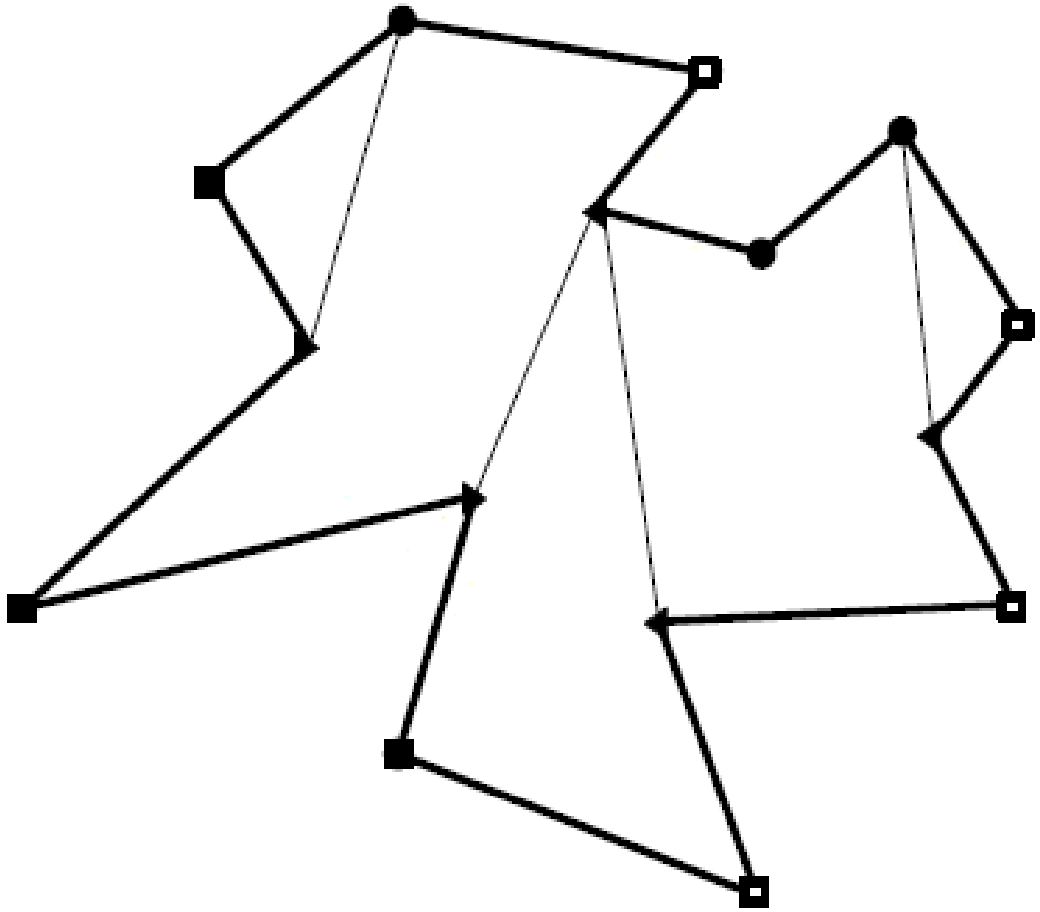
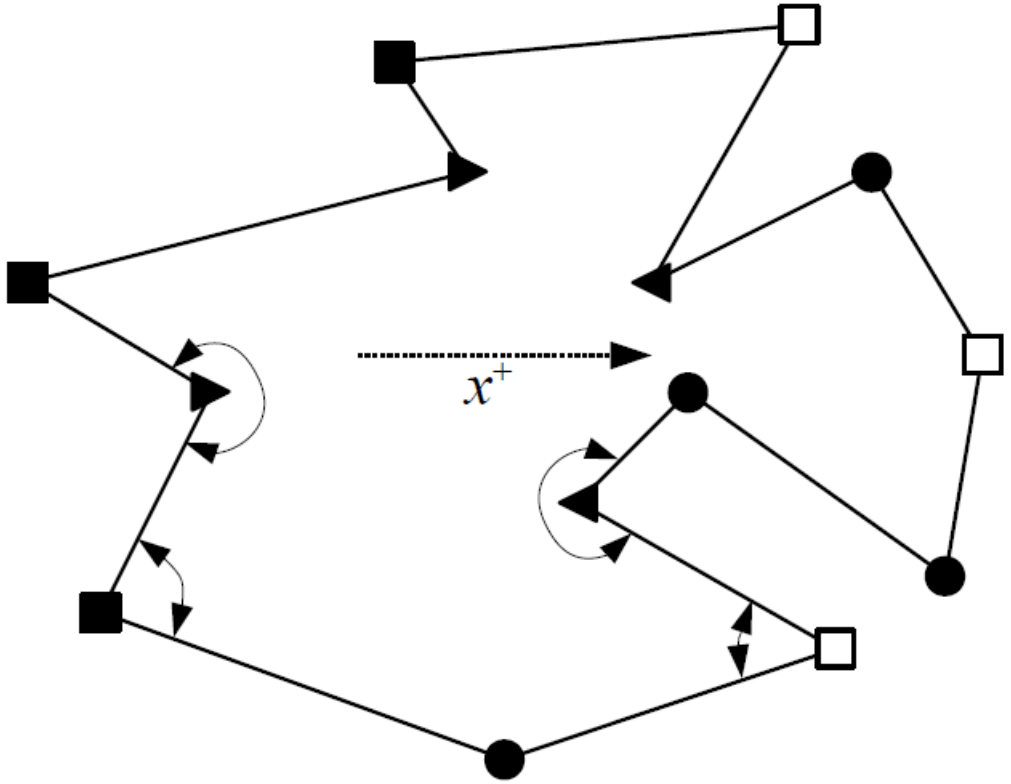
Polygon decomposition into monotone polygons



Vertex types

- START vertex (2 edges on the right and $\alpha < \pi$)
- END vertex (2 edges on the left and $\alpha < \pi$)
- SPLIT vertex (2 edges on the right and $\alpha > \pi$)
- MERGE vertex (2 edges on the left and $\alpha > \pi$)
- REGULAR vertex (1 edge on the left and 1 edge on the right)



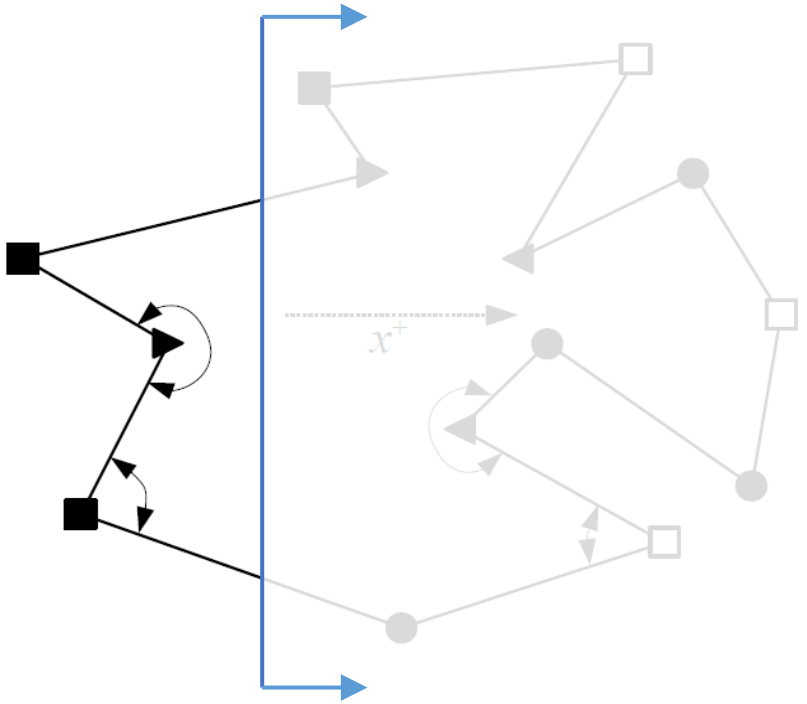
Decomposition of polygon **P** into monotone polygons

Global algorithm

1. Sort the vertices of **P** in a lexicographic order (i.e. first on x and then on y) and stock them in a priority queue **Q**
2. Initialize an empty binary tree **T** referred to as the status
3. **while Q** is not empty
   Remove the vertex v_i from **Q**
   Call the appropriate procedure to handle the vertex, depending on its type (see following slides)

Polygon decomposition into monotone polygons

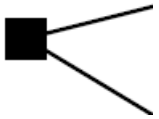
The algorithm proceeds like a planar left-to-right scan. Edges are added to and removed from the status **T**:



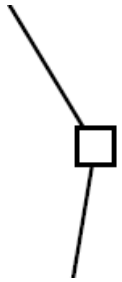
For each edge e_j a corresponding vertex $corr(e_j)$ exists. The corresponding vertices are defined by the algorithm itself while it scans the polygon.

NB: The corresponding vertex of an edge can change during the algorithm execution!

START and END vertices



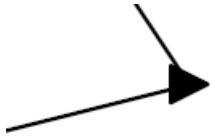
handle_start_vertex(v_i)
{
  Insert e_i in **T** and set *corr*(e_i) to v_i
}



handle_end_vertex(v_i)
{
  if *corr*(e_{i-1}) is a MERGE vertex
    Insert the diagonal connecting v_i to *corr*(e_{i-1})
  Delete e_{i-1} from **T**
}

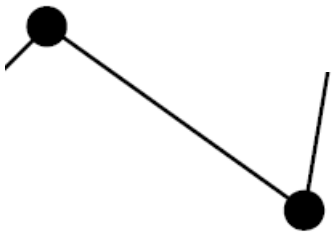
SPLIT and MERGE vertices

handle_split_vertex(v_i)
{
  Search in **T** to find the edge e_j directly below v_i
  Insert the diagonal connecting v_i to *corr*(e_j)
  Set *corr*(e_j) to v_i
  Insert e_i in **T** and set *corr*(e_i) to v_i
}



handle_merge_vertex(v_i)
{
  if *corr*(e_{i-1}) is a MERGE vertex
    Insert the diagonal connecting v_i to *corr*(e_{i-1})
  Delete e_{i-1} from **T**
  Search in **T** to find the edge e_j directly below v_i
  if *corr*(e_j) is a MERGE vertex
    Insert the diagonal connecting v_i to *corr*(e_j)
  Set *corr*(e_j) to v_i
}

REGULAR vertices



handle_regular_vertex(v_i)
{
  if the interior of **P** lies above vertex v_i
    if *corr*(e_{i-1}) is a MERGE vertex
      Insert the diagonal connecting v_i to *corr*(e_{i-1})
    Delete e_{i-1} from **T**
    Insert e_i in **T** and set corr(e_i) to v_i
  else
    Search in **T** to find the edge e_j directly below v_i
    if *corr*(e_j) is a MERGE vertex
      Insert the diagonal connecting v_i to *corr*(e_j)
    Set *corr*(e_j) to v_i
}