Intersections and Clipping

LESSON 7

Computer Graphics 1

Intersections

line-line, point-polygon, line-curve, line-plane, line-polygon, line-surface

Intersections

- Frequent operation
  - Ambition to accelerate computation
- Bounding volumes for complex objects
  - Box, sphere, ellipsoid, cylinder,…
- Partial scene organization
  - Octant tree, BSP tree, …

Line-Line Intersection

$$L_1(t) = A + \vec{u}t; t \in \langle 0,1 \rangle \qquad L_2(t') = B + \vec{v}t'; t' \in \langle 0,1 \rangle$$

- 1. Treat special cases
  - Parallel lines
    - Intersect only if they are collinear
- 2. Nonparallel lines

$$A + \vec{u}t = B + \vec{v}t'$$

$$\vec{u}t - \vec{v}t' = (B - A)$$

$$\vec{u}\vec{v}^{\perp}t - \vec{v}\vec{v}^{\perp}t' = (B - A)\vec{v}^{\perp}$$

$$t = \frac{(B - A)\vec{v}^{\perp}}{\vec{u}\vec{v}^{\perp}}$$

$t \in \langle 0,1 \rangle \Rightarrow$ Intersection lies on line L_1

Line-Line Intersection (2)

$$t' = \frac{(B-A)\vec{u}^{\perp}}{-\vec{v}\vec{u}^{\perp}} = \frac{(B-A)\vec{u}^{\perp}}{\vec{u}\vec{v}^{\perp}}$$

$$\vec{v}\vec{u}^{\perp} = (v_1, v_2)(-u_2, u_1) = -u_2 v_1 + u_1 v_2 = (u_1, u_2)(v_2, -v_1) =$$
$$= -(u_1, u_2)(-v_2, v_1) = -\vec{u}\vec{v}^{\perp}$$

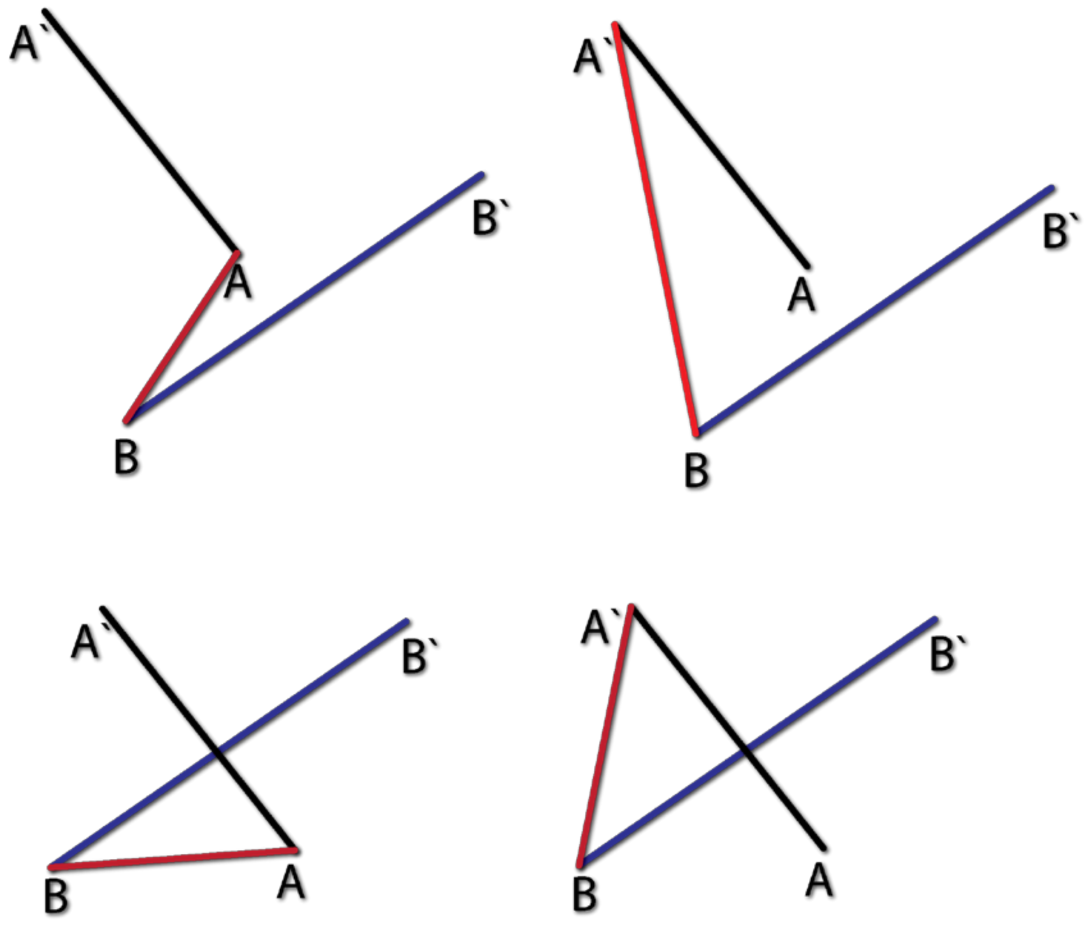
$$t = \frac{(B-A)\vec{v}^{\perp}}{\vec{u}\vec{v}^{\perp}}$$

$$t' = \frac{(B-A)\vec{u}^{\perp}}{\vec{u}\vec{v}^{\perp}}$$

Intersection exist if:

$$t \in \langle 0,1 \rangle \wedge t' \in \langle 0,1 \rangle$$

Detecting Line-Line Intersection



Comparing basis orientation

Cross product

- 2D->3D

$$\vec{u} = (u_1, u_2) \rightarrow (u_1, u_2, 0)$$

$$\vec{v} = (v_1, v_2) \rightarrow (v_1, v_2, 0)$$

$$\vec{u} \times \vec{v} = \left(\begin{vmatrix} u_2 & 0 \\ v_2 & 0 \end{vmatrix}, \begin{vmatrix} 0 & u_1 \\ 0 & v_1 \end{vmatrix}, \begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix} \right) = \left(0, 0, \begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix} \right)$$

$$|\vec{u}, \vec{v}| = \begin{vmatrix} u_1 & u_2 \\ v_1 & v_2 \end{vmatrix}$$

Basis Orientation

- Compare basis orientation

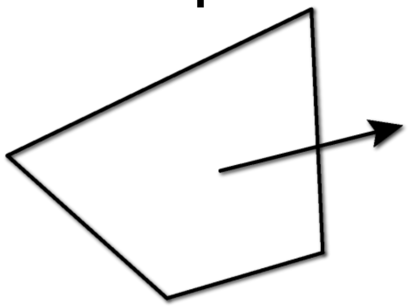
$$\operatorname{sgn}\left|B' - B, A - B\right| = \operatorname{sgn}\left|B' - B, A' - B\right|$$

