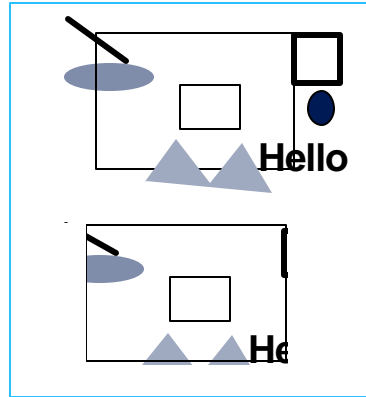


Raster graphics alg's for drawing 2D primitives

- Points of view
 - Application programmer's
 - Package implementor's
 - Scan-converted clipped primitives



Overview ...

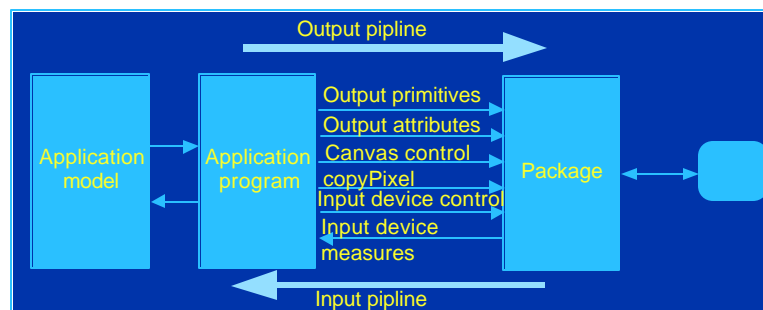
- Implications of display-system architecture ...
- Output pipeline ...

Implications of disp-sys architecture ...

- Output & input pipelines ...
- Displays with frame buffers & display controllers ...
- Displays with frame buffers only ...
- Hardcopy devices ...

Output & input pipelines

- Output
- Input



Output



- **Primitive-generation function: What to generate**
- **Attribute functions: How to generate**
- **copyPixel: How to modify image**
- **Canvas-control functions: Where**



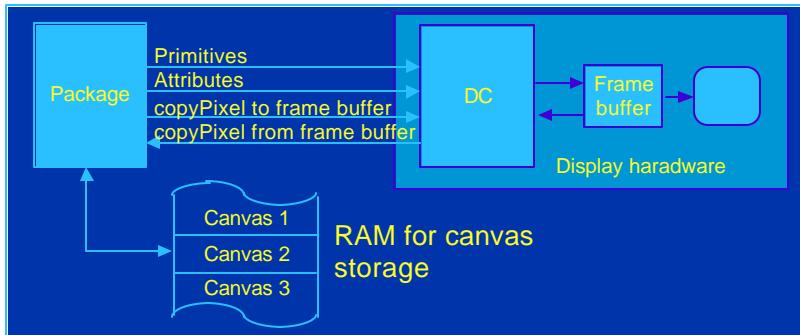
Input



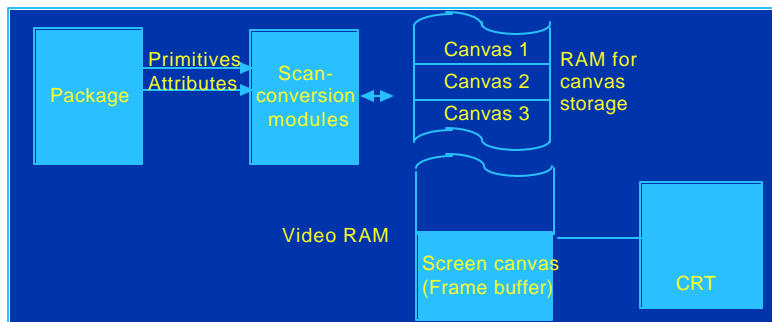
- **User interaction (display) --> measure values**
- **==> modify model or image**



Displays with frame buffers & display controllers ...



Displays with frame buffers only ...



Hardcopy devices ...

- One scan line at a time
 - Package: generate complete bitmap
 - Scan out 1 line at a time
- Entire frame (page)
- Raster Image Processor (RIP): built-in scan-conversion HW
- PostScript engine: interpret program
 - Device independent
 - ==> Primitives + attributes



Output Pipeline ...

