MISSION #8

Sequential Planning with Certainty

Due: August 26, 2010

Introduction:

With the conditional plan you created by hand last few weeks, you got a taste of what planning is all about. In this lab, you will develop full-fledged planners. First you will write a sequential planner for when the robot has complete information about its position and about the environment. Next, you will have to deal with the possibility that the robot doesn't know its starting position, giving rise to sequential planning with uncertainty. Finally, you will develop a conditional planner, which is unique because the plan it generates contains branches. The common thread here is that all of these programs return plans.

For all of the assignments in this lab, we will test your code by placing the robot in a real environment, giving you the map and a problem and watching your robot solve the problem, physically!

Sequential Planning with Certainty:

In the easiest case, the robot has complete information about the world: it has a correct map of the environment and it knows its initial position and orientation. A sequential planner accepts a specification of this knowledge and is then capable of creating a sequential plan that, if executed, would take the robot from the initial position to the goal position.

Assignment 5.0: Sequential Planning with Certainty

Task: Write a robot function called ***SPC(robot position)*** *that returning a plan*

Description: Write a sequential planner that finds and executes sequential plans when initial conditions are specified with certainty. You may assume that the map is complete and that the robot position specification will be a singleton state set.

NOTE: this must be a node-marking BFS solution. (see last page)

For this lab, we consider the commands GTNN(1), (TurnTo 90), (TurnTo -90) to be atomic (i.e. the building blocks of plans, for example G= GTNN(1), R=(TurnTo 90), and L=(TurnTo -90)). Therefore, the length of a plan is the number of atomic actions it contains.

Although a plan can only contain the atomic actions described above, you are free to optimize your plans by combining strings of GTNN's into single GTNN calls. But remember, the concept of ***maxDepth*** refers to the specific maximum length of the un-optimized plan, not the optimized one!

Breadth-First Search (BFS)

Expand shallowest unexpanded node

- Implementation: