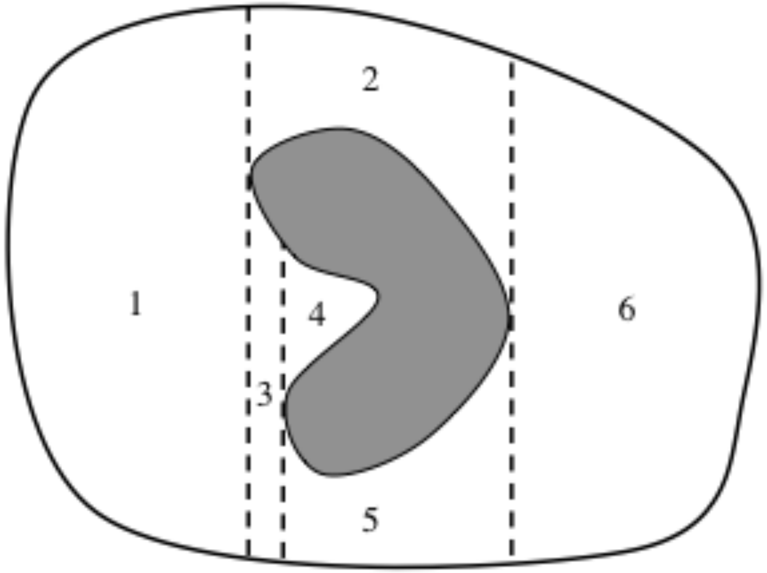
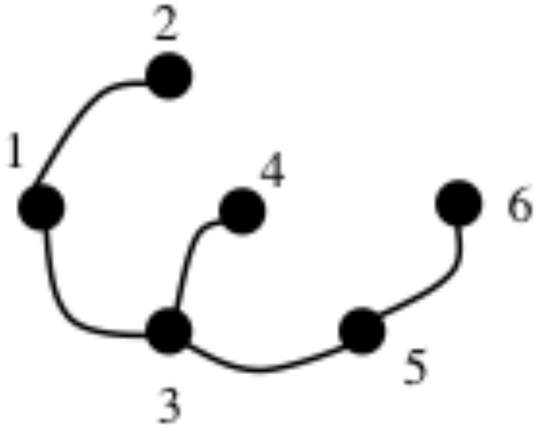
Cell decomposition

- Division of the free space in a set of **cells**.
- Adjacent cells share a **boundary**, and based on this, an **adjacency graph** can be built.
- **Path planning** is done by first determining the cells that contain the start and goal positions, and then finding a path within the adjacency graph. The A* or other graph search algorithms can be used.
- The adjacency graph can also be considered as a **topological map**.
- Cell decomposition is often used for **coverage path planning**.

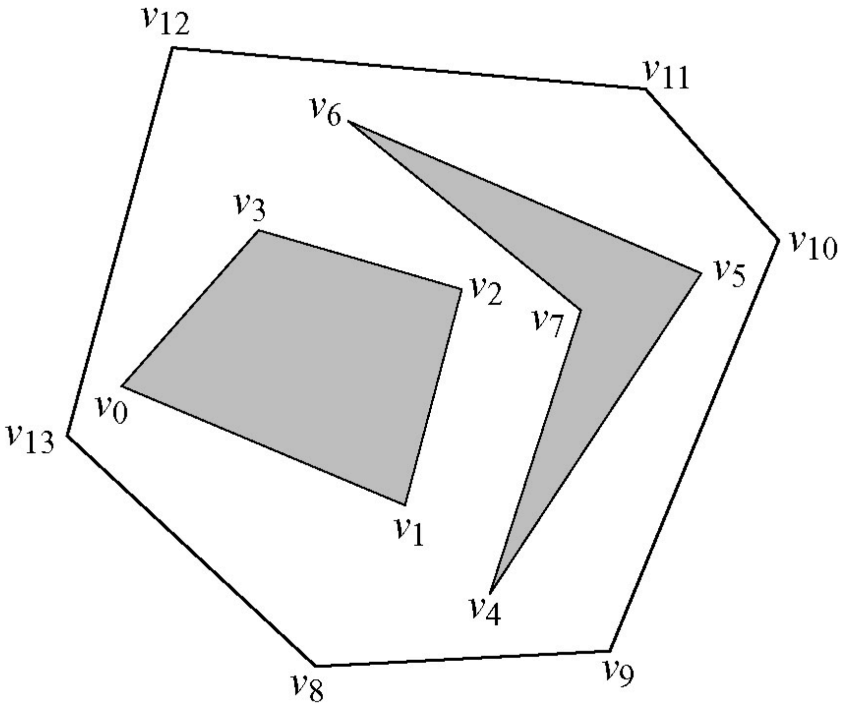




Cell decomposition

Trapezoidal Decomposition

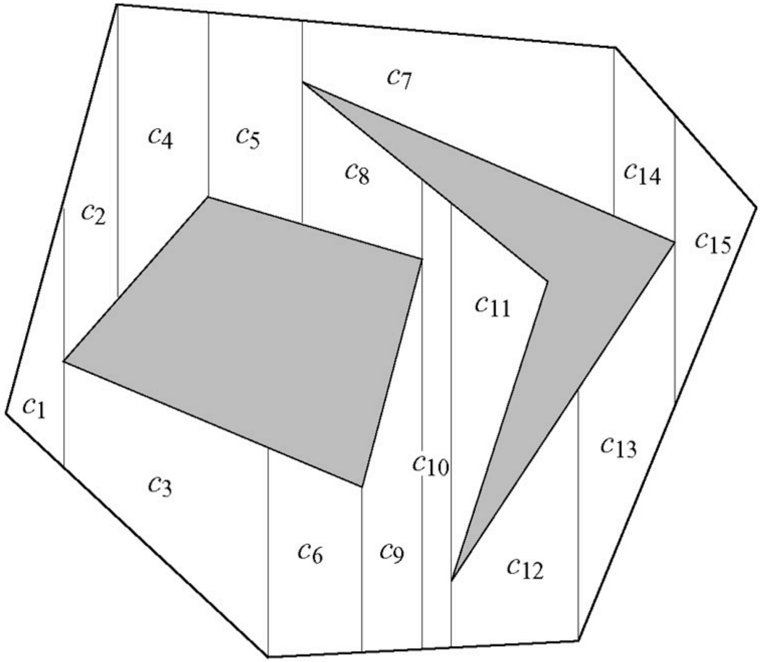
- Cells that are shaped like trapezoids: 4 sides, and also triangles (1 side has 0-length).
- Only for polygonal obstacles, which will have a set of vertices.
- At each vertex v_i, an upper and/or lower vertical edges appear, which will generate the boundaries between adjacent cells.
- At each vertex v_i, the adjacency graph is also updated accordingly.

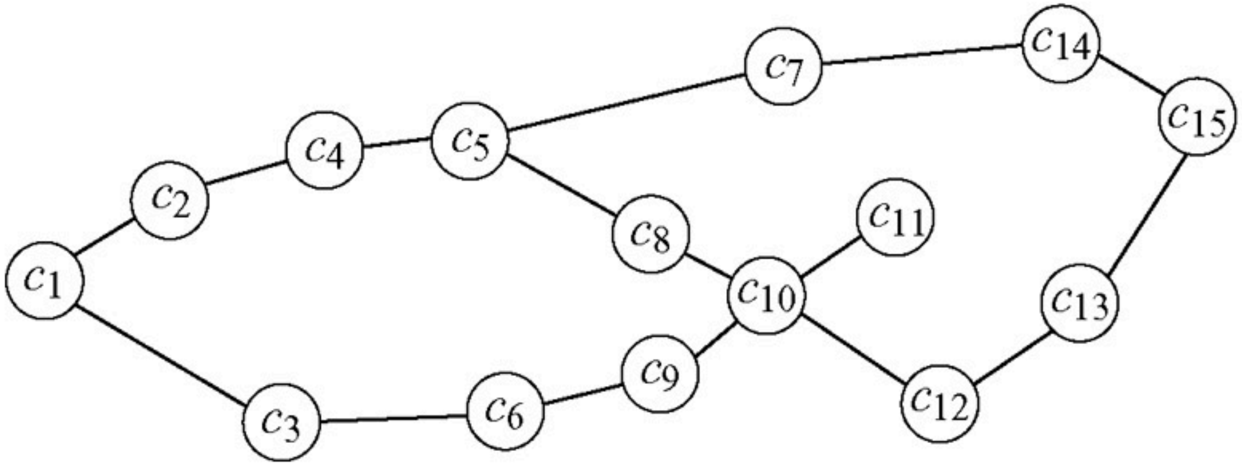


Cell decomposition

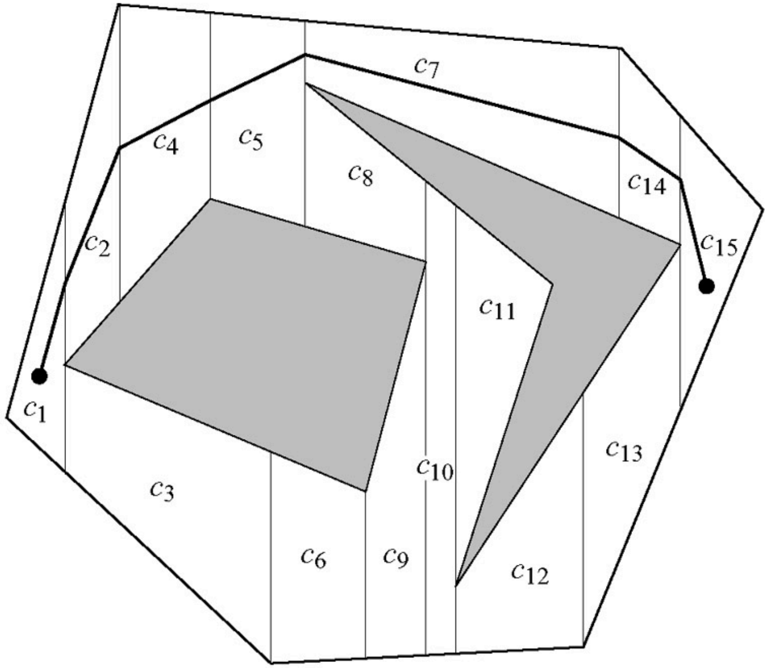
Trapezoidal Decomposition

- Each cell has its corresponding graph node.
- Cells which contain the start and goal positions must be found.
- Planning will take place at the adjacency graph.
- Midpoints will be used to translate the plan found in the graph into the free space.



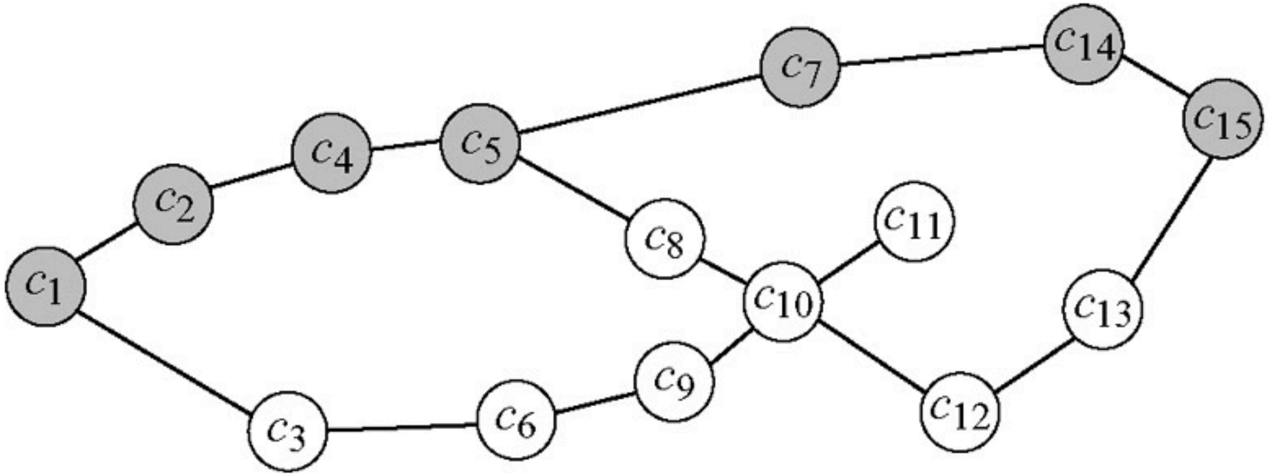


Cell decomposition



Trapezoidal Decomposition

- Each cell has its corresponding graph node.
- Cells which contain the start and goal positions must be found.
- Planning will take place at the adjacency graph.
- Midpoints will be used to translate the plan found in the graph into the free space.

