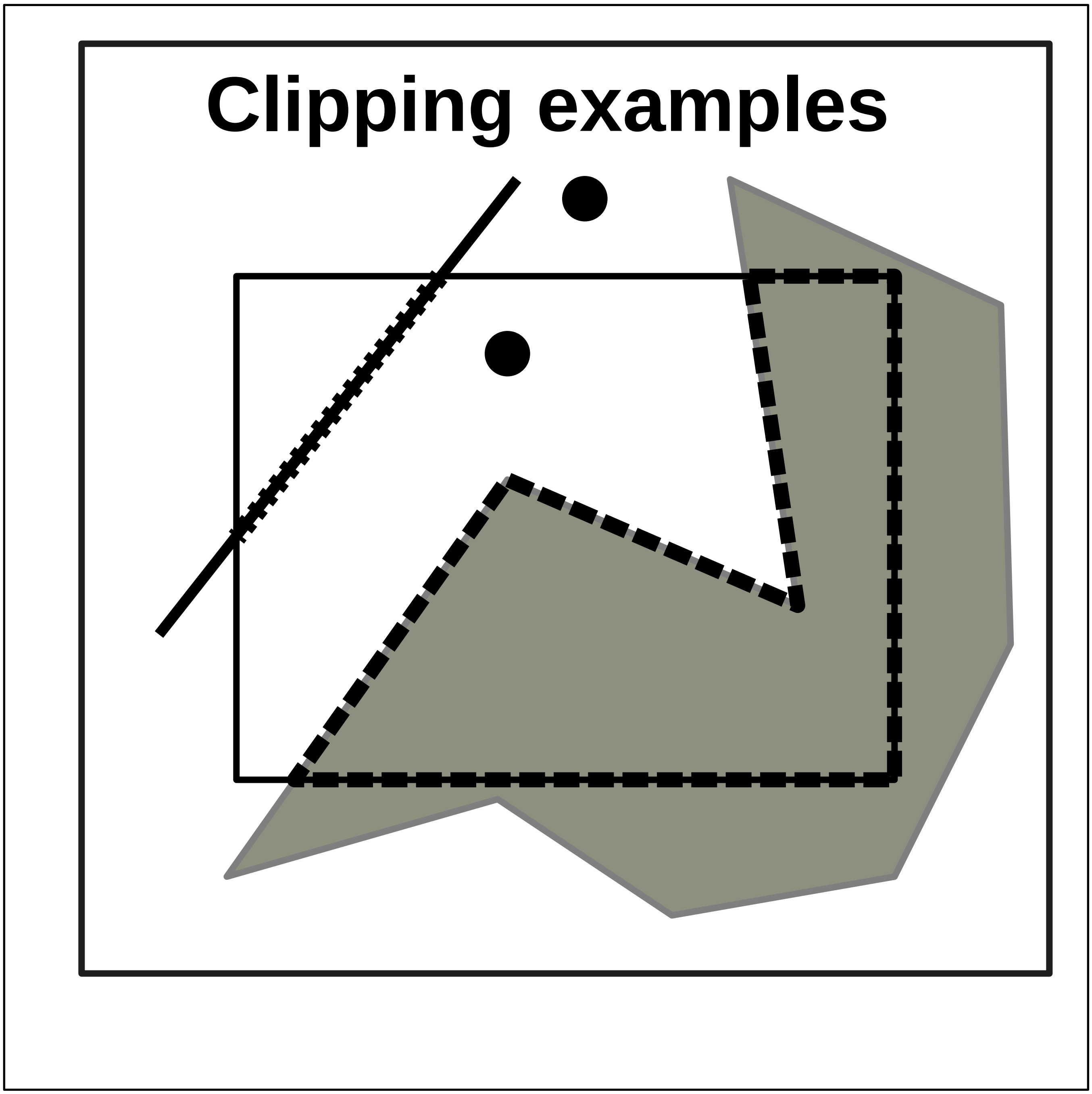
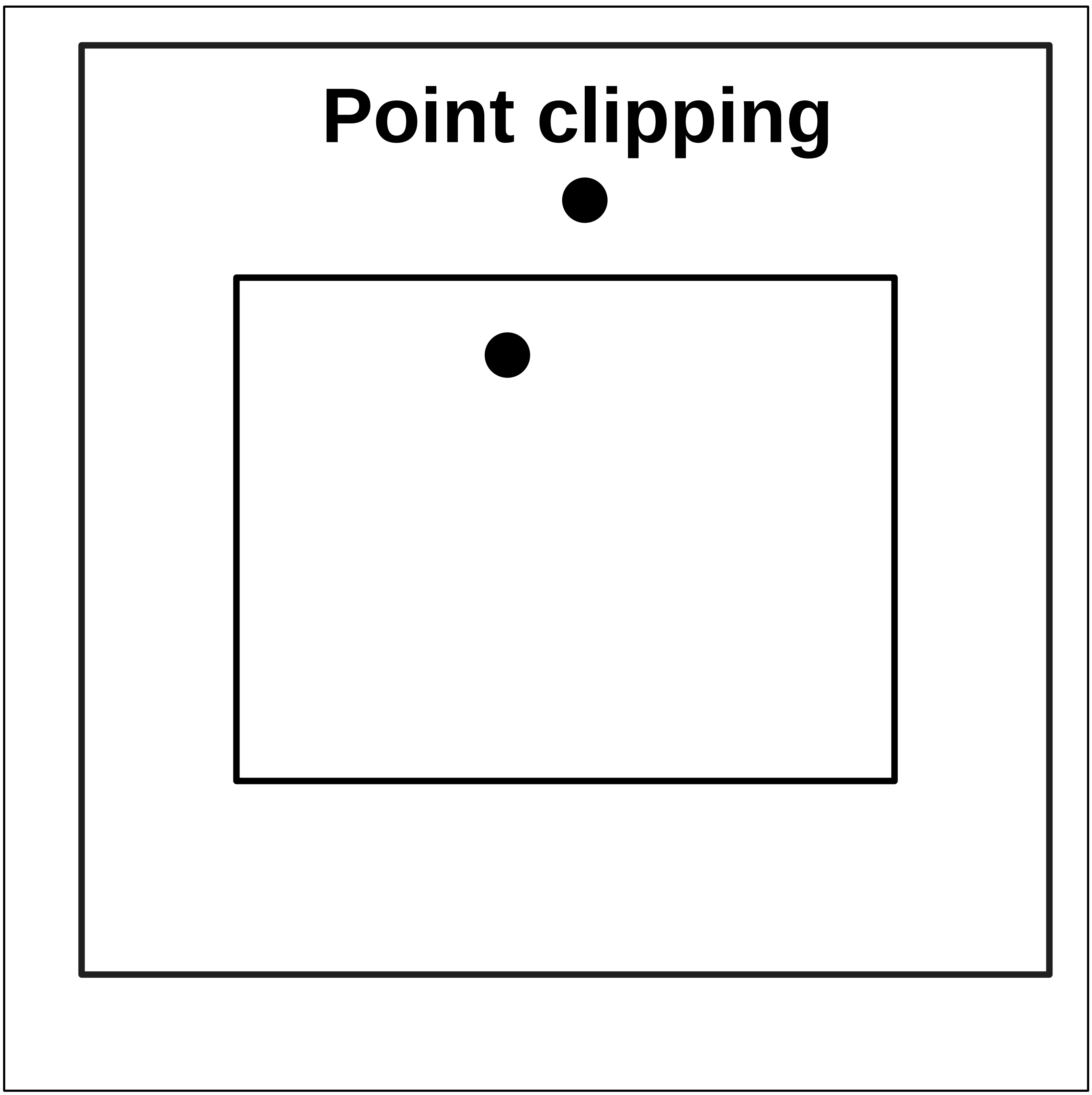
2D Clipping