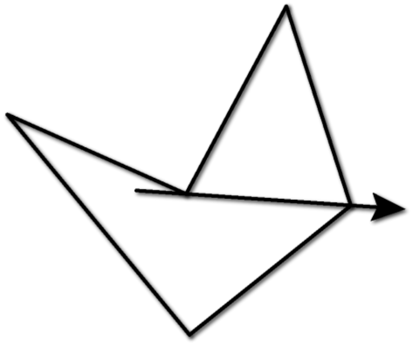
- Basis have the same orientation
  - A and A` are in the same half plane from BB`
  - Intersection does not exist

$$\left.\begin{aligned}\operatorname{sgn}\left|B' - B, A - B\right| \neq \operatorname{sgn}\left|B' - B, A' - B\right| \\ \operatorname{sgn}\left|A' - A, B - A\right| \neq \operatorname{sgn}\left|A' - A, B' - A\right|\end{aligned}\right\} \Rightarrow \text{intersection exists}$$

Point-Polygon Intersection

- Test if point lies inside the polygon
- Sum of oriented angles
  - If the sum is 0 point lies outside
- Count intersections of a Ray from the point P with the polygon
  - #even – point is outside
  - #odd – point lies inside
  - Treat special cases

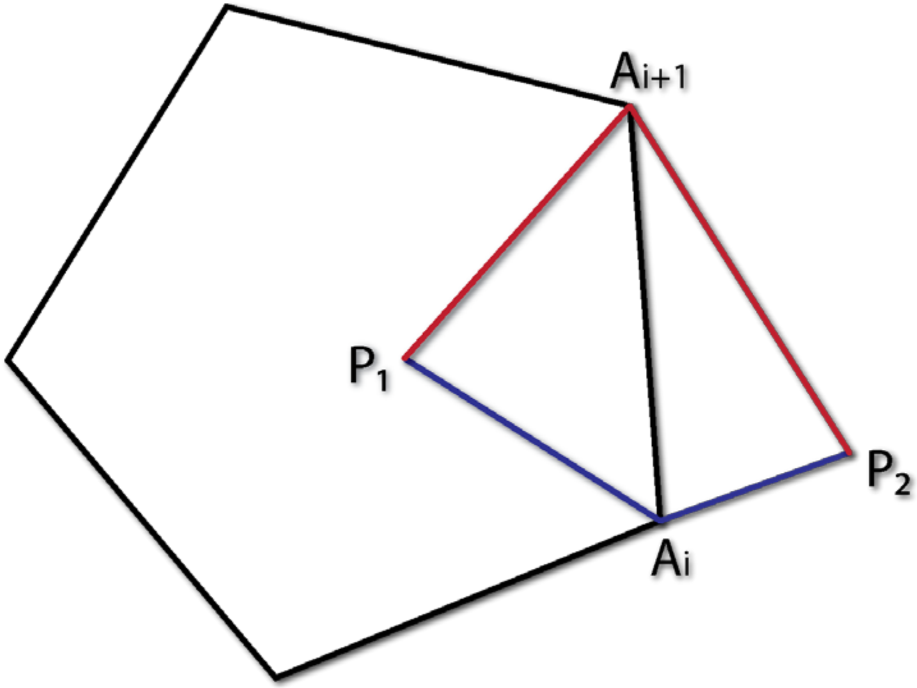




Convex Polygon

- Point lies inside if the basis (A_i-P, A_{i+1}-P) is positively oriented

$$\left| A_i - P, A_{i+1} - P \right| \geq 0$$



Line-Curve Intersection

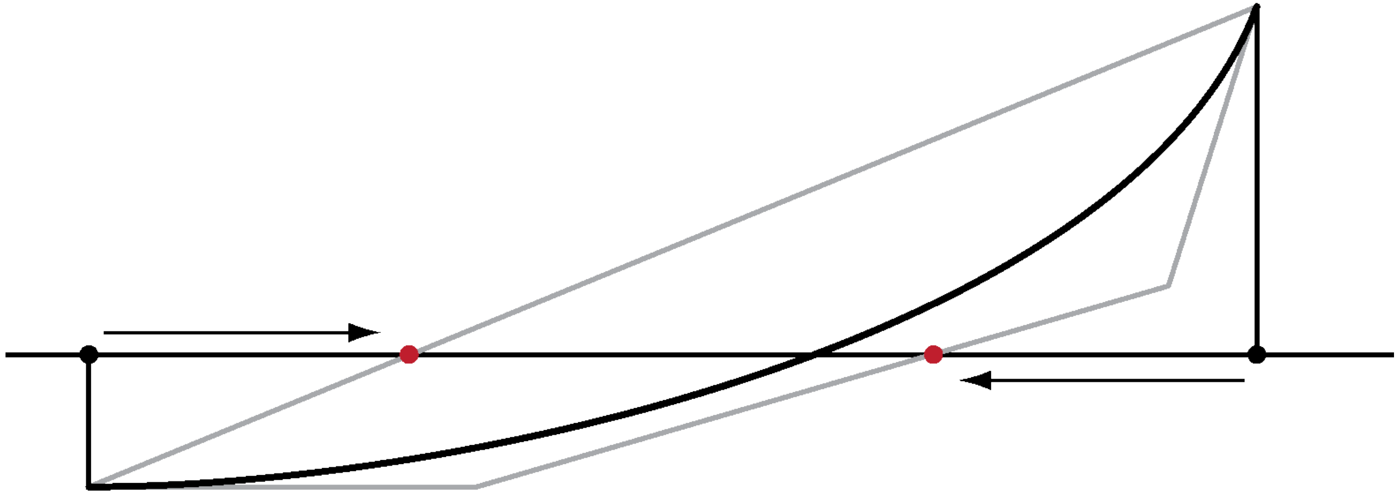
- Quadratic curve
  - Substitute into equation and solve
- Polynomial of a higher degree
  - e.g. Bezier Clipping
- Other functions
  - Finding roots: Newton method, interval bisection, approximating with polyline, …

$$f(x, y) = 0, \qquad P = (p_1, p_2) - \text{intersection point}$$

$$n = \left(\frac{\partial f}{\partial x}(p_1, p_2), \frac{\partial f}{\partial y}(p_1, p_2) \right)$$

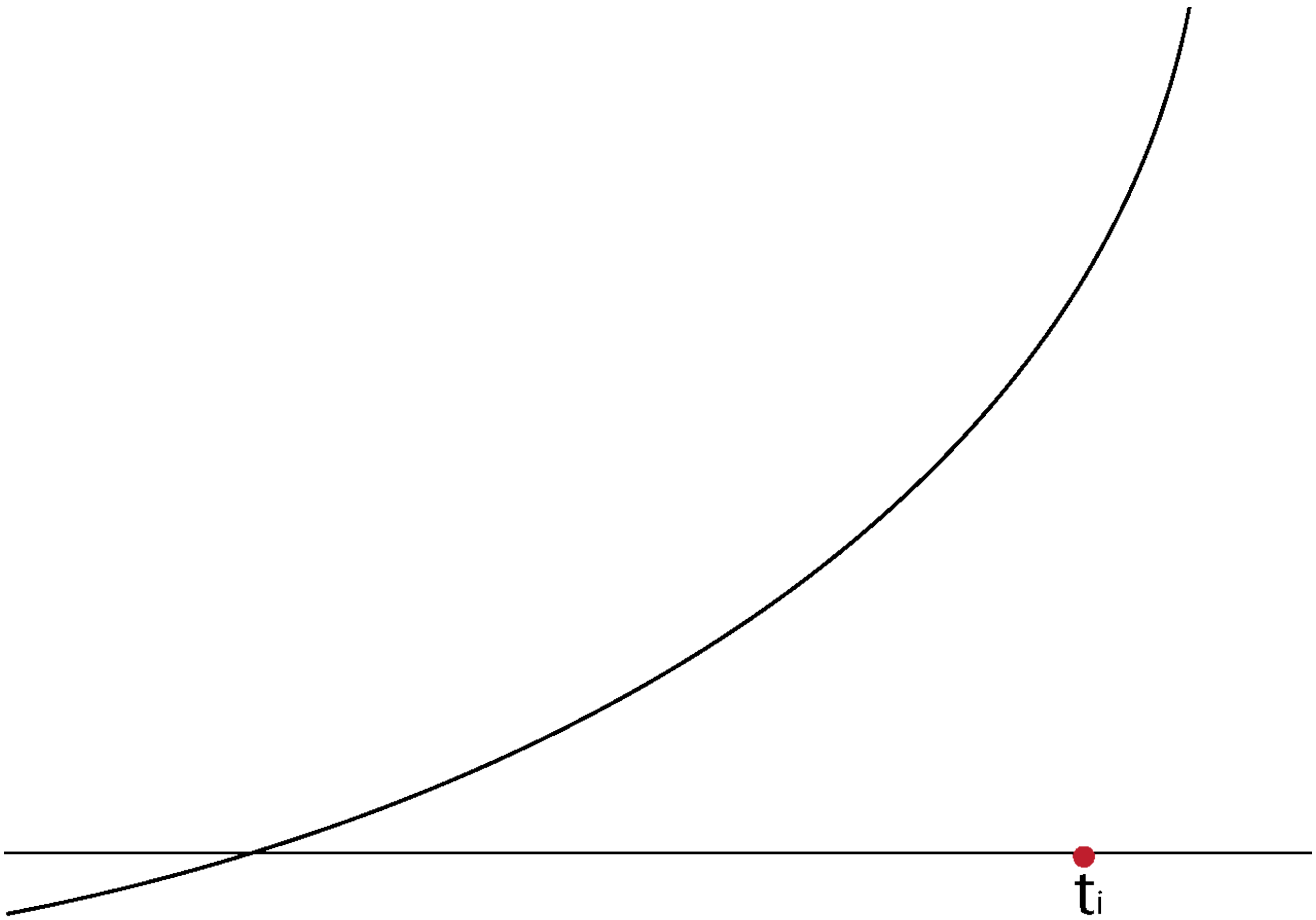
Bezier Clipping

- Express the polynomial curve as Bezier curve
- Control points form a convex hull
  - Exploit this property

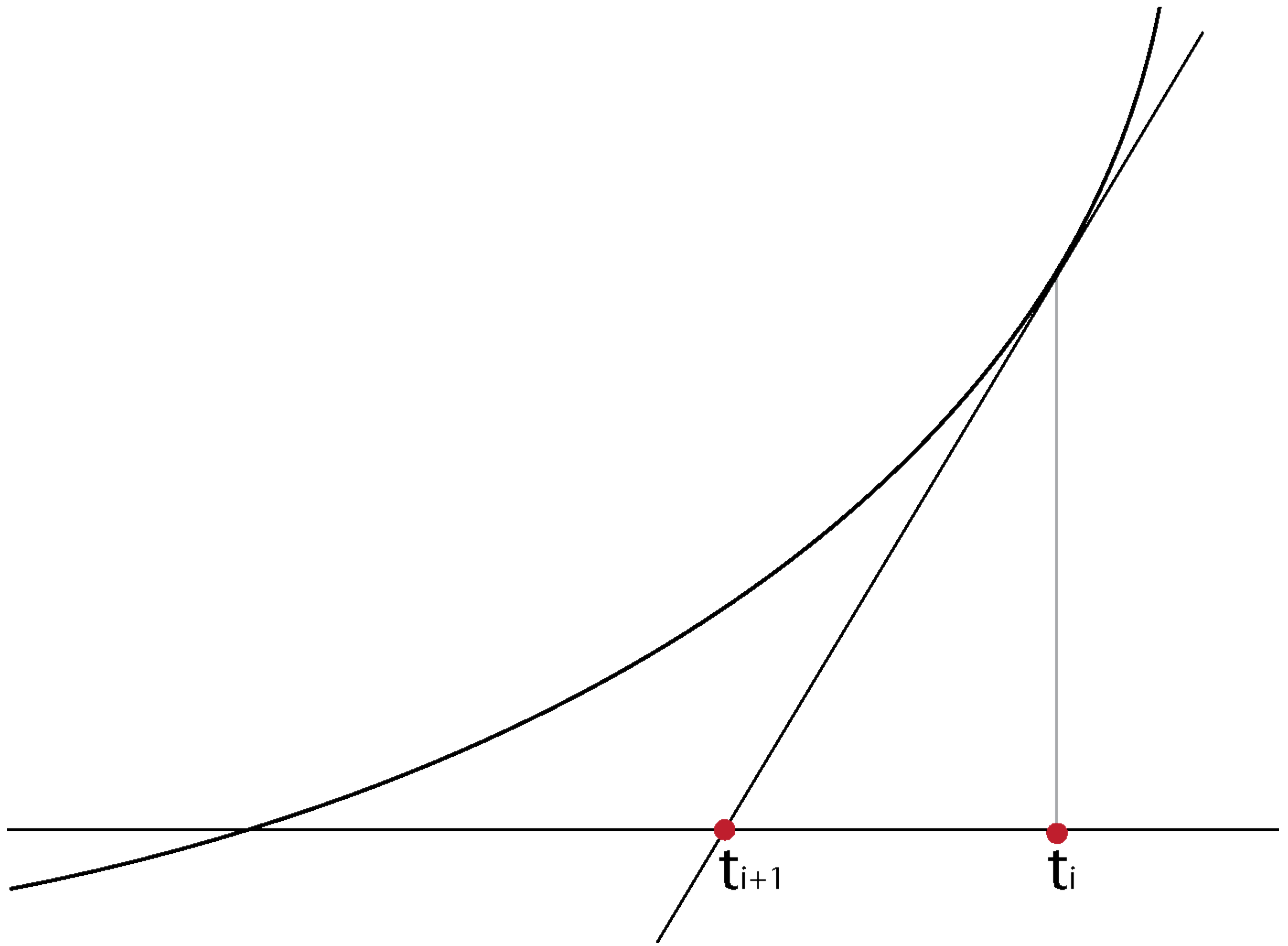


Example of Bezier Clipping

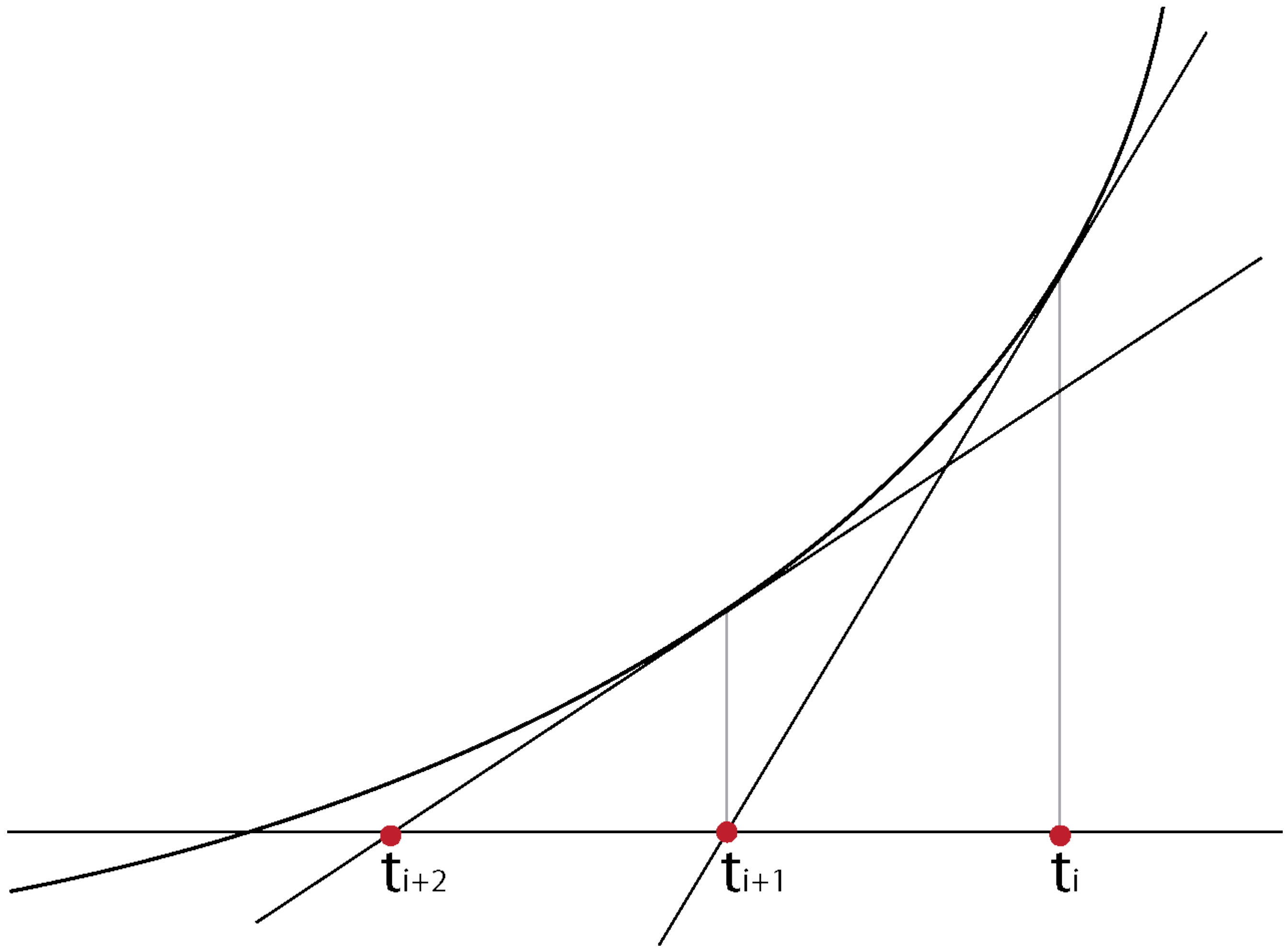
Newton Method - Example



Newton Method - Example



Newton Method - Example



Newton Method

- Tangential direction in (t_i, $f(t_i)$) is $f`(t_i)$

$$y = f'(t_i)x + c$$

- (t_i, $f(t_i)$) lies on the line

$$f(t_i) = f'(t_i)t_i + c$$

$$y = f'(t_i)x + f(t_i) - f'(t_i)t_i$$

- Intersection with x axis (y = 0)

$$f'(t_i)x = f'(t_i)t_i - f(t_i)$$

$$x = t_i - \frac{f(t_i)}{f'(t_i)}$$

$$t_{i+1} = t_i - \frac{f(t_i)}{f'(t_i)}$$

Line-Plane Intersection

- Test if the intersection exist
  - Intersection exist only if sgn(f(A)) <> sgn (f(A`))

$$t = \frac{|AP|}{|AA'|} = \frac{|AP|}{|AP| + |A'P|} = \frac{|f(A)|}{|f(A)| + |f(A')|}$$

$$|AP| = |A\rho| = \frac{|f(A)|}{\sqrt{a^2 + b^2 + c^2}}$$

$$P = A + \frac{|f(A)|}{|f(A)| + |f(A')|}(A' - A)$$

Line-Polygon Intersection in 3D

- Transform into 2D case
- Project the coordinates into the polygon`s plane
  - Forget the largest coordinate of the surface normal
  - Produces the projected polygon with larges area
  - Better numerical stability