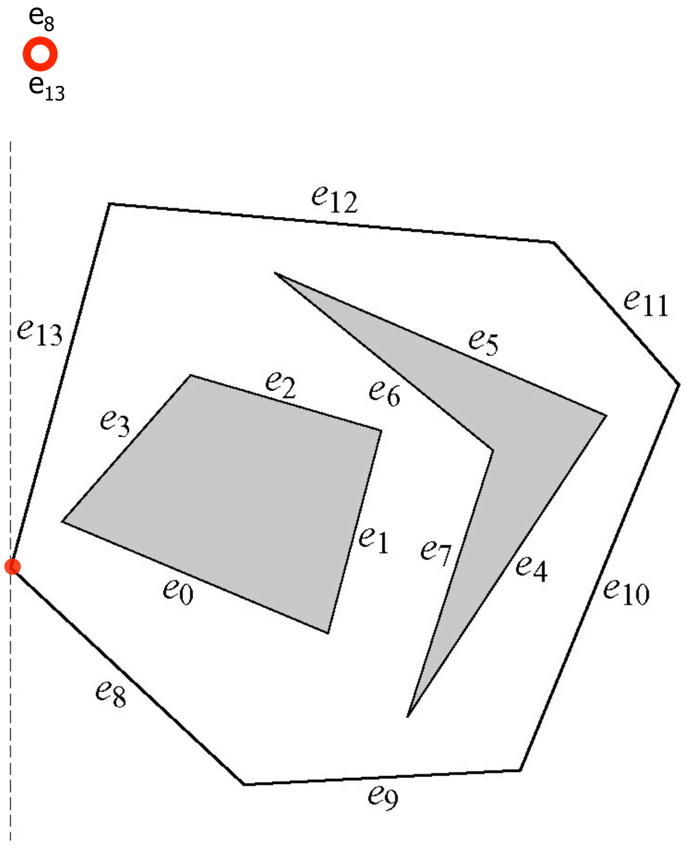


Cell decomposition

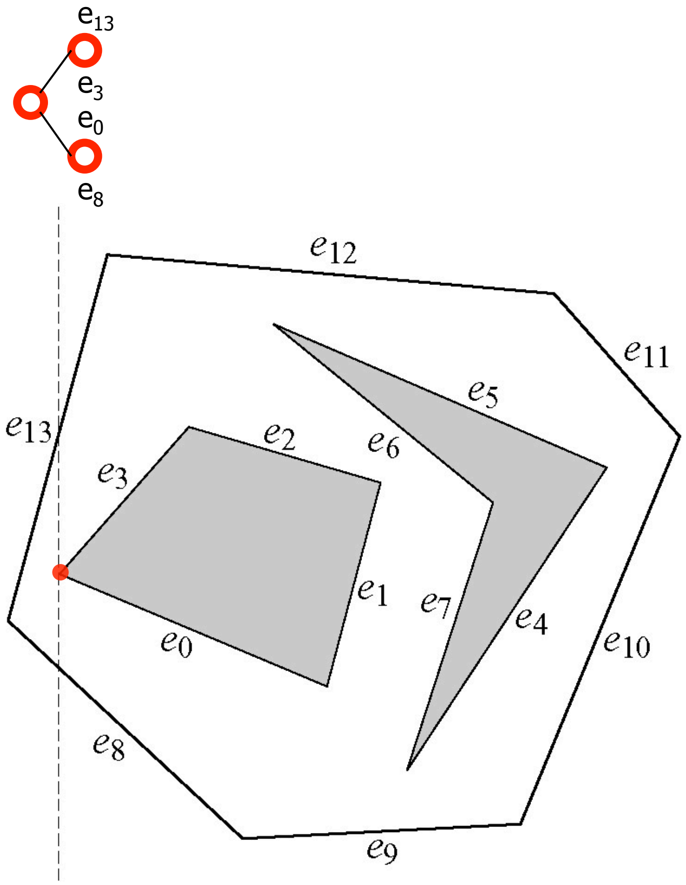
Trapezoidal Decomposition

- In order to build cells, a vertical sweep line from left to right is used.
- All vertices are sorted from left to right.
- The sweep line stops at each vertex v_i and a list L, containing intersected edges, is updated.
- By calculating the y coordinate of the intersection between the sweep line and each vertex contained in L, we can easily know the upper (e_{UPPER}) and lower (e_{LOWER}) edges of the current vertex.
- The update of L is done considering the 2 edges that belong to v_i. If an edge belongs to the list, it is removed; and if it is not in the list, it is added.
- If both are not in L, the second vertex of each edge is used to sort them.
- The edge on the left of the vertex edges in L will be e_{LOWER}, and the edge on the rigth will be e_{UPPER}.

