The University of Queensland
Semester 2 2015
COMP3702/7702 Artificial Intelligence – Assignment 1

IMPORTANT Notes:

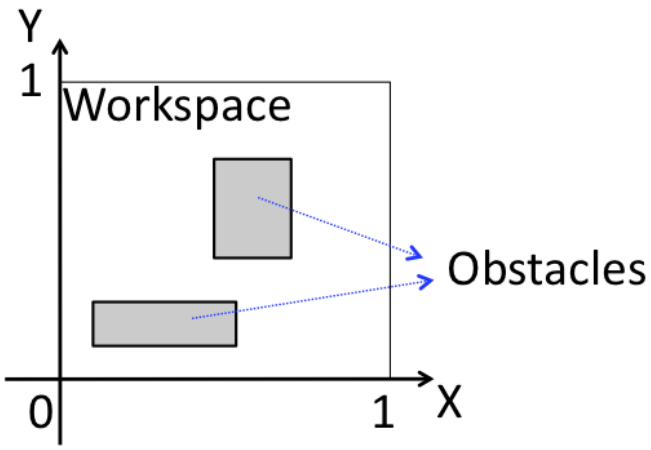
- Due: 9 September 2015, 23:59
- You can work in a group of at most 3 students, but all students in the group must be enrolled in the same class, i.e., all 3702 students or all 7702 students. If you work in a group of 2 or 3 students, please register your group in `http://bit.ly/1IyI8z8` **before 20 August 2015, 23:59**.
- Your program should compile from command prompt (i.e., please use ant, make, or cmake) and generate an executable that can be run on command prompt as:
  `> a1-3702 inputFileName outputFileName` for 3702 students OR
  `> a1-7702 inputFileName outputFileName` for 7702 students.
- Your report should be in a .pdf format and named:
  `a1-3702-studentID1[-studentID2-studentID3].pdf` for 3702 students
  OR
  `a1-7702-studentID1[-studentID2-studentID3].pdf` for 7702 students.
- You should submit a .zip of the entire of your source code and report via turnitin before the due date [need to check if 2 files can be uploaded].

Congratulations!!! You have been hired as an engineer at QUPnF Inc. To increase productivity, QUPnF Inc. has recently purchased a new robot (shown in Figure 1). Your job is to design and develop a software to control this new robot. However, to ensure that you will not damage the new robot, your boss asked you to first develop the software for a simplified version of the robot. This assignment is about designing and developing the program for this simplified robotics system. The following sections provide the specifications of the simplified system and the exact problem you need to solve.

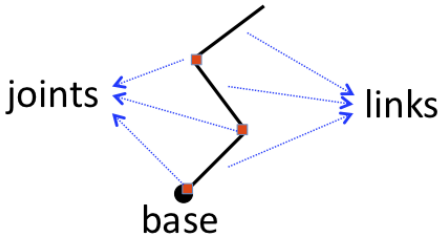
Figure 1: The robot recently purchased by QUPnF Inc. Source: `http://www.neobotix-robots.com/mobile-manipulator-mm-500.html`

Description of the simplified robotic system

The simplified robot is a planar robot, which means it operates in a 2D workspace (rather than 3D). In particular, the 2D workspace is a plane, represented as $[0, 1] \times [0, 1] \subset \mathbb{R}^2$, and is populated by rectangular obstacles. The exact dimension and position of each obstacle in the environment is known prior to execution. Figure 2(a) illustrates this environment.



(a) A 2D workspace.



(b) The simplified robot with 3 links and 3 joints.

Figure 2: An illustration of the simplified system.

The robot consists of two parts: A base and an arm (illustrated in Figure 2(b)). The base can be considered as a point object. A local coordinate system is attached to the base. The origin of this coordinate system is at the position of the base, while the X and Y axis are parallel to the X and Y axis of the workspace.

The robot's arm forms a chain and consists of n links and n joints, where $n \in \mathbb{N}_0$. Each link is essentially a line segment attached to a joint. We number the joints and links sequentially, i.e., joint-1 is located at the base, link-1 is a line-segment attached to joint-1, joint-2 lies at the other end of link-1, etc. Each joint is a rotational joint. A local coordinate system is attached to each joint. The origin of this coordinate system is the position of the joint. For joint-1, this coordinate system coincides with the coordinate system of the base. For other joints, the X axis is the line that coincides with the previous link. Each joint is a rotational joint, which means it can only rotate. We define the joint angle of joint-i as the angle between link-i and the X axis of the coordinate system attached to joint-i. The joint angle of each joint is limited to be within $[-150^0, 150^0]$. Figure 3 illustrates the rotational joints. Each link is of size 0.05 unit length.

