AI for Robotics II
Tutorial: Task and Motion Planning

1 Introduction

The aim of this class is to familiarize you with modeling integrated task and motion planning problems. Kindly note that there exists different approaches to integrate task planning and motion planning and what we discuss here is one such approach.

Let us consider a coffee shop scenario where a waiter robot need to serve the orders in an optimal way (example, distance travelled). This might require choosing a certain sequence of tables to deliver the orders. Thus, deciding the discrete sequence of table visits require reasoning about the distance traversed by the robot. Keeping this scenario in mind, we will model a very abstract version of the same wherein we consider a robot starting at a certain region in the environment and has to navigate to four different regions. A region, for example correspond to a coffee table. In reality the robot can be anywhere around the coffee table while delivering the order. However for the sake of brevity of we associate a single way-point (location and orientation considered here) to a region. Thus a task of delivering to a table geometrically correspond to the robot going to the way-point associated with that table.

The first step is to set up the popf-tif planner, the details of which can be found here- https://github.com/popftif/popf-tif. Make sure that the planner is running without any errors on your laptop. Once this is done, extract the domain and module folders given along with this tutorial. The domain folder contains a PDDL domain file consisting of a single action and a corresponding problem file where a robot has to visit four regions. The REGION_POSES file contain the mapping from a region to its corresponding way-point. The WAYPOINT.TXT file contain the geometric way-point locations for the four regions to visit as well as the starting region. The LANDMARK.TXT file contain landmarks to localize the robot (please ignore for now).

The module folder contains the external module files. Let us take a look at the domain file given. The functions *triggered, act-cost, dummy* are the ones that involves the external module or semantic attachment part. The results of the computations in the external module part are stored in the *dummy* variable and this is assigned to *act-cost*. The function *triggered* is implemented so that the task planner can communicate the start and goal region to the ex-

ternal module. These can be comprehended upon careful examination of the **loadSolver**, **callExternalSolver** and **calculateExtern** functions within the VISITSOLVER.CPP file in the module folder. The instructions to build the external module can be found in the BUILDINSTRUCTION.TXT file. Once built, the planner can be run using the command (refer- https://github.com/popftif/popf-tif)

```
../popf3-clp -x dom1.pddl prob1.pddl
                    ../visits_module/build/libVisits.so region_poses
```

If the external module is build correctly, the planner should run without any errors. The variable *dummy* is randomly given a value of 2 and the cost returned by the planner should therefore be 8. You can check its working by varying the value in **calculateExtern** function. Study the domain, problem files (in the domain folder) and the files MAIN.CPP, VISITSOLVER.CPP. This should give you sufficient understanding regarding the external module implementation.