- Clip primitives then scan-convert
 - ==> Fewer to scan-convert
- Scissoring: Scan-convert then clip
 - Write only visible pixels
- Temp. canvas, copy only clip rectangle
 - Wastes time & space
 - Easy to implement
 - Often for text
- Incremental methods: minimize number of calculations

Scan Converting Primitives

- primitives

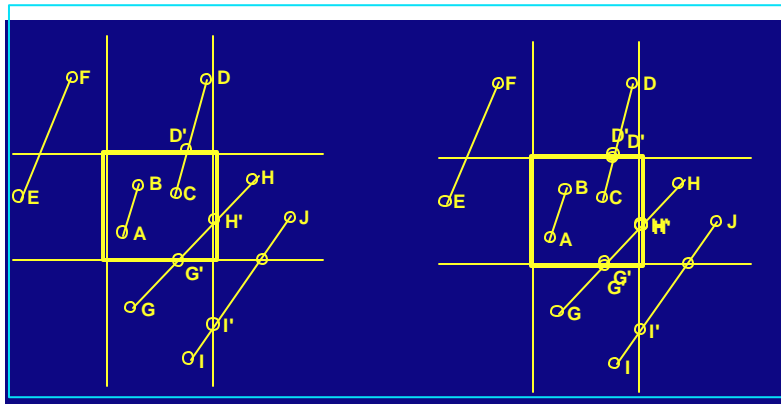
Clipping

- Clipping endpoints ...
- Cases for clipping lines ...
- Solving simultaneous equations ...
- Cohen-Sutherland Algorithm ...
- Parametric Line-Clipping Algorithm ...

Clipping endpoints

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

Cases for clipping lines ...



Solving Simultaneous Equations ...

- Both endpoints inside ==> trivially accept
- One inside, one outside ==> find intersection point
- Both outside: may / may not intersect
- Brute force: line equation + clip rectangle edges equation



Cohen-Sutherland Algorithm ...

1. Check endpoints: trivially accept?
 2. Region checks: trivially reject?
 3. Divide into 2 segments s.t. one can be trivially rejected
 - Iterative clip
 - Trivially accept? / reject?
- Example ...

1001	1000	1010	
0001	0000	0010	
0101	0100	0110	
above	below	right	left

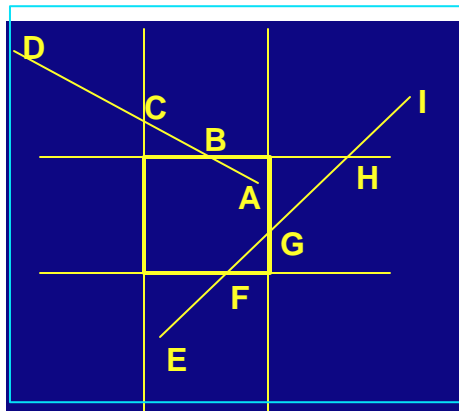


Cohen-Sutherland (cont.)

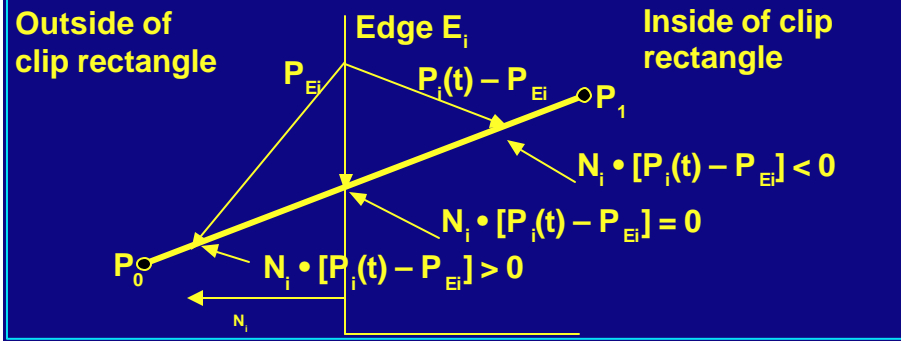
1001	1000	1010	
0001	0000	0010	
0101	0100	0110	
above	below	right	left



Cohen-Sutherland example



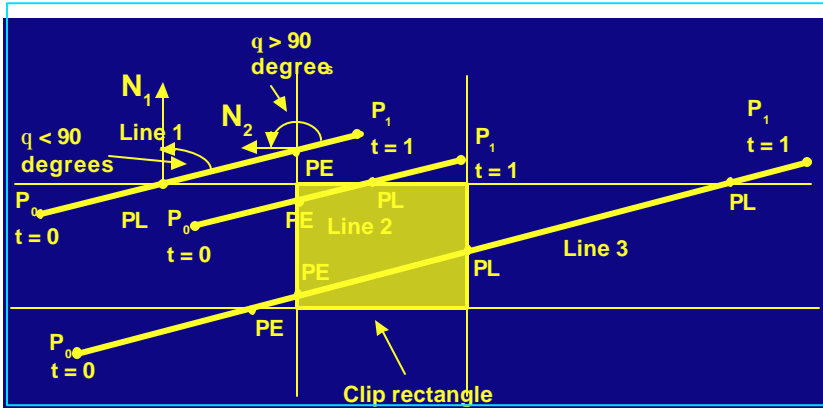
Parametric Line-Clipping algorithm



- Example ...