Cell decomposition

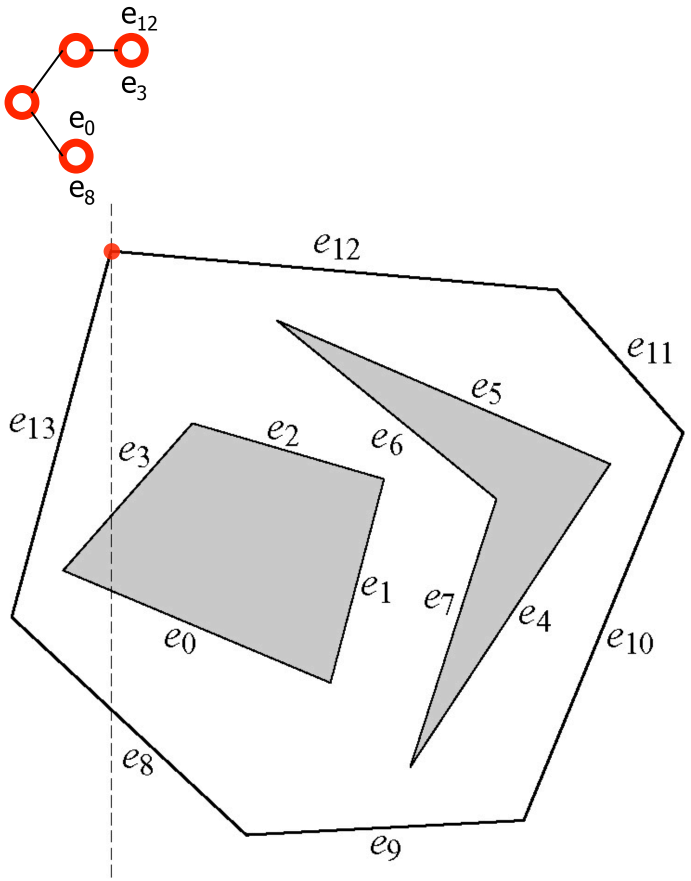


L:$\{e_8,e_{13}\}$
e_{UPPER}: -
e_{LOWER}: -



L:$\{e_8,e_0,e_3,e_{13}\}$
e_{UPPER}: e_{13}
e_{LOWER}: e_8

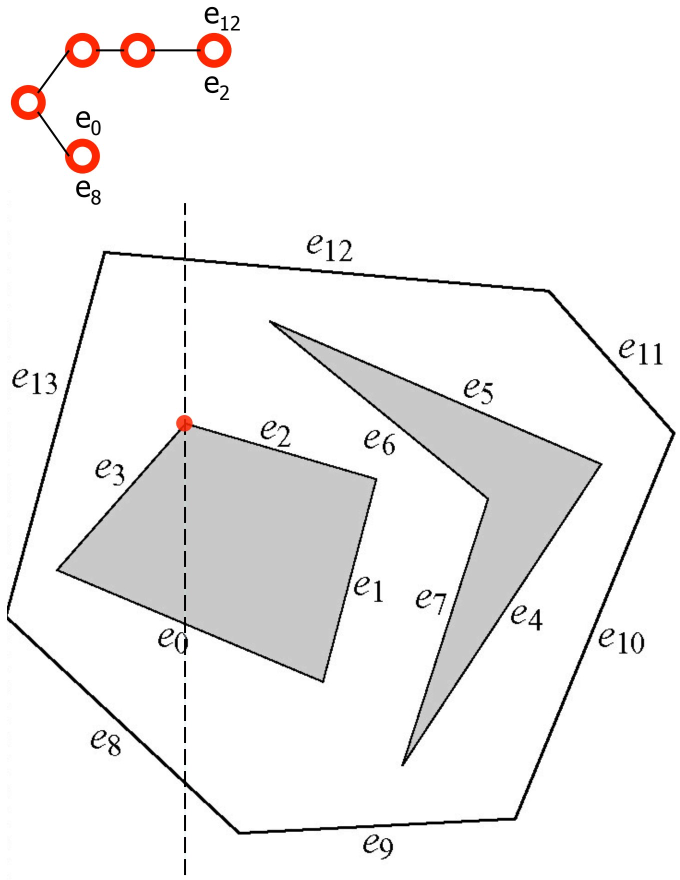
Cell decomposition



L:{e_8,e_0,e_3,e_{12}}

e_{UPPER}: -

e_{LOWER}: e_3

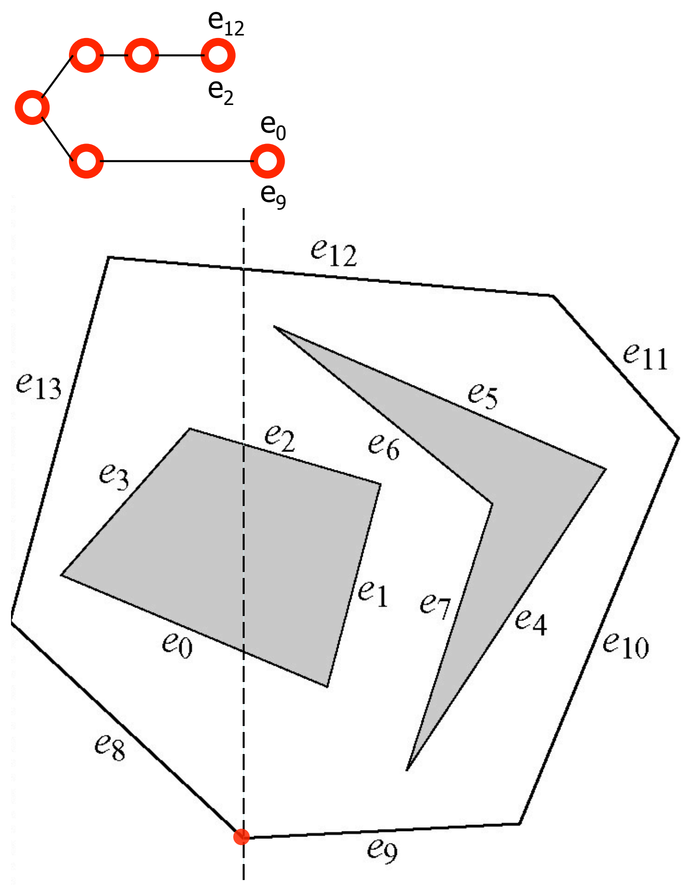


L:{e_8,e_0,e_2,e_{12}}

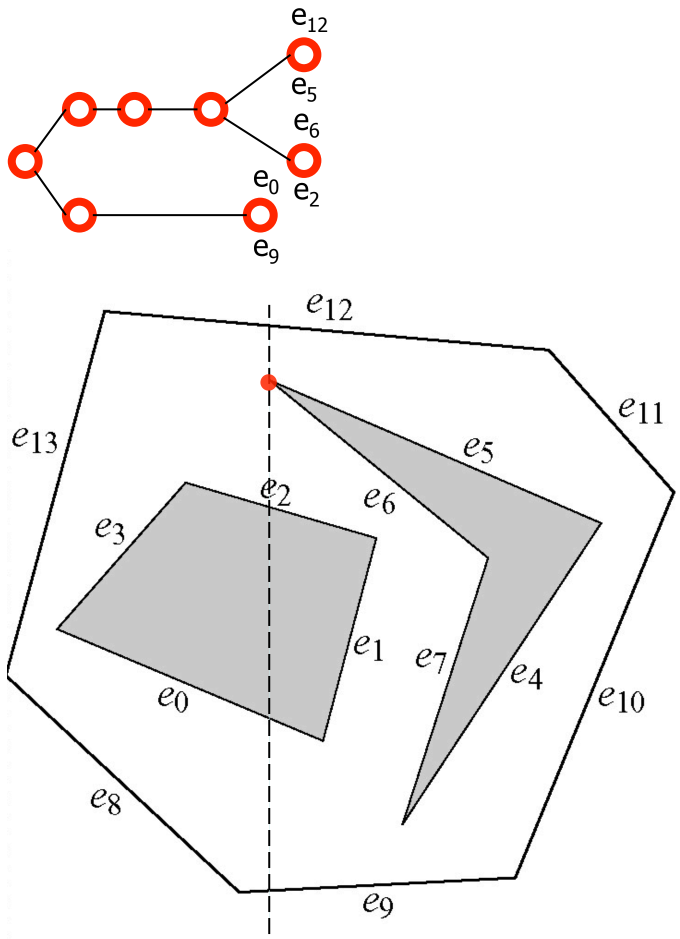
e_{UPPER}: e_{12}

e_{LOWER}: e_0

Cell decomposition

