- Both in outside plane of same edge, *trivially reject*
 - if logical **and** of outcode not 0 *trivially reject*
- If neither, subdivide at edge
 - throw away outside segment
- Convention : go top to bottom, right to left
- Outcode property: if bit set, line crosses edge
 - makes it easy to divide segment at edge

2/8/2000

CS 4/57101 Lecture 7

10

Cohen-Sutherland Procedures

- Compute outcodes of endpoints
- Check for trivial acceptance or rejection
- if neither
 - find outside endpoint
 - test outcode to find edge crossed, compute intersection
 - clip by replacing outside point by intersection
 - compute outcode of new endpoint



2/8/2000

CS 4/57101 Lecture 7

11

Cohen-Sutherland Procedure

- Efficiency
 - use bitwise arithmetic for outcodes
 - do not recalculate slopes
- Not most efficient
 - sometimes needless clipping
 - Nicholl, Lee, Nicholl avoids this
- Advantage of Cohen-Sutherland
 - extension to 3D orthographic view volume straightforward

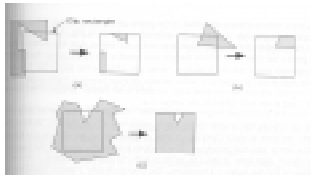
2/8/2000

CS 4/57101 Lecture 7

12

Clipping Polygons against Rectangles

- Clipping rectangle gives (at most) rectangle
- Clipping convex polygon gives convex polygon
- Clipping concave polygon may produce more than one concave polygon
- Clipping a circle or ellipse may result in up to 4 arcs

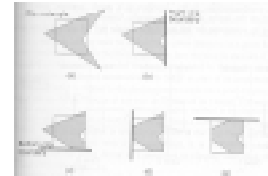


2/8/2000

CS 4/57101 Lecture 7

13

Sutherland-Hodgman Algorithm



2/8/2000

CS 4/57101 Lecture 7

14

Improvements and Generalizations

2/8/2000

CS 4/57101 Lecture 7

15