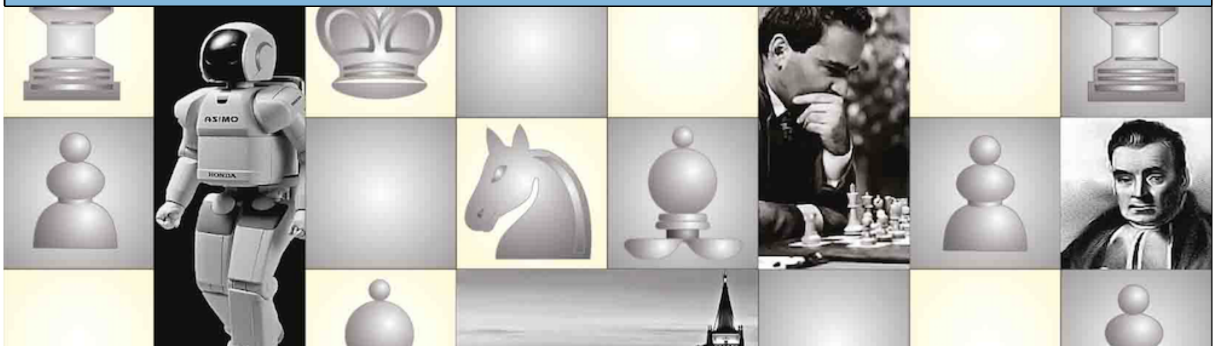
Fall 2024 - CSC545/645 Artificial Intelligence - Assignment 3



Due date: Thursday, September 19, 2024, 2 pm. Please create a folder called `assignment3` **in your local working copy of the repository and place all files and folders necessary for the assignment in this folder. Once done with the assignment, add the files and folders to the repo with** `svn add` *`files,folders`* **and then commit with** `svn ci -m "SOME USEFUL MESSAGE"` *`files,folders`*.

Exercise 3.1
Read chapter 3.5 – 3.6 (informed search strategies and heuristic functions) of the textbook.

The **heuristic path algorithm** is a best-first search in which the objective function is $f(n) = (2-w)g(n) + wh(n)$. For what values of w is this algorithm guaranteed to be optimal? What kind of search does this perform when $w = 0$? When $w = 1$? When $w = 2$?
[4 points]

Exercise 3.2
Consider the route finding problem between 2 points in a plane that has convex polygons as obstacles (see figure 1). Find the shortest path. This is an idealistic navigation problem for a robot (e.g. navigating through a museum or library).
[16 points in total]

1. Suppose the state space consists of all positions (x, y) on the plane. How many states are there? How many paths are there to the goal?
   [2 points]

2. Explain briefly why the shortest path from one polygon vertex to any other in the scene must consist of straight-line segments joining some of the vertices of the polygons. Define a good state space now. How large is this state space?
   [2 points]

3. Define the necessary functions to implement the search problem, including a successor function that takes a vertex as input and returns the set of vertices that can be reached in a straight line from the given vertex. (Also think about the neighbors on the same polygon.) Use the straight-line distance for the heuristic function.
   [2 points]

4. Implement and apply one or more of the classical search algorithms that we have discussed in class to solve a range of problems in the domain, and comment on their performance. A* should be among them. CSS645-students: you need to use at least three such as A*, Greedy, and Uniform-Cost search. Make sure that the implementation is independent of the specific domain. This means that the same algorithm should be able to solve other variants of the

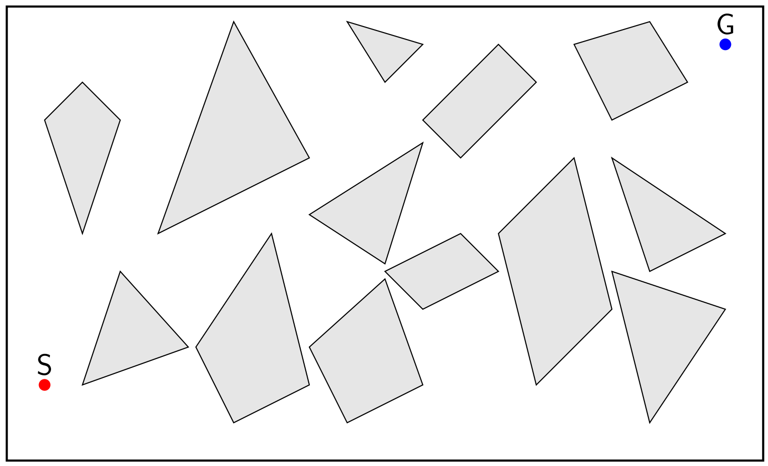


Figure 1: A scene with random polygonal obstacles in a 2D plane. S and G are the start and goal states.

problem (e.g. the polygons in the plane are different). You may want to use the framework provided by Michael Davis (download framework for Java, C++, Python).
[10 points]