Parametric Line-Clipping example



Clipping circles and ellipses

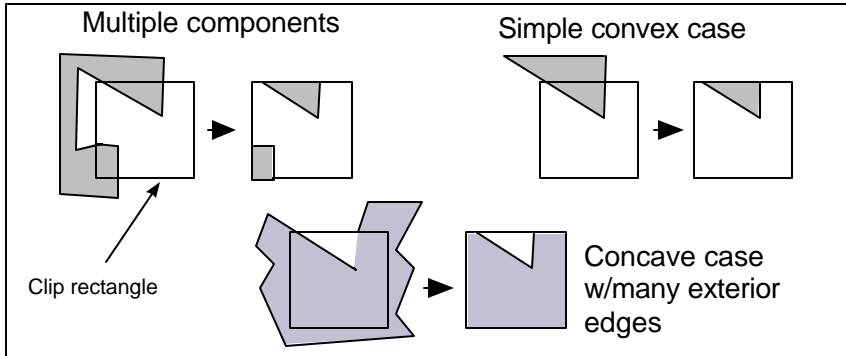
- Accept / reject against square extent
- If intersects, divide into quadrants, repeat accept / reject
- Divide into octants if necessary
- Compute intersection of edge-circle analytically
 - Simultaneous equations
- Scan convert resulting arcs
- Fast scan conversion ==> scissoring faster
- Filled: clip spans then fill

© Copyright 2000 Haim Levkowitz

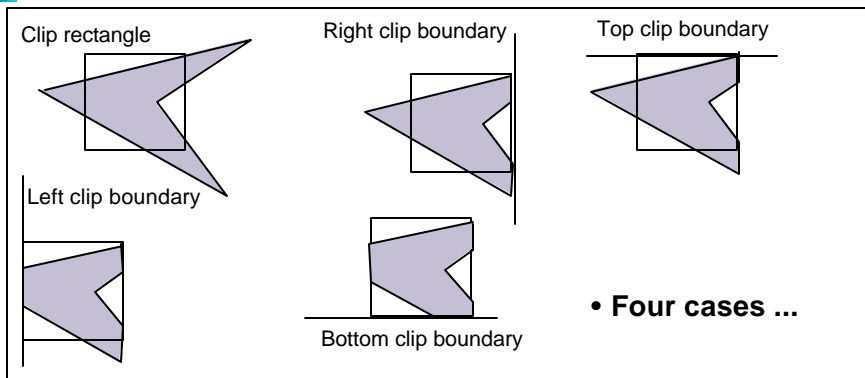
Clipping Polygons

- General ...
- Sutherland-Hodgman Algorithm ...
 - Four cases ...

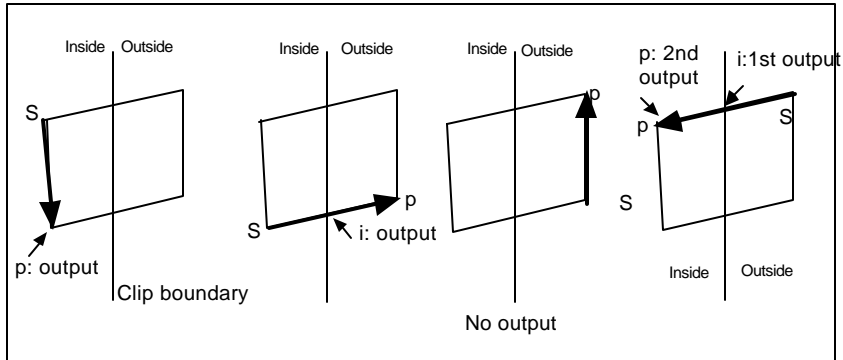
General



Sutherland-Hodgman Algorithm



Four cases



Antialiasing

- Increasing resolution ...
- Area sampling ...
 - Weighted
 - Unweighted
- Other

Increasing Resolution

- **Best, but**
 - **Limitations**
- **Jaggies / staircasing**
- **Aliasing / antialiasing**



Area sampling

- **Area represented by pixel**
- **Unweighted area sampling**
 - **All pixels equal**
- **Weighted area sampling**
 - **Some contribute more than others**
 - **Based on weight function**

