ME 145
Robotic Planning and Kinematics

Lab Session No. 4

BFS ALGORITHM & TRAPEZOIDATION

Instructions

Submit your code and reports through Canvas. These are due on Monday, 13th May, 11:59 pm. No late submissions will be accepted.

Exercise 2.9, Sweeping Trapezoidation, will count towards extra credit.

E2.8 **Programming: BFS algorithm (30 points)**

Consider the following functions:

computeBFStree (15 points)

Input: a graph described by its adjacency table *AdjTable* and a start node *start*

Output: a vector of pointers *parents* describing the BFS tree rooted at *start*

computeBFSpath (15 points)

Input: a graph described by its adjacency table *AdjTable*, a start node *start* and a goal node *goal*

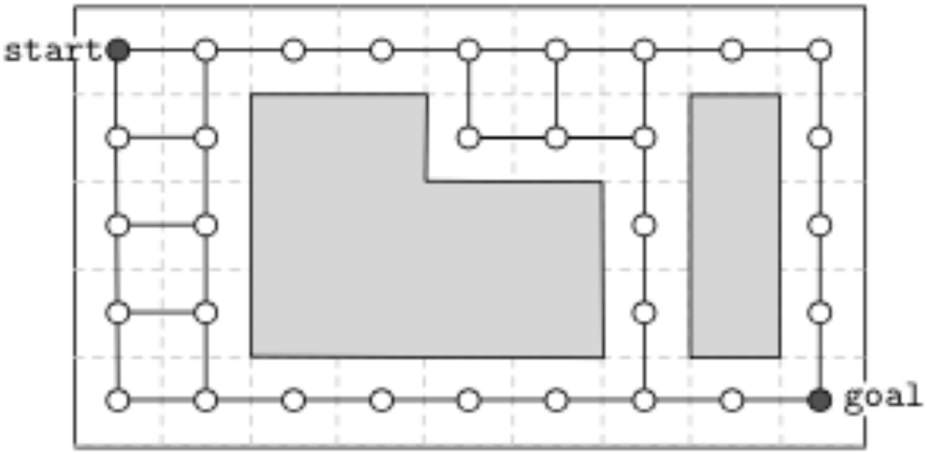
Output: a path from start to goal along the BFS tree rooted at *start*

For each function, do the following:

(i) In your report, explain how to implement the function, possibly deriving analytic formulas, and characterize special cases,

(ii) program the function, including correctness checks on the input data and appropriate error messages, and

(iii) verify your function is correct on a broad range of test inputs and, specifically, on the graph and the start/goal problem drawn in figure.



E2.9 **Programming: The sweeping trapezoidation algorithm (60 points EXTRA CREDIT).**

In this project you will write a function that, given as input a rectangular workspace with multiple polygonal obstacles (each described by a counter-clockwise sequence of vertices), computes the trapezoidal decomposition of the workspace and the associated roadmap graph. For simplicity, assume that the polygonal obstacles do not overlap and are strictly inside the rectangular workspace.

(i) Write in detailed pseudocode a function that classifies each obstacle vertex in the 6 possible types. (10 points)

(ii) Implement the main function (50 points):

SweepingTrapezoidation

Input: a rectangle *W* with polygonal holes *L*, where *L* is a list of non-overlapping polygons inside *W*.

Output: a collection of trapezoids (including the degenerate case of triangles), and a collection of edges (each describing a shared side between two trapezoids).