Automated Planning - Lab 6

Victor Tranell Erik Linder-Norén

2017-05-16

In this report the following two environments and robots will be used:

- Environment 1: Maze, robot: hook_planar_robot.dae
- Environment 2: Barriers, robot: barriers_easy_robot.dae

Planner 1: PRM

This is a probibalistic planner that randomizes states along the available state space. Creates roadmaps around milestones which leads to a high granularity map along obstacles.

Observations: Since it explores around obstacles the path will be very good around them. However, not all obstacles must be avoided since they are not along the best paths and therefore PRM does not compute the best paths and is generally pretty slow.

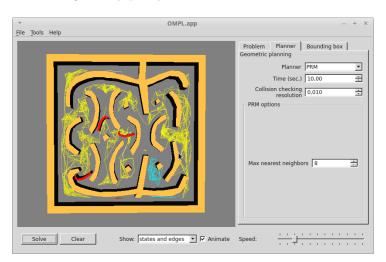


Figure 1: PRM, Maze

```
max_nearest_neighbors =
Start states:
Compound state {
Real/vectorState [-0.91-0.24]
SO2State [0]
]
Goal state, threshold = 1e-06, memory address = 0x279cf70, state =
Compound state {
Real/vectorState [-31-30]
SO2State [0.349066]
]
Average state cost: 0
There are 0 solutions
Starting with 2 states
Created 649 states
Solution found in 1,510653 seconds
SimpleSetup: Path simplification took 0,028901 seconds and changed from 20 to 65 states
Interpolating solution path to 66 states
```

Figure 2: PRM, Maze log file

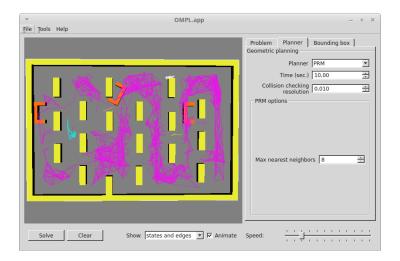


Figure 3: PRM, Barriers

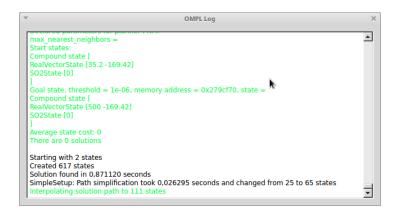


Figure 4: PRM, Barriers log file

Planner 2: RRTConnect

RRTConnect uses two RRT searches, one from the start position and one from the goal and tries to connect them. An RRT search is performed by randomly placing state on the map and then connects it to the closest state already explored until the goal state is found.

Observations: This is generally very quick and finds good solutions. This is due to the fact that it starts both from the goal and start state and only expands states that are closer to the current node than any previously explored state.

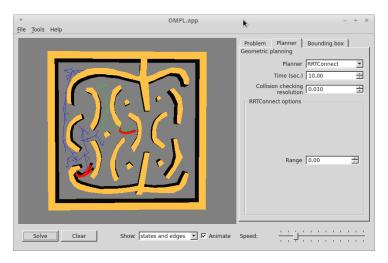


Figure 5: RRTConnect, Maze

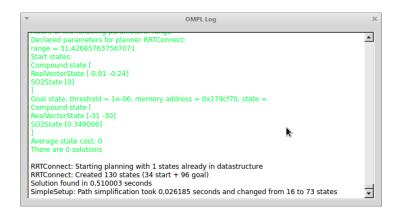


Figure 6: RRTConnect, Maze log file

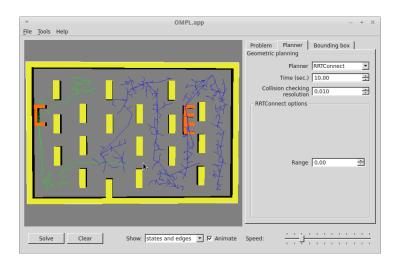


Figure 7: RRTConnect, Barriers



Figure 8: RRTConnect, Barriers log file

Planner 3: SPARStwo

Places randomly uniformly distributed states among the entire search space and tries to connect it between each other to generate a roadmap. Since it doesn't consider distance between the connected states it is not very reliable in finding good paths. Since the points are so sparsly distributed it only considers a small amount of states.

Env1. This is very similar to the PRM in the same environment since the PRM expands around obstacles and the whole map consist of obstacles and therefore sparstwo and PRM has similarly sized search spaces. However it is far from optimal.

Env2. Since this map is very big the available search space is huge which leads to it taking a long time as sparsetwo selects randomly over the whole map. However it is far from optimal.

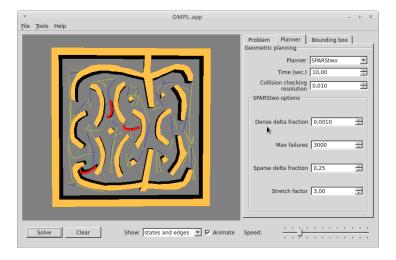


Figure 9: SPARStwo, Maze

```
Compound state [
RealVectorState [-0.91 -0.24]
SO2State [0]
]
Goal state, threshold = 1e-06, memory address = 0x279cf70, state =
Compound state [
RealVectorState [-31 -30]
SO2State [0.349066]
]
Average state cost: 0
There are 0 solutions
SPARStwo: Starting planning with 2 states already in datastructure
SPARStwo: Created 133 states
Solution found in 2.077313 seconds
The solution path was slightly touching on an invalid region of the state space, but it was successfully fixed.
SimpleSetup: Path simplification took 0.039098 seconds and changed from 19 to 61 states Interpolating solution path to 98 states
```

Figure 10: SPARStwo, Maze log file

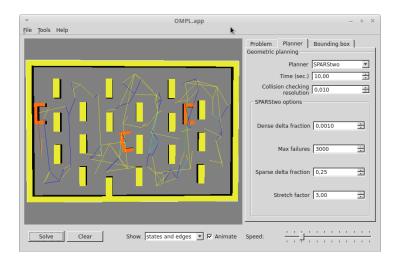


Figure 11: SPARStwo, Barriers



Figure 12: SPARStwo, Barriers log file