

Assignment 5

CSCI-4290/6290: Robot Motion Planning

Due: Tuesday, November 4, 2003

Assignments are due on Tuesday, November 4, and are to be done individually. See the course syllabus for the late submission policy.

In this assignment, you will use the Motion Strategy Library (MSL) to experiment with probabilistic motion planners and observe their performance on different tasks. This assignment is intended to familiarize you with MSL, so you can use it to aid your implementation of your final project.

1. MSL has several nice features. MSL supports a set of planners, most of which are based on rapidly-exploring random trees (RRTs) and probabilistic roadmaps (PRMs). It also interfaces to the PQP collision detection software, provides graphics support to view the generated paths, permits you to change the viewpoint, plot search graphs, save images, etc.
2. MSL is available in the directory `/projects/rmp/msl-2.0` on the CS Solaris machines in Amos Eaton. Read the `README` and note the executable file `plangl`. To bring up the MSL interface on the example problem specified in the subdirectory `data/cage`, use the command `plangl data/cage`.
3. For an introduction to MSL, go to the MSL webpage `msl.cs.uiuc.edu/msl/`. Also read the `README` files in the `/projects/rmp/msl-2.0` directory and its subdirectories. Of course, the source code in the `src` subdirectory is useful too.
4. You should test the planners on the following examples in the `data` subdirectory: `cage`, `wrench`, `boor`, `multi2`, `car2`. For each of these, identify the planner (and any related settings) that most effectively generates a solution, and document the time taken, the number of nodes generated, the number of collision checks, and any other relevant information. If you are unable to find a plan, indicate at what stage you terminated the planning.

In addition to the above information, you must hand in plots of the generated search graphs for any two of the above examples that you find interesting. Clearly identify the variables being plotted. You can generate these plots using the “plot” option from the MSL menu.

5. The subdirectory `tests` has examples showing how you can create your own motion models and extend MSL. You can use the examples in this directory (and the `model2d.*` and `model3d.*` files in the `src` directory) to create example robot motion models. You can use the examples in the `/projects/rmp/msl-2.0/data` directory to create example environments with obstacles and robots that you specify. (You may create examples that you can use for your final project.) The `Makefile` in the `tests` subdirectory can be used as a template to easily compile your examples.

6. Create a model of a robot and an example environment with obstacles, and identify the planner and any related settings that best solves this. The example you create may be in a 2D or 3D world, and may be a simple free-flying robot, an articulated robot, a nonholonomic robot, or anything you'd like to experiment with. Start with something simple first, get it working, and then make it as complex as you like. Your grade for this part of the assignment will be based on how "interesting" your example is.

You should submit your example model and data files, much like the `tests` directory, with a `README` file describing your models and data files, any special features that make it interesting, and compilation instructions.

It is recommended that you use C++ to write any code. Your program will be tested on the Solaris (UNIX) workstations in the computer labs in Amos Eaton 215 and Amos Eaton 217.

Submission

The planner information for the examples (item 4 above) and related data must be submitted at the beginning of class on Tuesday, November 4.

The rest of your assignment (item 6 above) must be submitted no later than 11:59 pm on November 4, 2003. **You are responsible for ensuring that any code you submit can compile and run on the Sun Ultra10s in the Sparc Lab (Amos Eaton 217) and in the SunRay lab (Amos Eaton 215).**

You must hand in your source code (source and header files), a Makefile to compile it, and the examples you created. Also include a `README` file with the following information: your name, email address, instructions on how to compile the code and run it, an overview of your examples and implementation, interesting features of your example, known bugs or limitations, and any other relevant information. You will follow the same web-based submission procedure you used for Assignment 2.

Please check the course web page frequently for announcements and additional information on the assignment.