Line-Surface Intersection

$$f(x, y, z) = 0$$

$$X = A + \vec{u}t; t \in \langle 0,1 \rangle$$

- Similar to 2D solutions
- Normal at point (p_1, p_2, p_3):

$$n = \left(\frac{\partial f}{\partial x}(p_1, p_2, p_3), \frac{\partial f}{\partial y}(p_1, p_2, p_3), \frac{\partial f}{\partial z}(p_1, p_2, p_3) \right)$$

Clipping

Point clipping, Polygon clipping, Curve and text clipping

Clipping

- Any procedure that identifies portions outside or inside a specified region
- Very important in computer graphics
- Mostly planar clipping
- Use
  - Visibility
  - Extracting part for viewing
  - CSG
  - Selecting part of an image
  - …

Point clipping

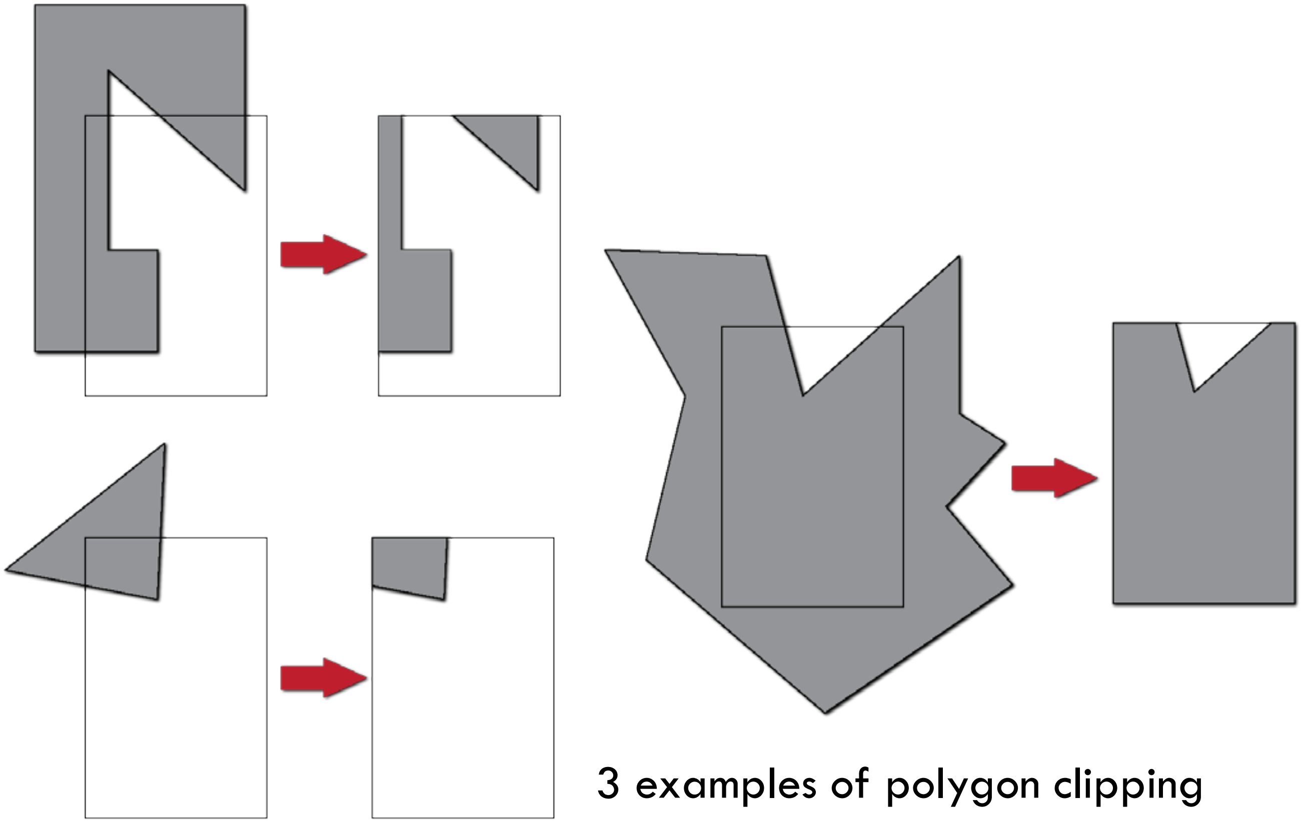
- Clip against a window (axis aligned)
  - Check if coordinates lie in the window
- Clip against a polygon
  - Test if point lies in the polygon