2D Clipping

- Points
- Lines
- Polygons

Point Clipping



Point Clipping

- **Assume that the window is defined as**

$$x_l$$

$$x_r$$

$$y_b$$

$$y_t$$

- **Then point clipping is straightforward and simple**
- **Point (x, y) is plotted if**

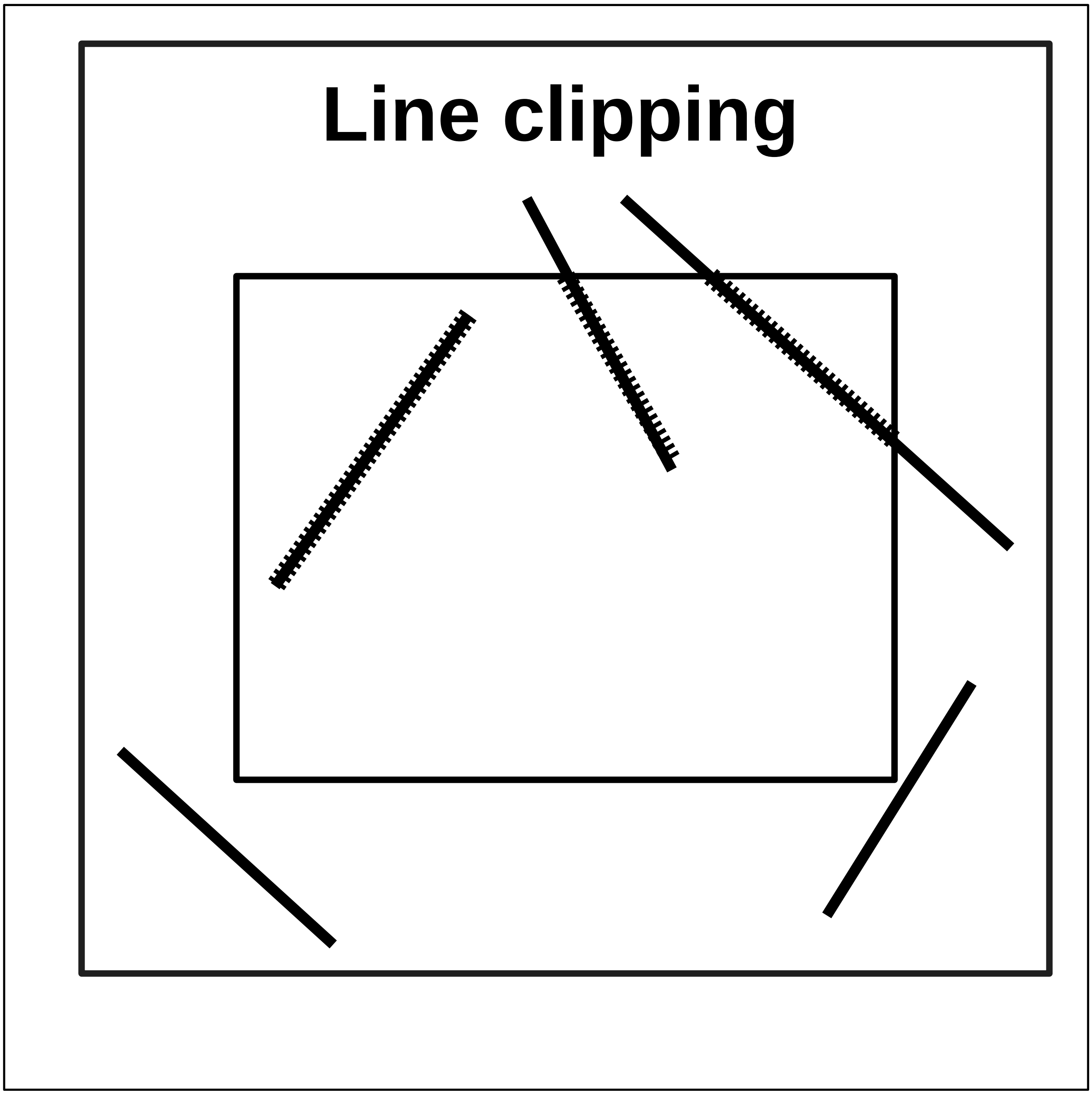
$$x \in [x_l, x_r],$$

and

$$x \in [y_b, y_t]$$

- **Pay attention to**
  (1) homogeneous coordinates
  (2) equations of lines

Line Clipping



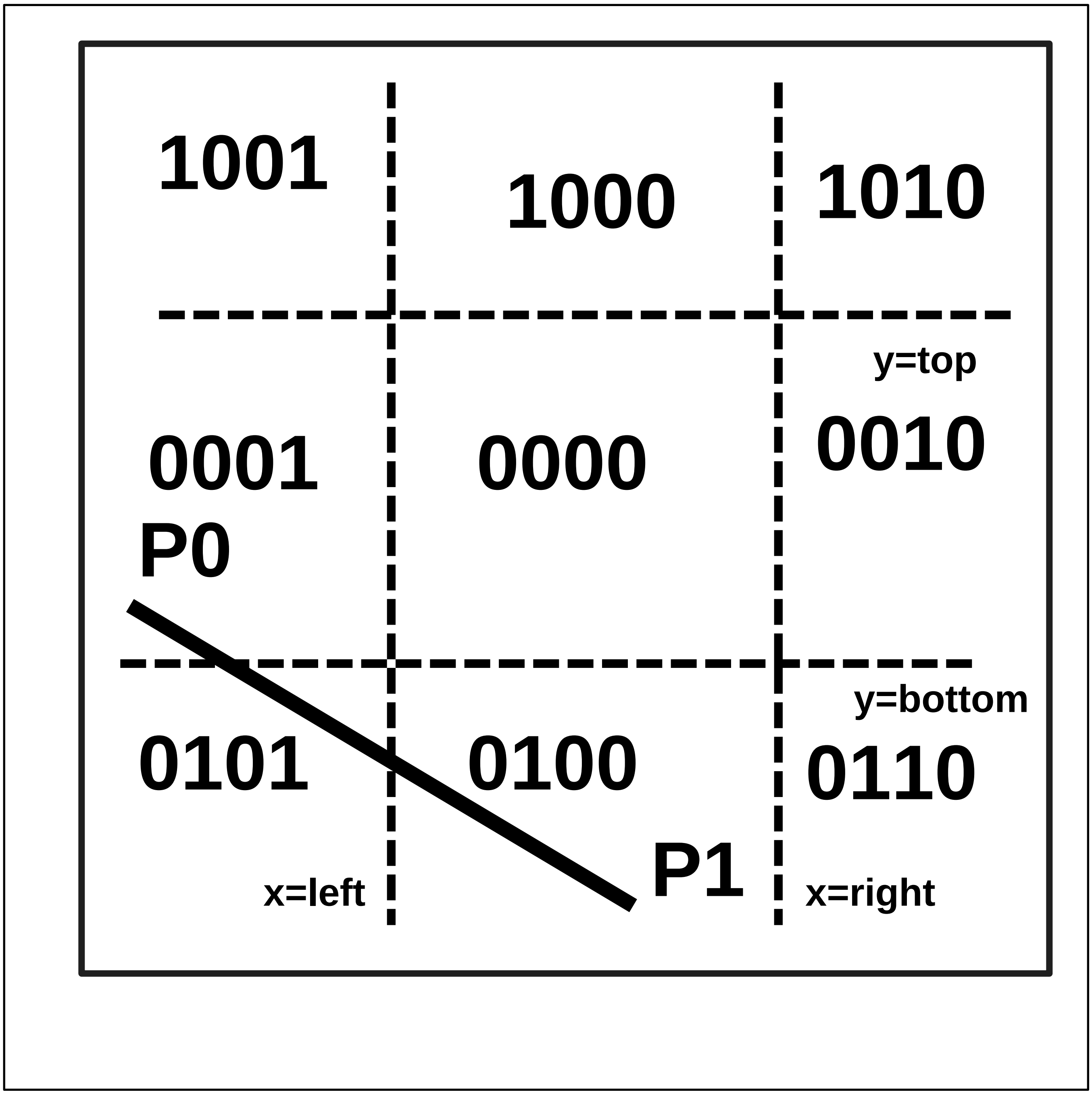
Line Clipping

- Line clipping operations should comprise the following cases
  - totally plotted
  - partially plotted
  - not plotted at all
- Please note that even though neither of two vertices is within the window, certain part of the line segment may be still within !
- There are many different techniques for clipping lines in 2D
- The fundamentals are (1) line equations and (2) intersection computation
- Next, we will discuss Cohen-Sutherland algorithm

Cohen-Sutherland Algorithm

- **It is not the most efficient algorithm**
- **It is one the most commonly used**
- **The key technique is 4-bit code:**
  $TBRL$ **where**
  T **is set (to 1) if** $y > top$
  B **is set (to 1) if** $y < bottom$
  R **is set (to 1) if** $x > right$
  L **is set (to 1) if** $x < left$