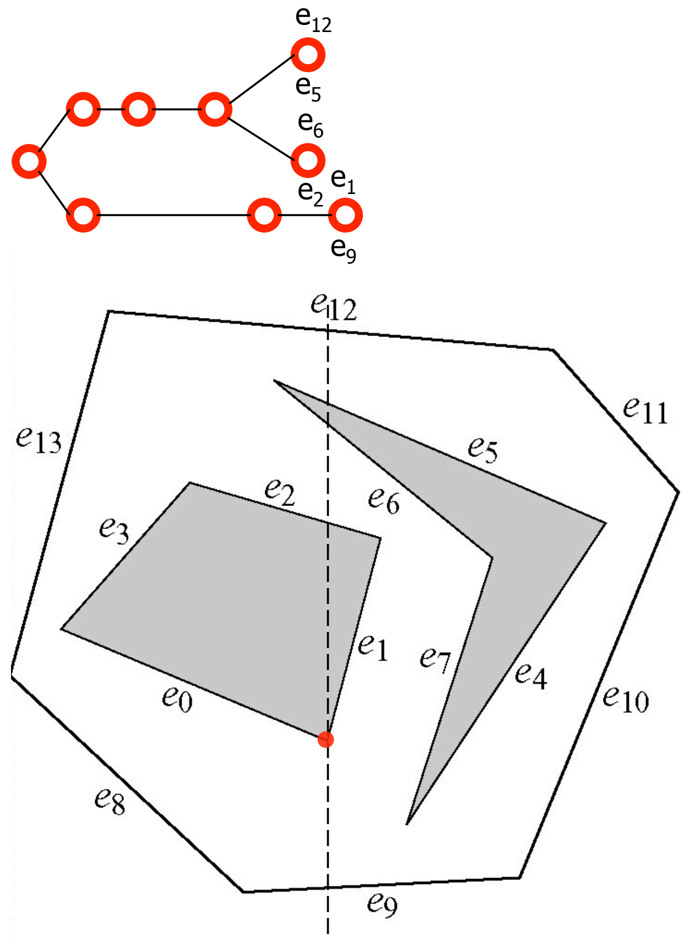


L:$\{e_9,e_0,e_2,e_{12}\}$
e_{UPPER}: e_0
e_{LOWER}: -



L:$\{e_9,e_0,e_2,e_6,e_5,e_{12}\}$
e_{UPPER}: e_{12}
e_{LOWER}: e_2

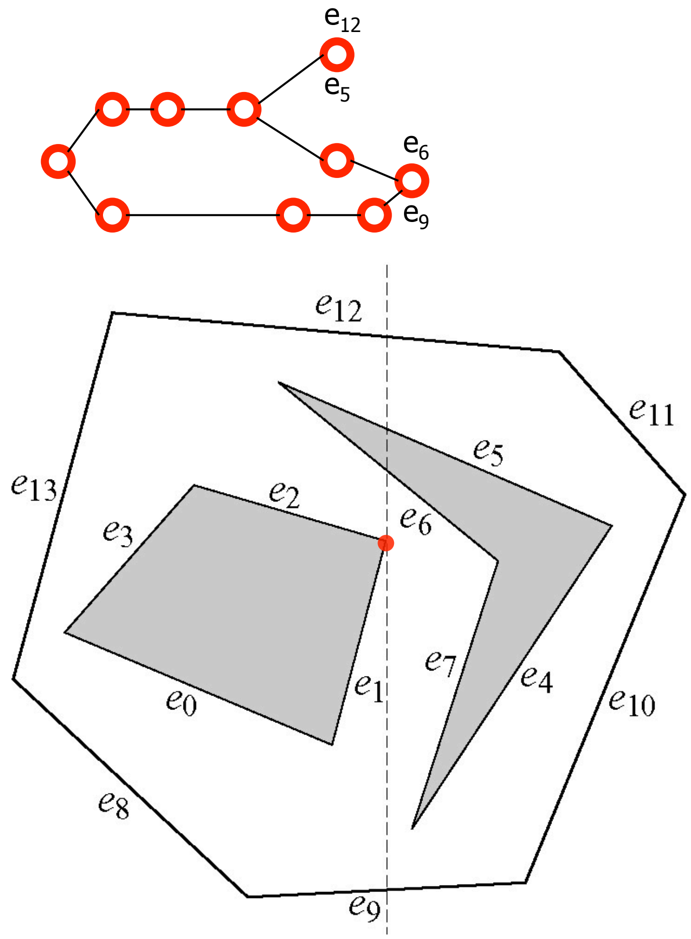
Cell decomposition



L:$\{e_9,e_1,e_2,e_6,e_5,e_{12}\}$

e_{UPPER}: e_2

e_{LOWER}: e_9

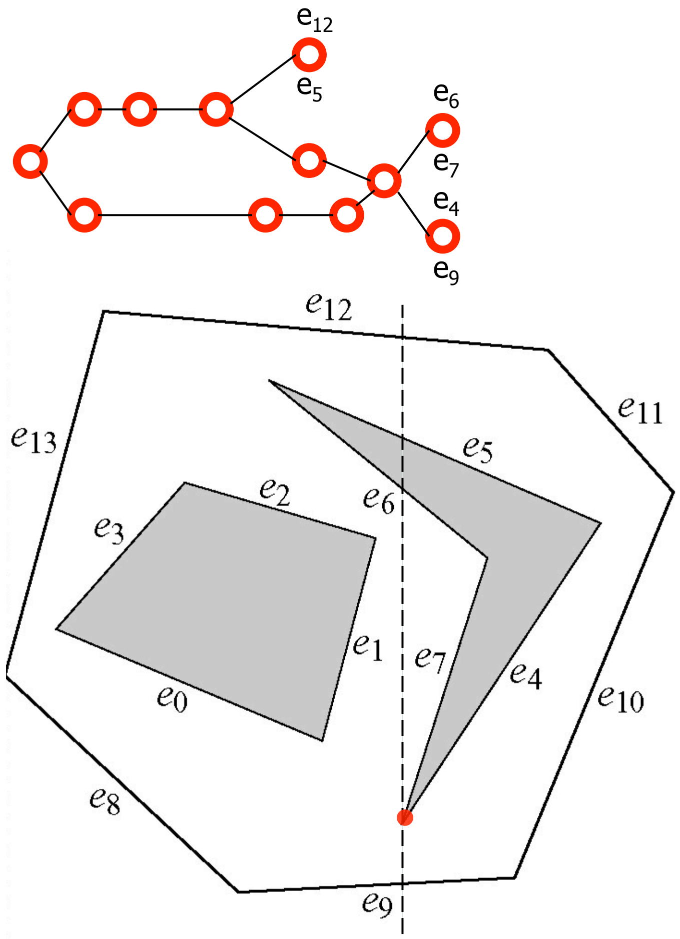


L:$\{e_9,e_6,e_5,e_{12}\}$

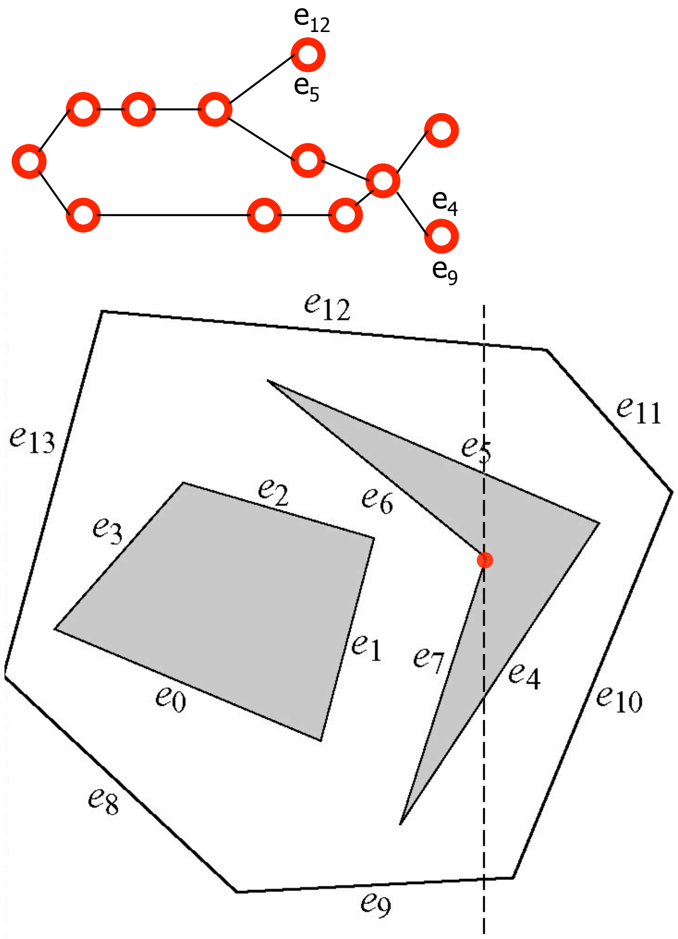
e_{UPPER}: e_6

e_{LOWER}: e_9

Cell decomposition