Line Clipping

- See Lesson 5

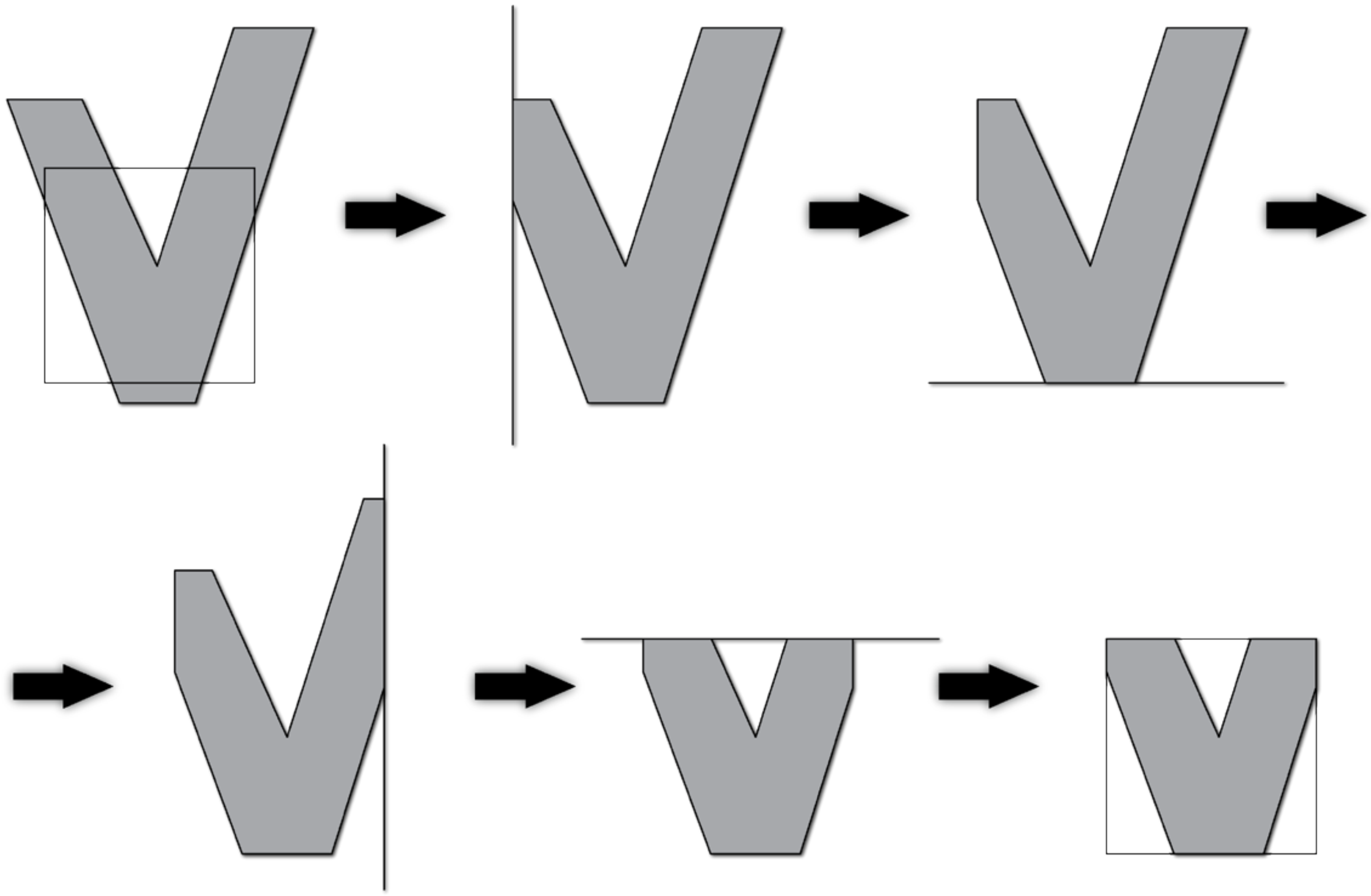
Polygon Clipping

- Deal with many different cases
  - Add and remove edges and vertices



3 examples of polygon clipping

Sutherland-Hodgeman - Example



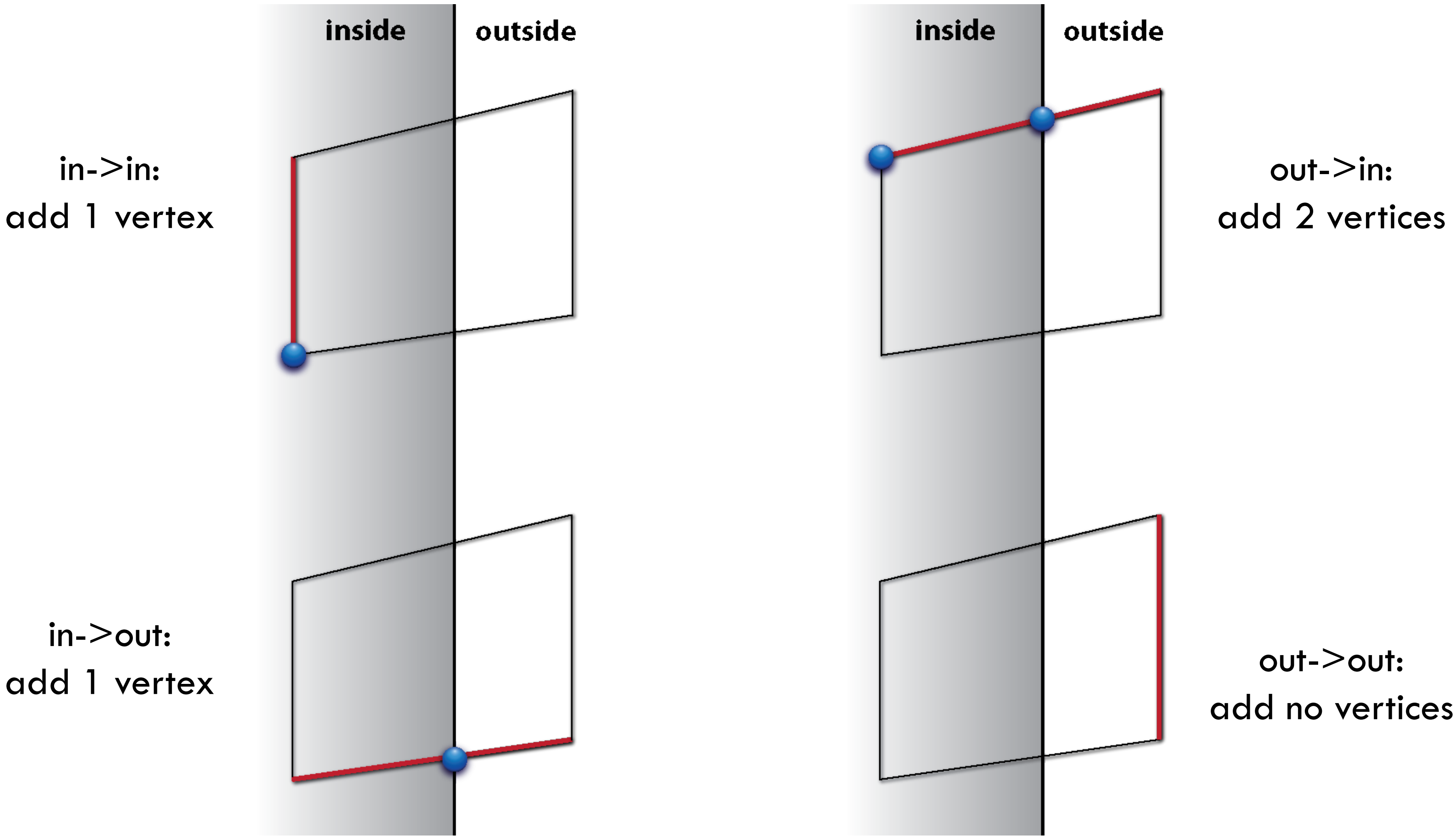
Sutherland-Hodgeman - Algorithm

- Divide and conquer approach
- Clip against infinite clip edges
  - Combine to get the solution
  - Clip against all edges of a polygon (no testing like in line clipping)
- Can be used to clip against arbitrary convex polygon
- Can be extended into 3D
  - Clipping against convex polyhedra

Clip Against Infinite Edge

- Input: list of vertices of a polygon
- Algorithm clips against single infinite edge
- Move along the polygon
- Examine relationship between successive vertices and the clip edge
  - 4 cases
  - 0, 1, or 2 vertices are added in each step
- Output: new list of the vertices of the clipped polygon