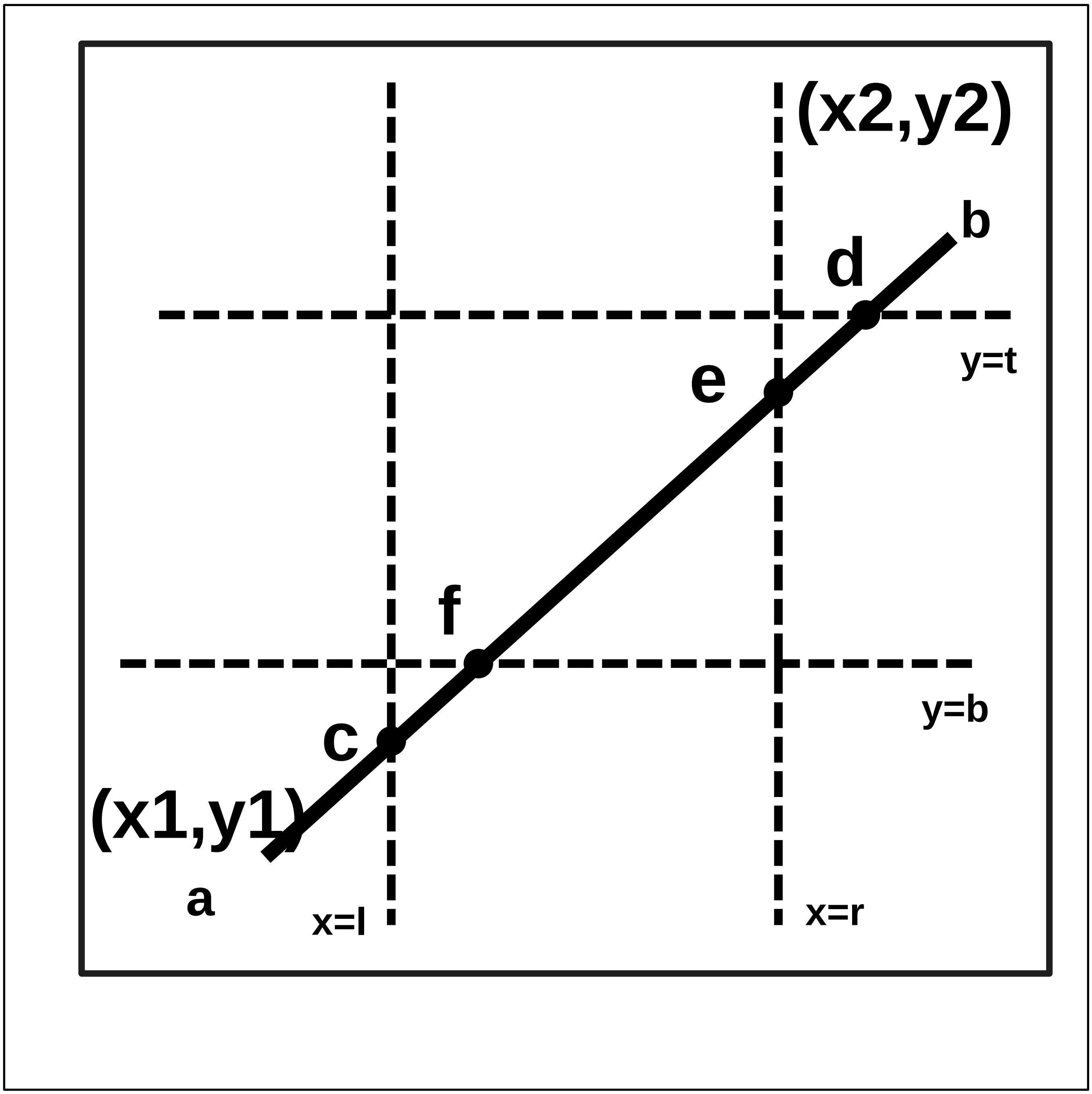
Window Regions



Algorithm

- **Assume two endpoints are p_0 and p_1**
- **If $code(\mathrm{p}_0)$ OR $code(\mathrm{p}_1)$ is 0000,
  the line can be trivially accepted,
  the line is drawn**
- **If $code(\mathrm{p}_0)$ AND $code(\mathrm{p}_1)$ is NOT 0000,
  the line can be trivially rejected,
  the line is not drawn at all**
- **Otherwise, compute the intersection points of
  the line segment and window boundary lines
  (make sure to check all the boundary lines)**