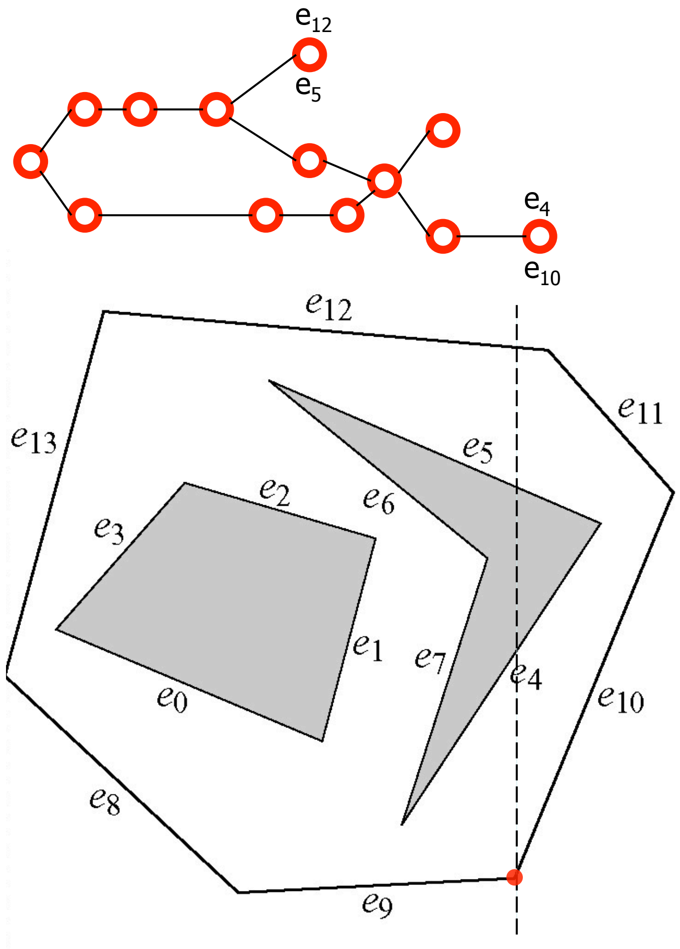


L:{e_9,e_4,e_7,e_6,e_5,e_{12}}
e_{UPPER}: e_6
e_{LOWER}: e_9



L:{e_9,e_4,e_5,e_{12}}
e_{UPPER}: e_5
e_{LOWER}: e_4

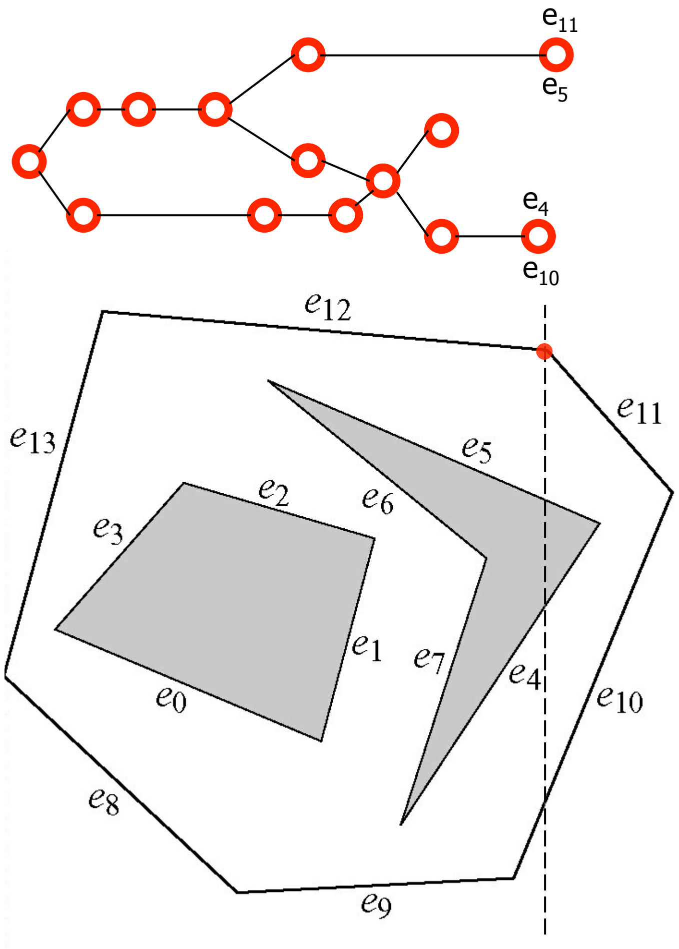
Cell decomposition



L:$\{e_{10},e_4,e_5,e_{12}\}$

e_{UPPER}: e_4

e_{LOWER}: -

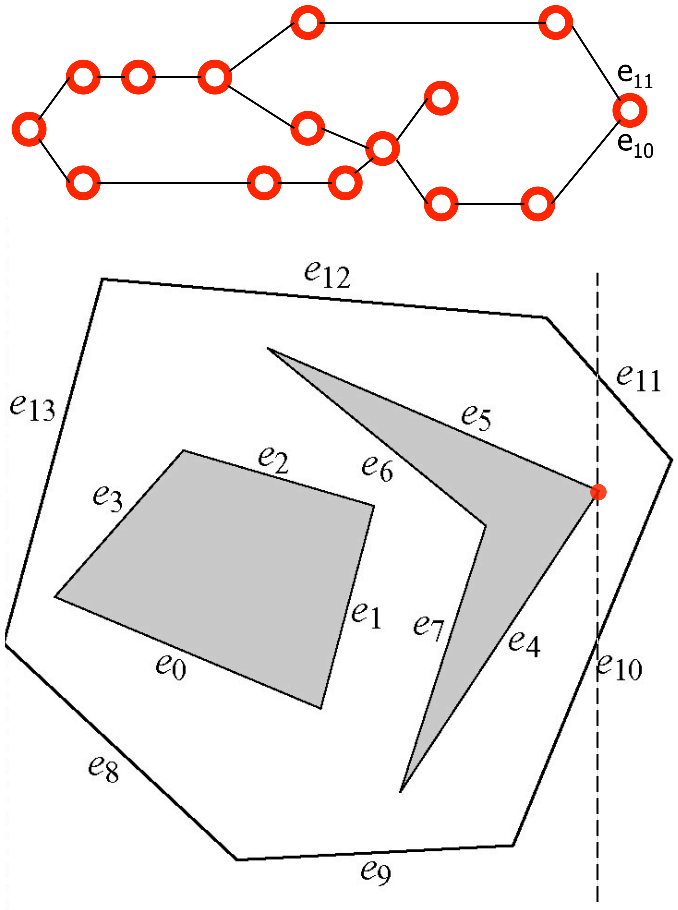


L:$\{e_{10},e_4,e_5,e_{11}\}$

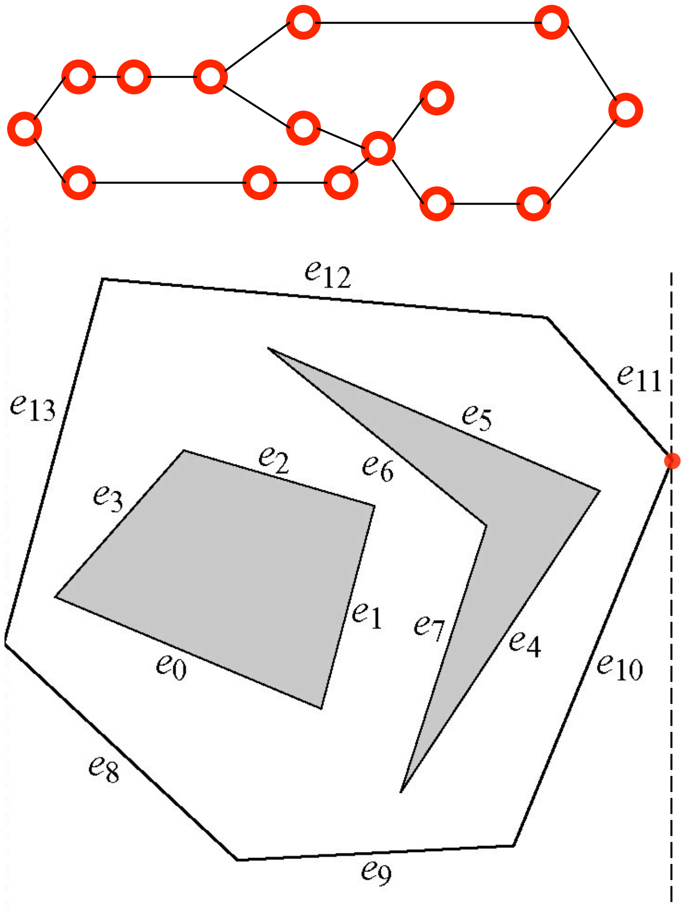
e_{UPPER}: -

e_{LOWER}: e_5

Cell decomposition



L:{e_{10},e_{11}}
