

Drawing in 2D

**CSCI 4229/5229
Computer Graphics
Fall 2017**

Coordinate Systems

- Cartesian coordinates
 - Most commonly used
 - Left or right handed
 - 2D is a trivial case in 3D
- Polar coordinates
 - Convenient in some instances
- Curvilinear Coordinates
 - Specialized applications

2D Cartesian Coordinate Systems

- World Coordinates
 - $Xmin - Xmax \times Ymin - Ymax$
- Normalized Device Coordinates
 - $0-1 \times 0-1$ or $0-1 \times 0-r$
 - may be isometric
 - Viewport $Umin - Umax \times Vmin - Vmax$
- Device coordinates
 - pixels, plotter increments
 - origin may be top-left

Transformations

- World to Normalized Device Coordinates

$$u = (x - X_{min}) / (X_{max} - X_{min}) * (U_{max} - U_{min}) + U_{min}$$

$$v = (y - Y_{min}) / (Y_{max} - Y_{min}) * (V_{max} - V_{min}) + V_{min}$$

- Normalized Device to World Coordinates

$$x = (u - U_{min}) / (U_{max} - U_{min}) * (X_{max} - X_{min}) + X_{min}$$

$$y = (v - V_{min}) / (V_{max} - V_{min}) * (Y_{max} - Y_{min}) + Y_{min}$$

- (x, y) may be outside $(X_{min} - X_{max}, Y_{min} - Y_{max})$

Vector Lines

- Line from (x_0, y_0) to (x_1, y_1)
- Explicit
 - $y = (x-x_0)*(y_1-y_0)/(x_1-x_0) + y_0$
 - $x = (y-y_0)*(x_1-x_0)/(y_1-y_0) + x_0$
- Parameteric
 - $x = (1-f)x_0 + f x_1$
 - $y = (1-f)y_0 + f y_1$
 - $f = 0 \Rightarrow (x_0, y_0); \quad f = 1 \Rightarrow (x_1, y_1)$

Vector Clipping

- Cohen-Sutherland Line Clipping
 - Determine region of start and end
 - Accept, reject or clip
- Parametric Line-Clipping Algorithm
 - Calculate parameter t
 - $0 < t < 1$ requires clipping
- Sutherland-Hodgman Polygon Clipping
 - Clips edges of polygon
 - Successive clips to half planes
- ***OpenGL does this for you***

Cohen-Sutherland Line Clipping

- Set bits to identify outside zones
- Trivial accept or reject
- Clip non-trivial cases
- Accept or reject

Parametric Line Clipping

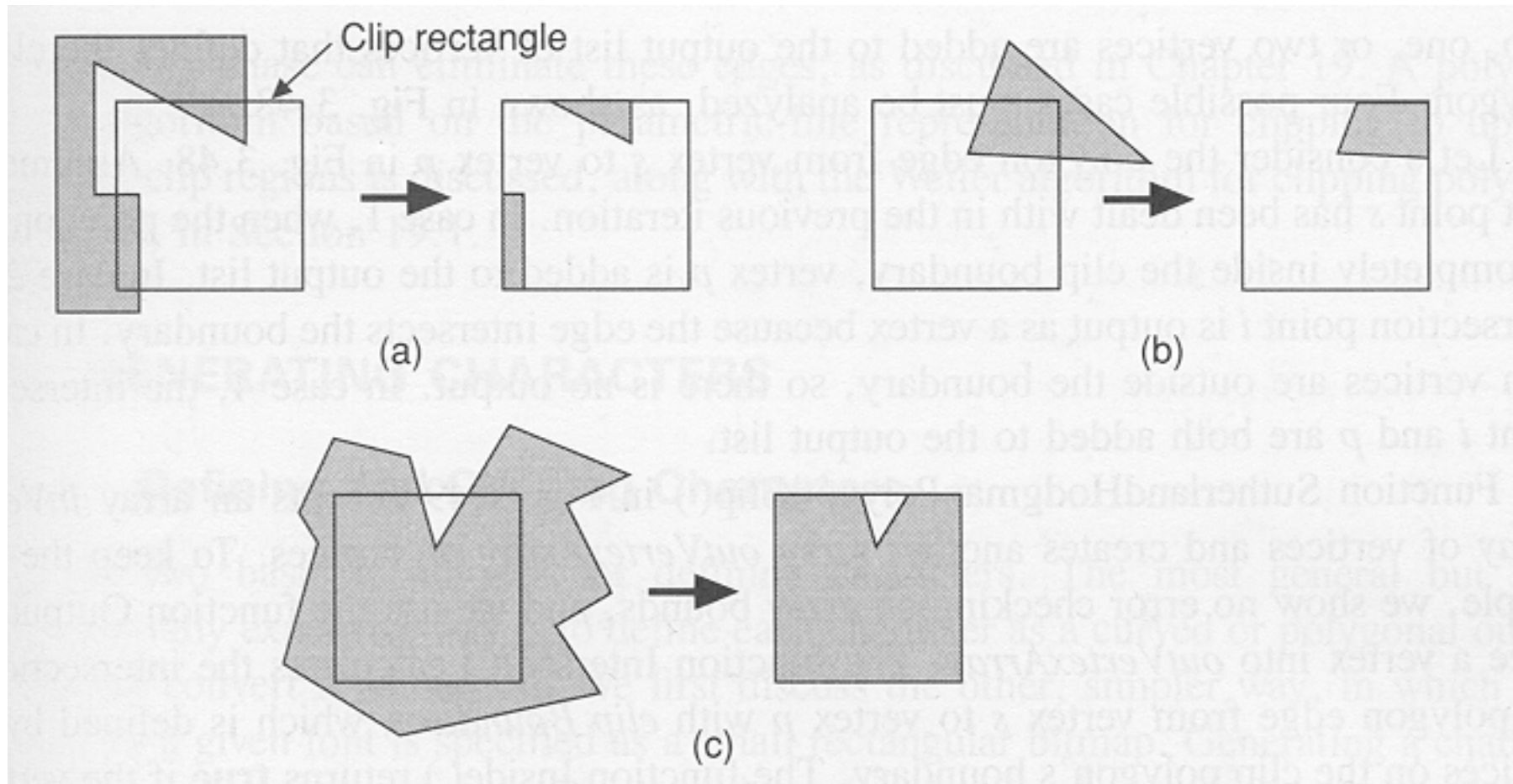
- Cohen-Sutherland may require up to 4 clips
- Parametric algorithm more efficient
 - Original Cyrus-Beck
 - Improved by Liang-Barsky
- Readily extends to 3D and irregular windows
- Basic equation for line from P_0 to P_1

$$t = (N \cdot [P_0 - P_E]) / (N \cdot [P_0 - P_1])$$

N is the outside normal

P_E is on the edge

Polygon Clipping Challenges



Polygon Clipping Algorithm