Line Intersection



Intersection Computation

- **Line equation**

$$y = y_1 + m(x - x_1)$$

where

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

- **Line intersection with the left vertical boundary**

$$x = l$$

Assume the intersection is c

$$\begin{cases} x = l \\ y = y_1 + m(l - x_1) \end{cases}$$

Line ab **is clipped w.r.t.** $x = l$**, now it becomes** cb

- **Line intersection with the top boundary**

$$y = t$$

Assume the intersection is d

$$\begin{cases} y = t \\ x = \frac{1}{m}(t - y_1) + x_1 \end{cases}$$

Line cb is clipped w.r.t. $y = t$, line cb becomes cd

- **Line intersection with the right boundary**

$$x = r$$

Assume the intersection is e

$$\begin{cases} x = r \\ y = y_1 + m(r - x_1) \end{cases}$$

Line cd is clipped w.r.t. $x = r$, line cd becomes ce

- **Line intersection with the bottom boundary**

$$y = b$$

Assume the intersection is f

$$\begin{cases} y = b \\ x = \frac{1}{m}(b - y_1) + x_1 \end{cases}$$

Line ce is clipped w.r.t. $y = b$, line ce becomes fe

- **So, the entire process is**

$$ab \Rightarrow$$

$$cb \Rightarrow$$

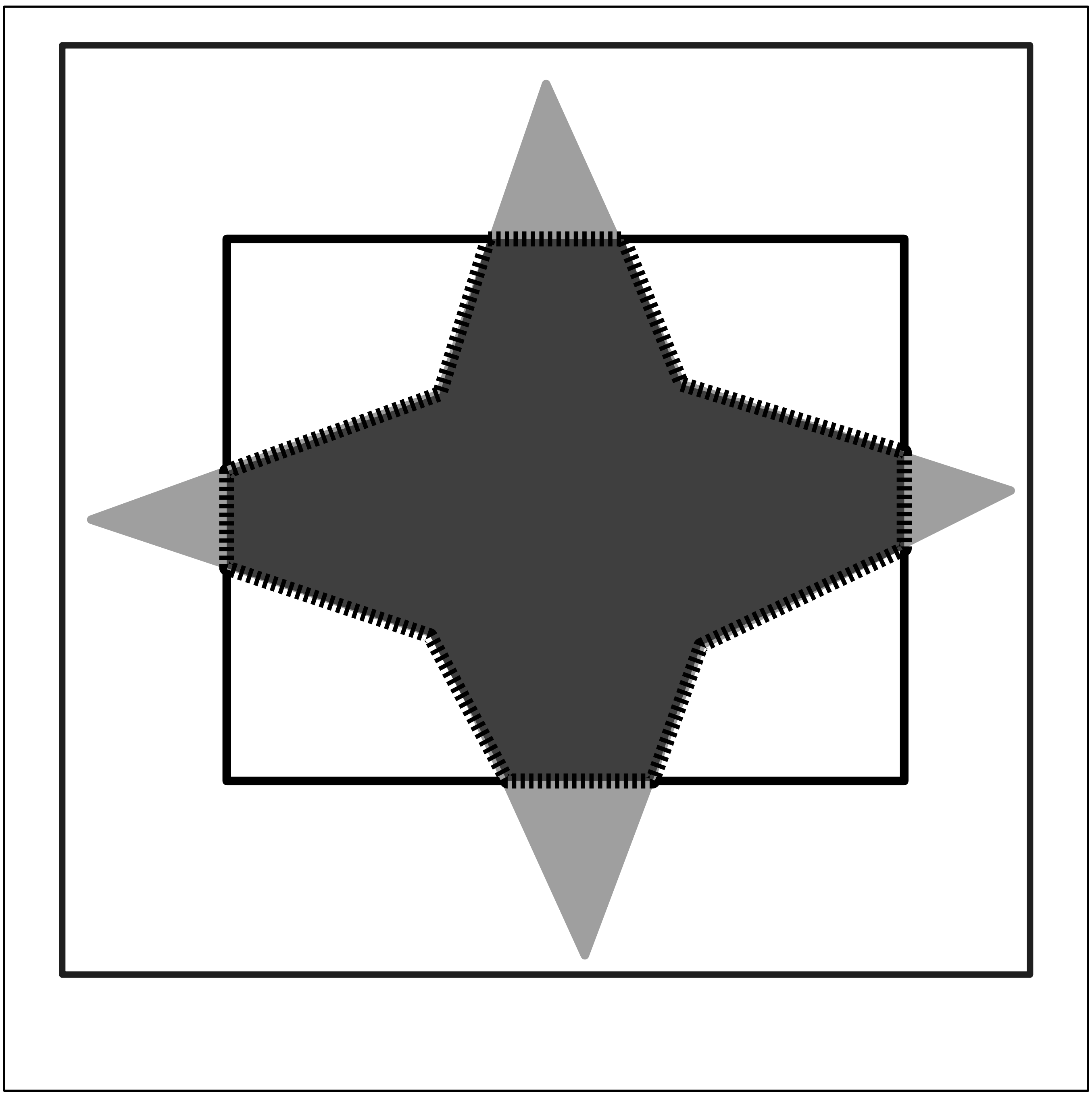
$$cd \Rightarrow$$

$$ce \Rightarrow$$

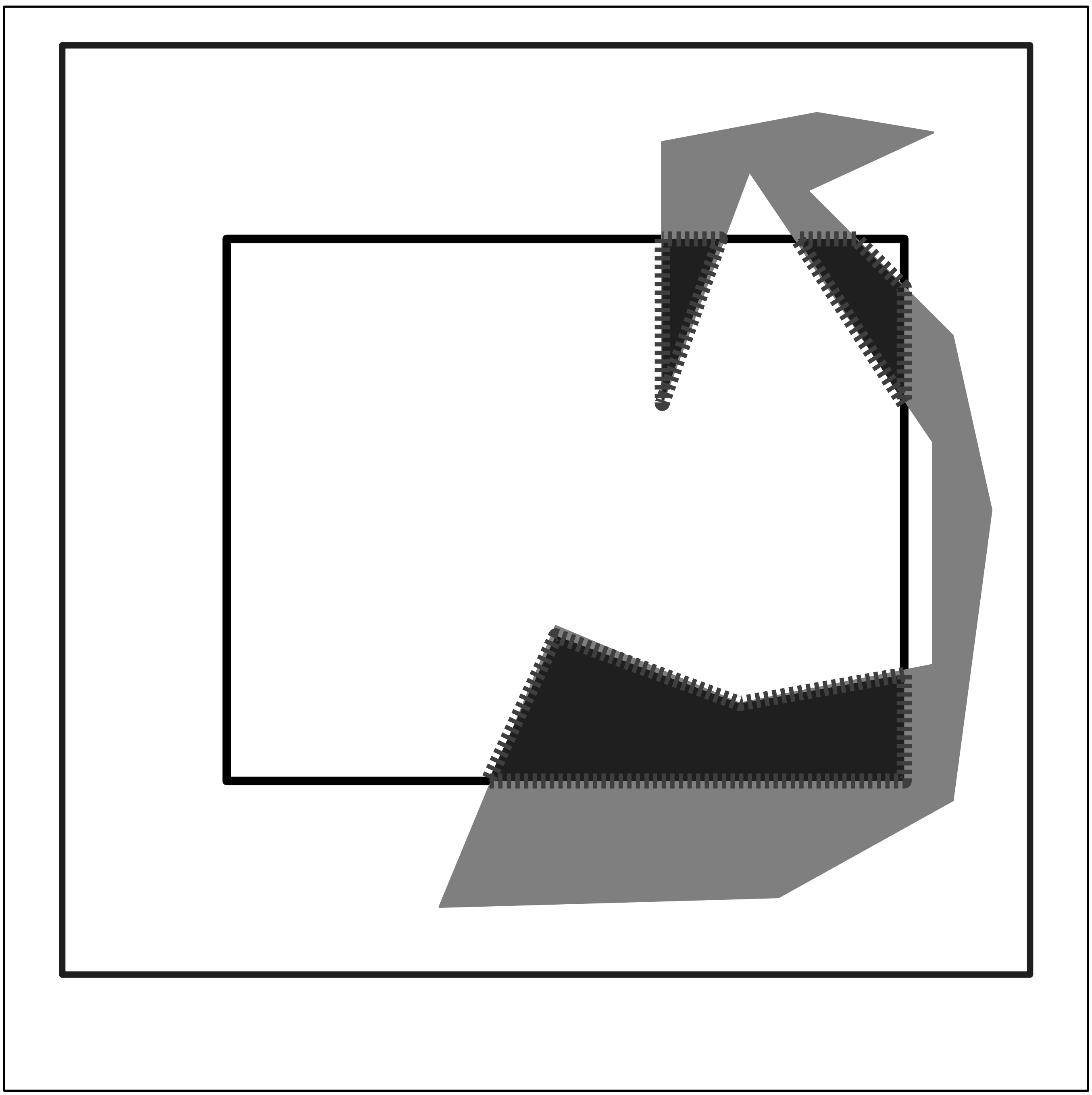
$$fe$$

- **Note that, various improvements are possible !**
  using parametric representation of line questions, P230
  create more regions around the clip window, P233
  line clipping using polygon
  - **convex polygon, P235**
  - **concave polygon, split in into several convex polygon, P236**