e_{UPPER}: e_{11}
e_{LOWER}: e_{10}

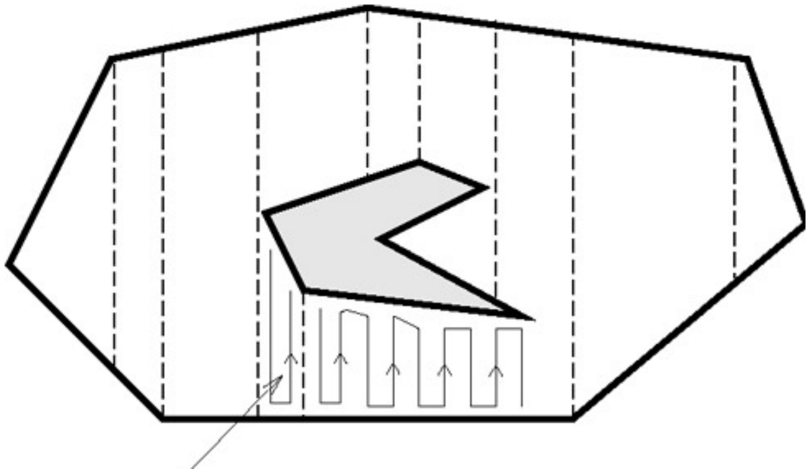


L:{}
e_{UPPER}: -
e_{LOWER}: -

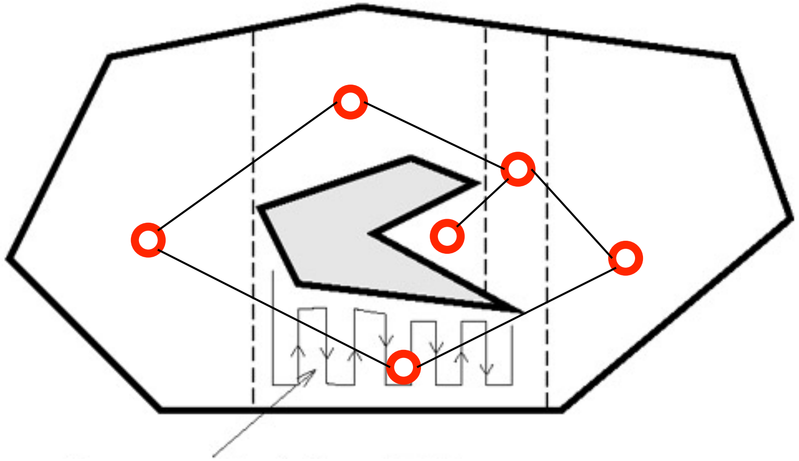
Cell decomposition

Boustrophedon decomposition

- Similar than trapezoidal decomposition but only vertices at which vertical line can be extended up and down are considered.
- Cells are bigger, not trapezoidal.
- Used for coverage path planning (i.e. cleaning robots).
- A lawnmower trajectory is followed inside each cell.