Sutherland-Hodgeman - Example



Sutherland-Hodgeman - Conclusion

- Clipping only against convex polygons
- Creates only a single polygon
- Problem with concave polygons
  - Clipped polygon may have multiple separate parts
- Solution
  - Postprocess and create multiple polygons
  - Modify the algorithm
  - Use other clipping algorithm

Weiler-Atherton

- Weiler-Atherton[77] improved in Weiler[80]
- Arbitrary polygon clipping regions
- Sometimes follow the window boundary
  - Depend if the edge is inside to outside or outside to inside
- Clockwise processing
  - out->in: follow polygon boundary
  - in->out: follow window boundary in clockwise direction

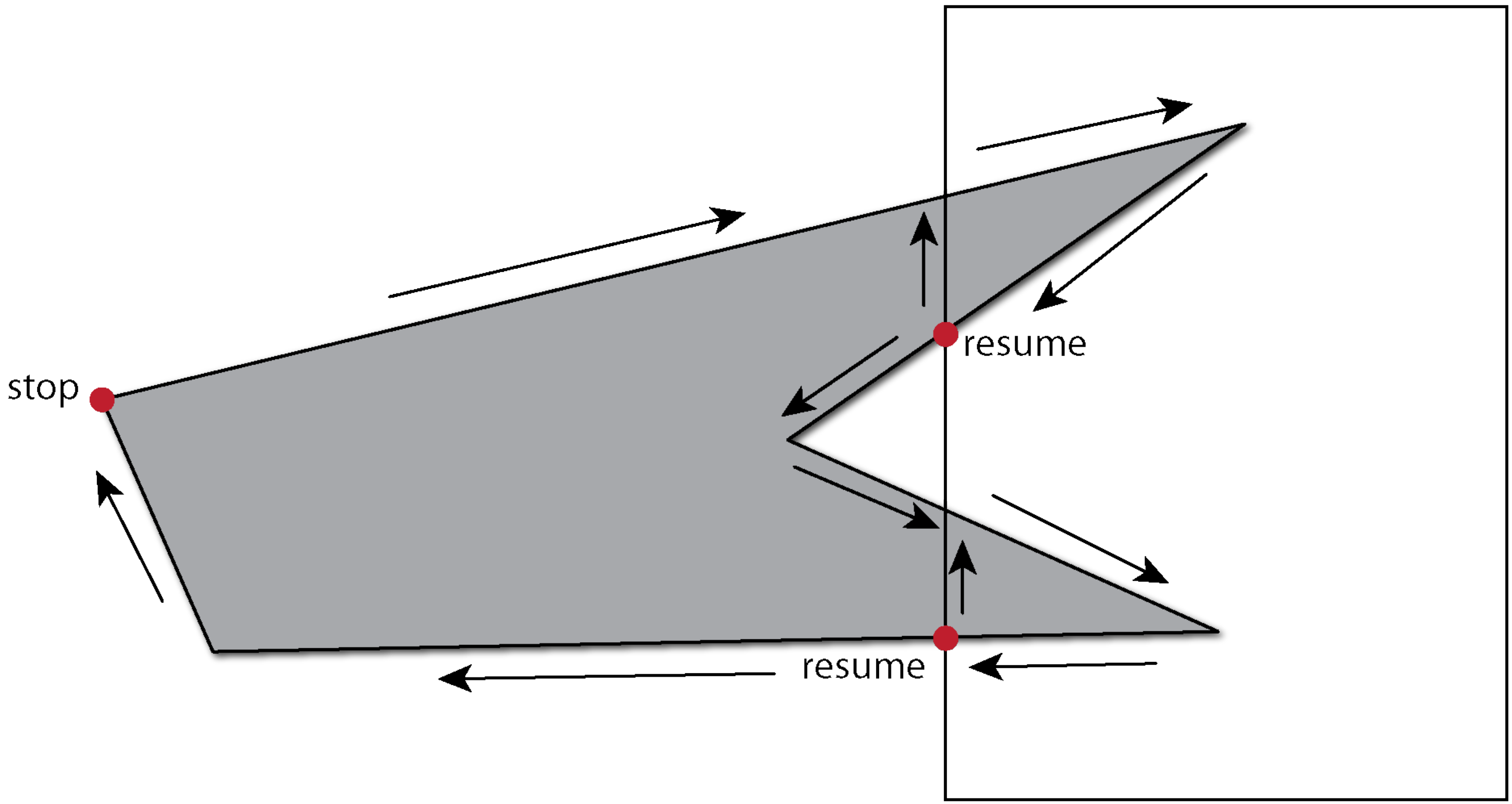
Weiler Algorithm Steps

- 1. Find intersection and insert into polygons
- 2. Mark the nonintersecting polygon points as inside and outside
- 3. Divide intersection points into two groups (create lists)
  - Entering list (out->in edge)
  - Leaving list (in->out edge)
- 4. Clip

Weiler Clipping Step

- 1. Remove intersection point
  - If there is none then we are done
- 2. Follow the clipped polygon vertices to the next intersection
- 3. Switch to clipping polygon vertex list
- 4. Follow the clipping polygon vertices to the next intersection
- 5. Switch to clipped polygon vertex list
- 6. Repeat 2-5 until we are at the starting vertex

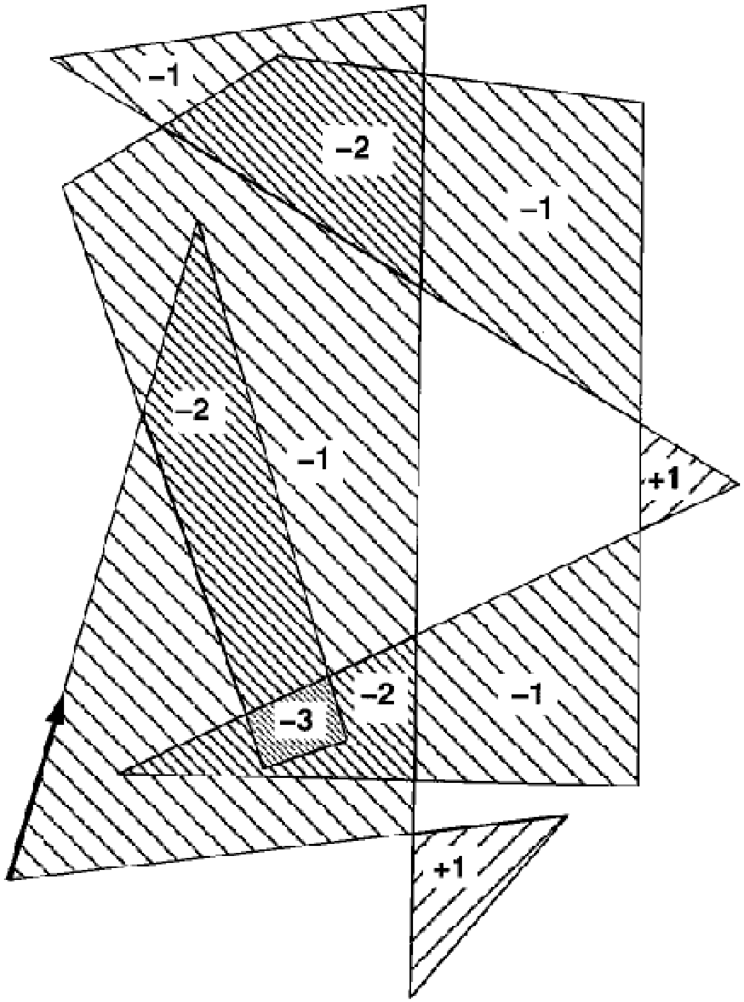
Weiler-Atherton - Example



Greiner-Hormann

- Similar to Weiler algorithm
- Less memory consumption
- Does not need whole boundary representation
- Exploit the winding number to check if we are inside or outside
- 3 phases
  - Compute intersections
  - Mark as entry and exit
  - Crete polygons

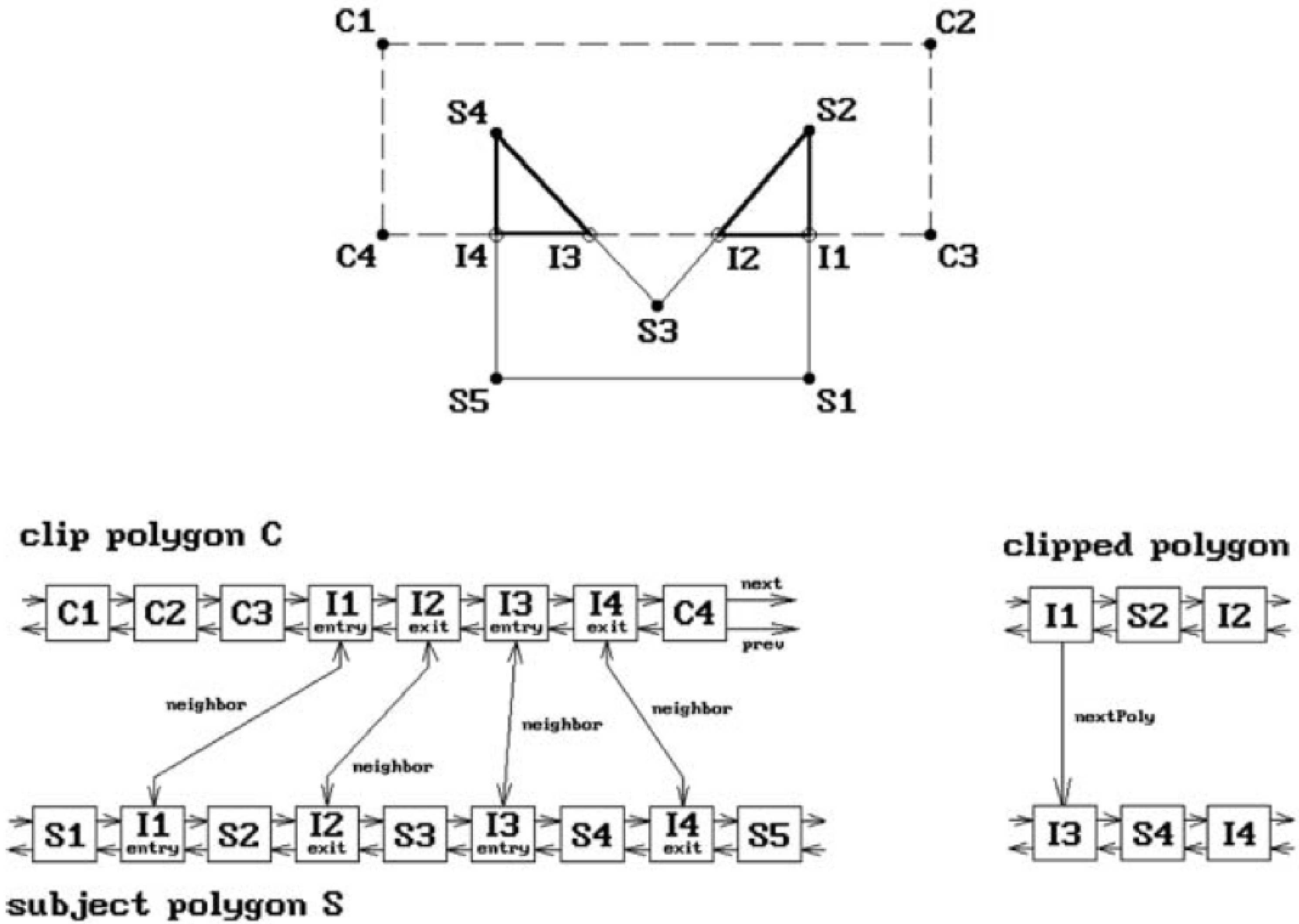
Winding Number



Greiner-Hormann: Vertex


```
vertex = record
  float          x, y;
  vertex pointer next, prev;
  boolean        intersect;
  boolean        entry;
  vertex pointer neighbor;
  float          alpha;
  vertex pointer nextPoly;
end;
```

Greiner-Hormann: Data Structure



Greiner-Hormann: Algorithm


```
vertex pointer current;

while  more unprocessed subject intersection points  do
    begin
        current := pointer to first remaining unprocessed subject intersection point;
        NewPolygon (P);
        NewVertex (current);
        repeat
            if  current→entry
                then
                    repeat
                        current := current→next;
                        NewVertex (current);
                    until  current→intersect
                else
                    repeat
                        current := current→prev;
                        NewVertex (current);
                    until  current→intersect
            current := current→neighbor;
        until  Closed (P);
    end;
```

Other Algorithms

- Rectangular Region
  - Liang-Barsky
  - Maillot
- Arbitrary polygon polygons with holes and self intersecting polygons
  - Vatti

Curve and text clipping

- Curve clipping
  - Similar methods to the polygon clipping
  - Nonlinear equation
  - Check if object lies in the clipping polygon (use bounding box)
- Text clipping
  - Clip whole string or character (use bounding box)
  - Clip the boundary representing the character

Questions ???