Polygon Clipping



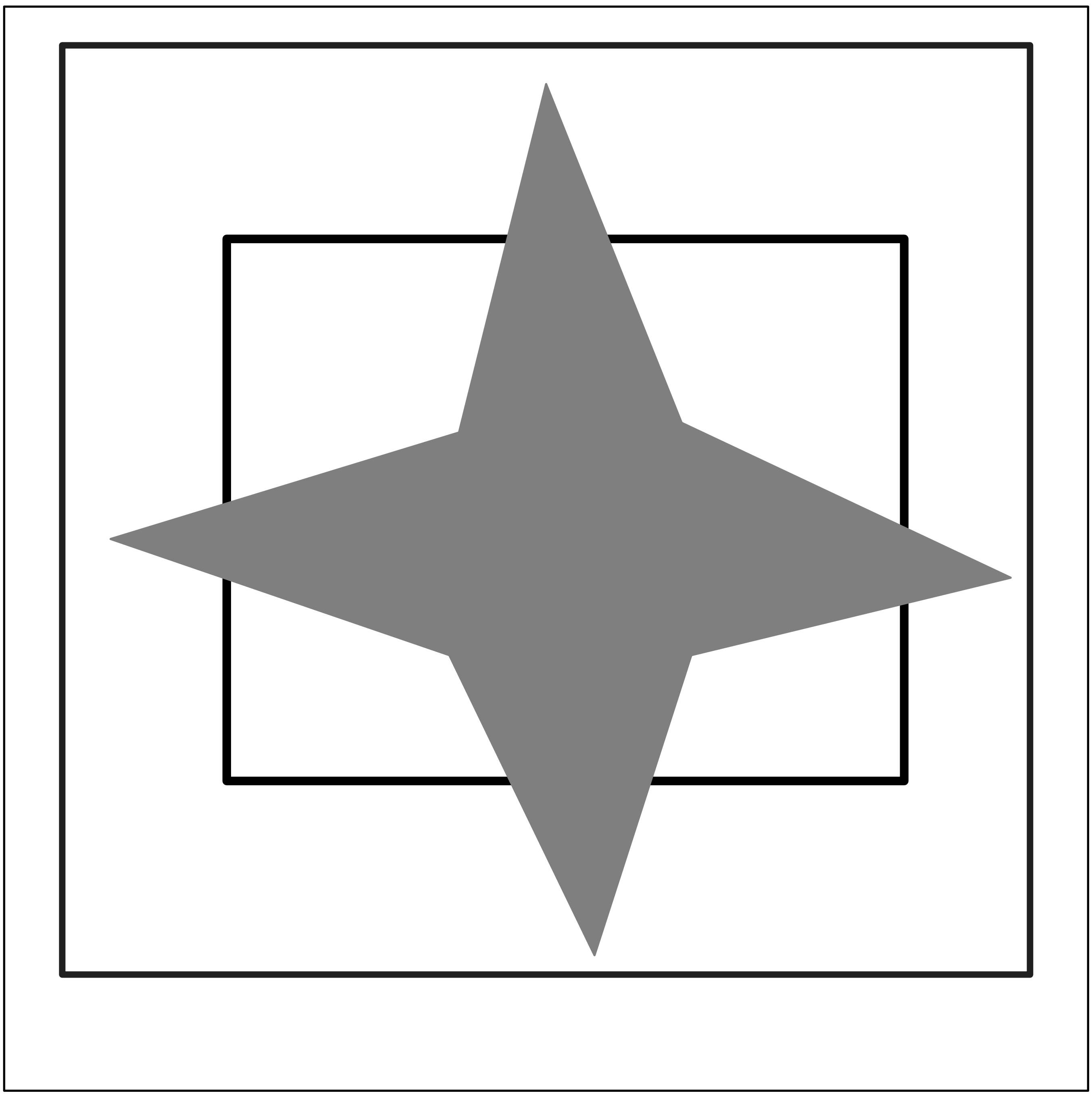
Polygon Clipping



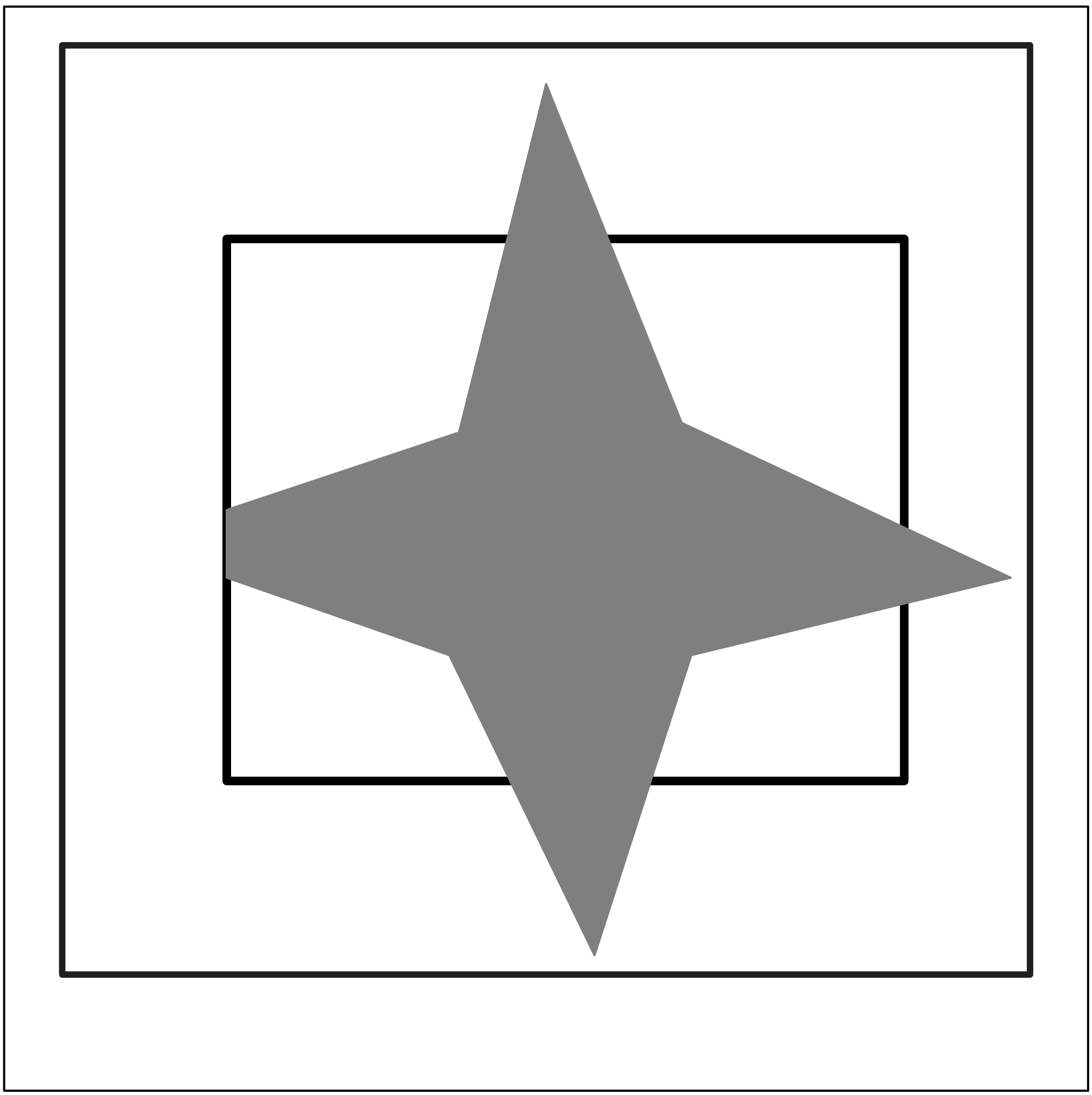
Polygon Clipping

- Line clipping algorithms will lead to a set of disjoint line segment chains
- In general, clipping each edge will not work !
- We shall not clip each edge of the polygon w.r.t. the window boundary one at a time
- We treat the polygon as a whole object
- Clip the entire object against each boundary of the window
- Sutherland-Hodgman algorithm
  - any polygon (convex or concave)
  - any convex clipping polygon