Trapezoidal decomposition

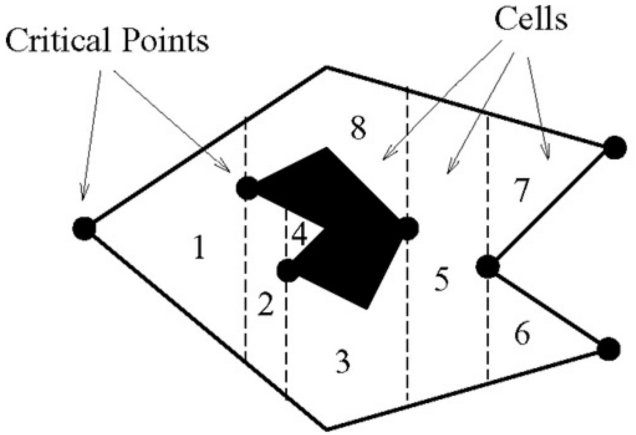


Boustrophedon decomposition

Cell decomposition

Boustrophedon decomposition

- Once the graph is generated, an exhaustive walk is first determined (depth-first search algorithm).
- Then, explicit robot motions are determined within each cell: straight lines separated by one robot width and short segments connecting them.


Critical Points
Cells
1
2
3
4
5
6
7
8