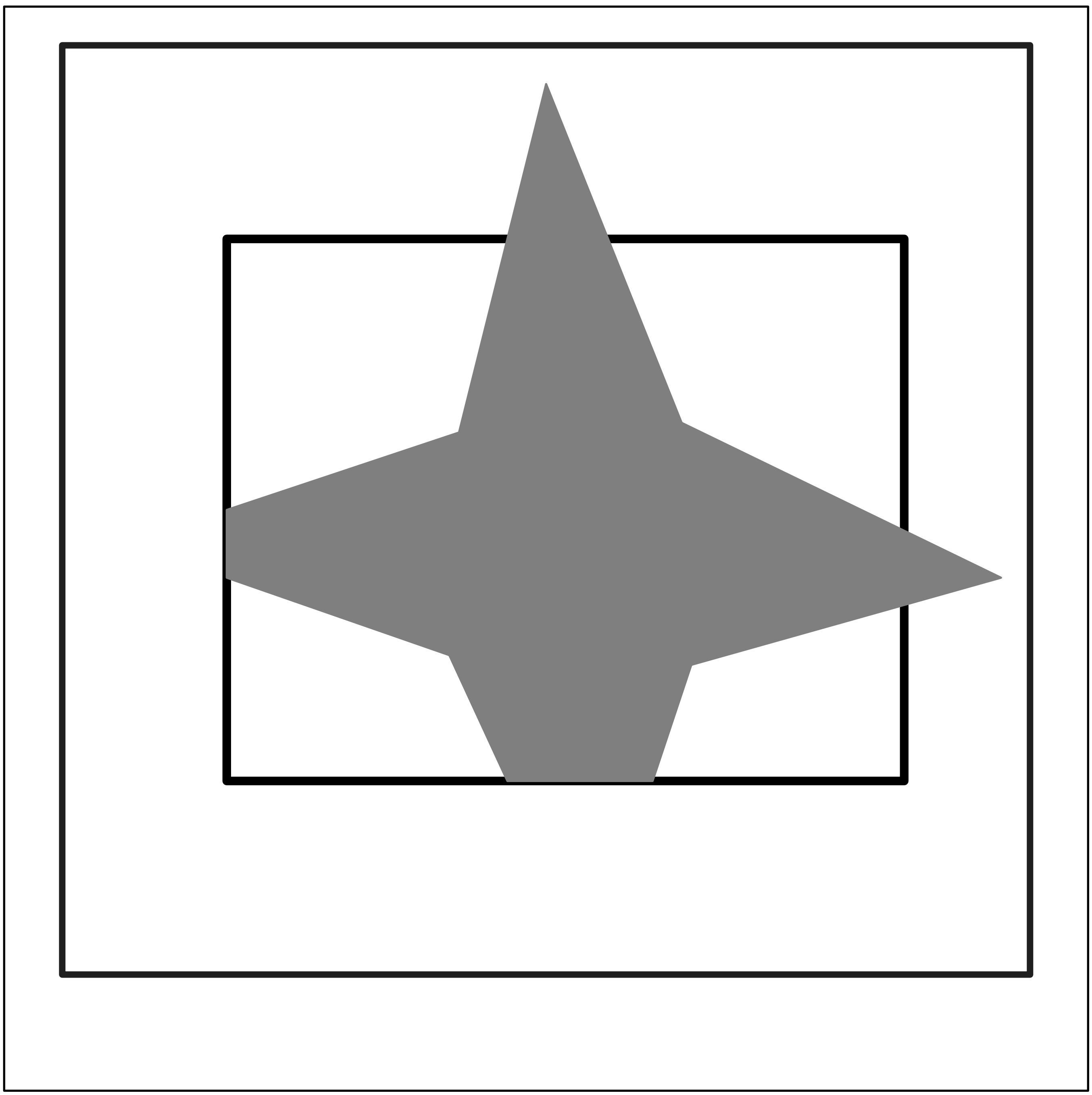
Polygon Clipping Example



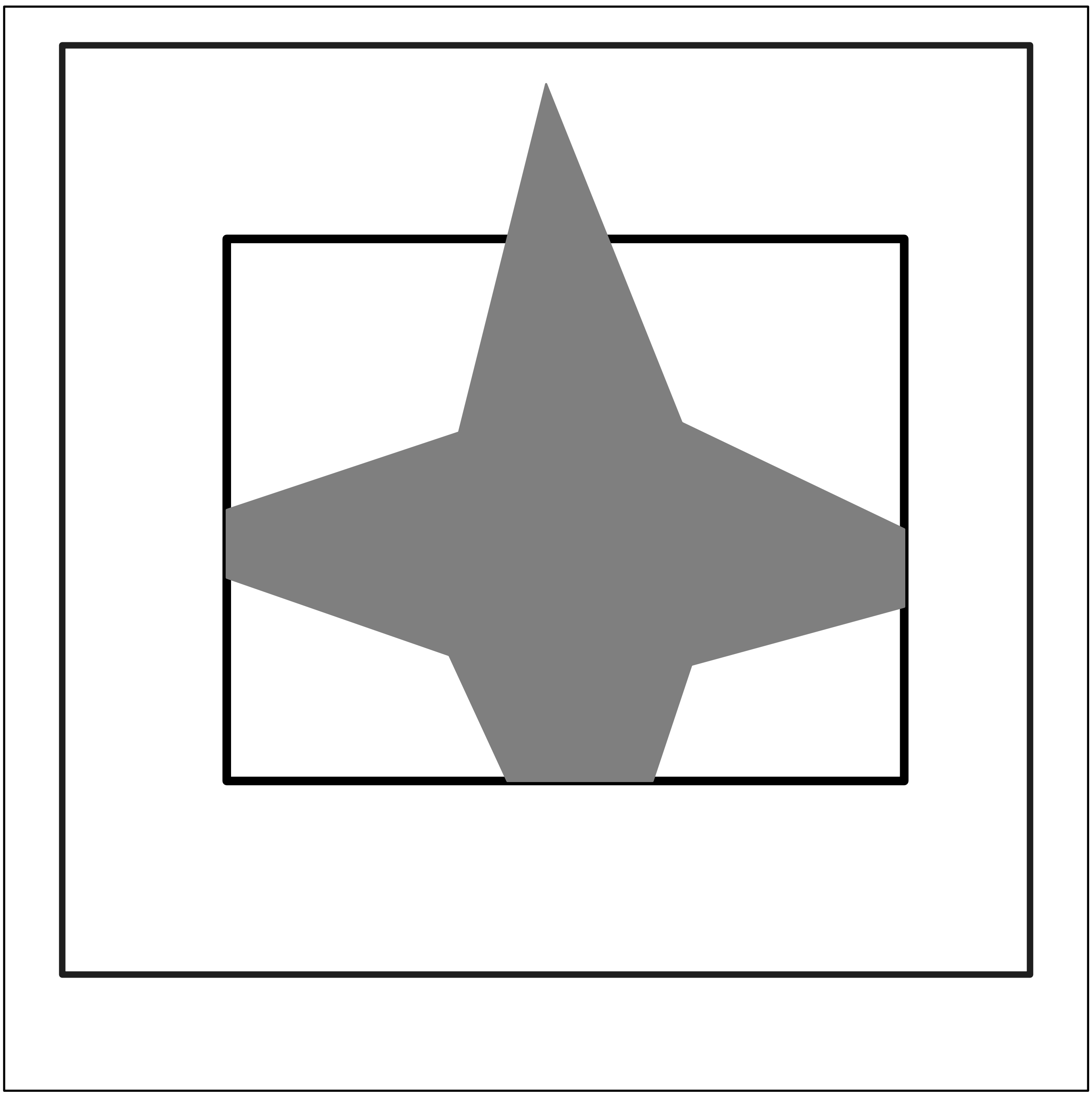
Polygon Clipping Example



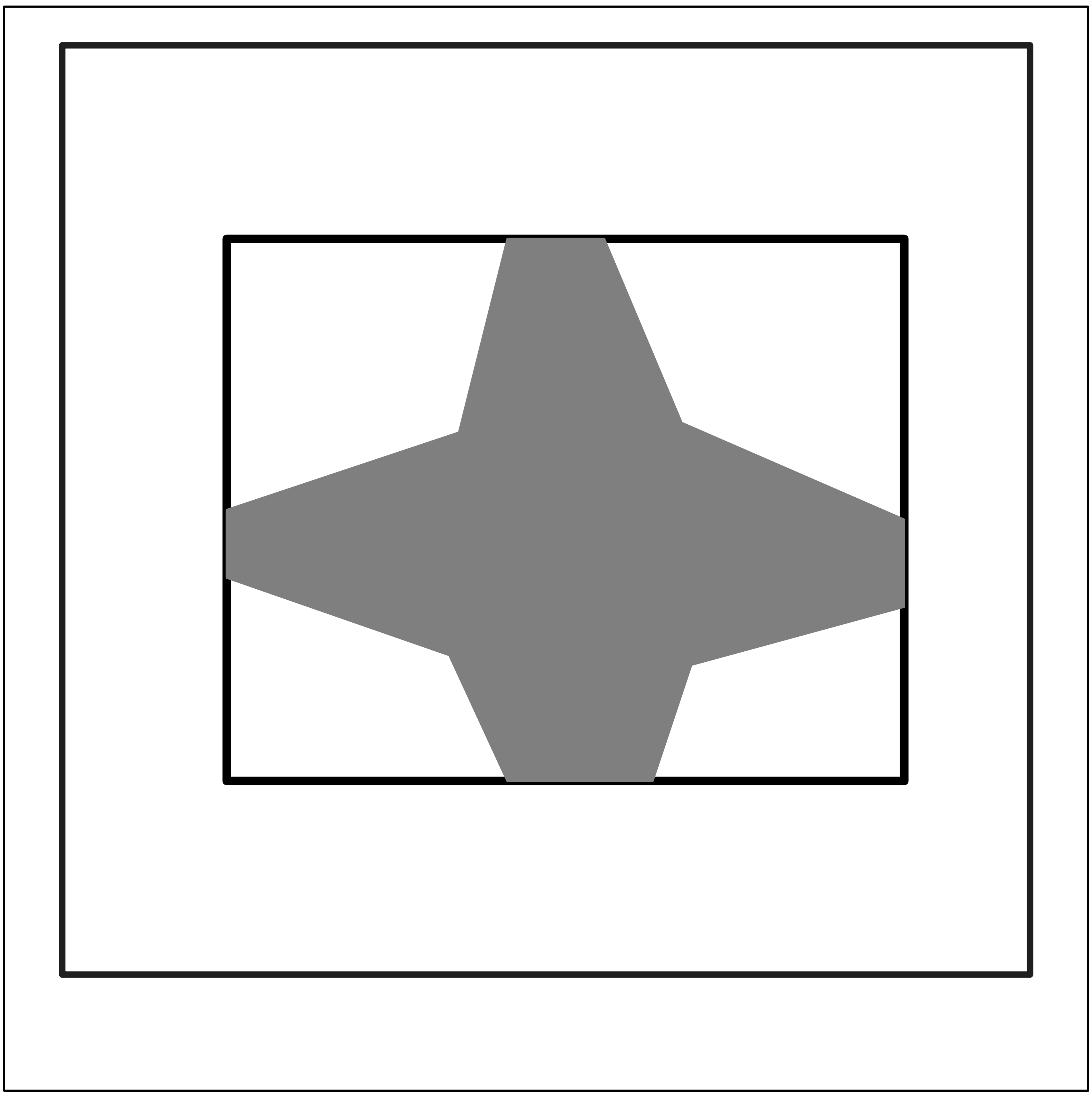
Polygon Clipping Example



Polygon Clipping Example



Polygon Clipping Example



Algorithm

- Sutherland-Hodgman algorithm
- Vertex list of the current polygon $\Rightarrow$
- Clip against edges of the window boundary $\Rightarrow$
- New vertex list of the new polygon
- The algorithm clips against all four edges in a sequential order, producing a new vertex list each time

Polygon Clipping Example