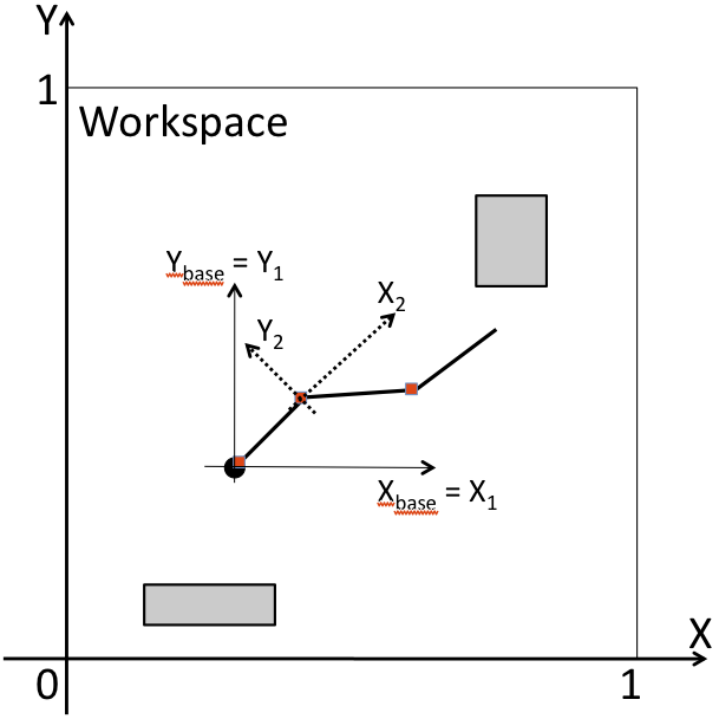


Figure 3: An illustration of the local coordinate systems, $(X_{\text{base}}, Y_{\text{base}})$, (X_1, Y_1), and (X_2, Y_2) are the coordinate systems attached to the base, joint-1, and joint-2, respectively.

What the program should do

Given the initial and goal configurations of the robot, as well as a map of the environment, the program must find a valid path from the initial to the goal configurations. A valid path means that when the robot executes the path, it will satisfy the following requirements:

1. The path consists of primitive steps. In each primitive step, the base of the robot cannot move more than 0.001 unit each joint angle cannot move more than 0.1^0.
2. It will not collide with any of the obstacles.
3. It will not collide with itself. This means there should not be any intersection between two or more links.
4. It will move inside the workspace.
5. The angles of all joints are within the valid range of joint angles (i.e., $[-150^0, 150^0]$ as described in the previous section).
6. Since a primitive step is very small, it is sufficient to satisfy requirements 2-5 at the beginning and end of each primitive step.

Input and output format

Before describing the format of the input and output of the program, we need to first describe the format of a configuration. This format will be assumed/used whenever the program needs to read/write a configuration.

Format of a configuration: A configuration is represented by $n + 2$ real numbers, where n is the number of joints of the robot. Each number is separated by a white space. The first two numbers are the position of the robot's base in the workspace. The subsequent n numbers are the joint angles in sequential order (i.e., the third number is the joint angle of joint-1, the fourth number is the joint angle of joint-2, etc.). Each joint angle is defined in terms of the angle (in radian) with the X axis of the coordinate system attached to the joint.

Input format. The program you develop should accept an input file. The file contains all inputs, i.e., the initial and goal configurations, and the obstacles position and dimension. The format of the input file is as follows.

1. The file consists of $k+3$ lines, where k is the number of obstacles in the environment.
2. The first line is the initial configuration.
3. The second line is the goal configuration.
4. The third line is the number of obstacles in the environment.
5. Each line in the next k lines represents an obstacle and consists of 4 real numbers. The first two numbers represent the X and Y position of the upper-left vertex of the rectangle, while the last two represent the X and Y position of the lower-right vertex of the rectangle.

Output format. Your program should output the path to a file with the following format.

1. The file consists of $m+2$ lines, where m is the number of primitive steps in your path.
2. The first line is the number of line-segments.
3. The second line is the initial configuration.
4. The next m lines are the end configuration of each primitive step.

Examples of the input and output files are in the accompanying supporting software.

What you need to do

Your tasks are:

1. Design the solution. This task contains two main components, i.e., framing the problem as a search problem and deciding what search method to use.
2. Implement your design using high level programming language to solve the problem. Your program solves the problem if it finds a path in under 1 minute on a PC with the same specification as those in the tutorial rooms. Note:
   - You are free to use the language of your choice. However, we only provide support for Java.
   - You are not allowed to use search and motion planning library.
3. Write a report. The report should explain your problem formulation, the method you use, and a convincing argument of why your solution solve the problem up to the level you want (more on this in "Marking Scheme" section). To make a convincing argument, you need to argue conceptually and show experimental results. This also means that you need to be able to design test scenarios that will show the good and bad properties of your solution.
   The report must not exceed four A4 pages (inclusive of all pictures). The font size should be at least 11pt. The top, bottom, left, and right margin should be at least 1cm.

Your deliverables are:

1. The program. This includes all the source codes and test scenarios in your report.
2. The report, in .pdf format.

Marking Scheme

The total mark is 100, with a maximum of 70 points for the program and a maximum of 30 points for the report.

The marking breakdown for the program is as follows.

- COMP3702 students:
  - $[10, 27.5]$: Program does not run. This spans from failing to compile to compile and run but does not solve even a single scenario for securing 30 points. The teaching staffs will have full discretion on the exact mark you will get.

  - $[30, 35]$: Solve the scenario(s) where the robot consists of only the base (i.e., 0 number of joint and link). There will be 3 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[37.5, 40]$: Solve the scenario(s) where the robot consists of the base and a single joint and link. There will be 2 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[42.5, 50]$: Solve the scenario(s) where the robot consists of the base and 2–4 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[52.5, 60]$: Solve the scenario(s) where the robot consists of the base and 5–8 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[62.5, 70]$: Solve the scenario(s) where the robot consists of the base and 9–12 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
- COMP7702 students:
  - $[10, 22.5]$: Program does not run. This spans from failing to compile and run but does not solve even a single scenario for securing 25 points. The teaching staffs will have full discretion on the exact mark you will get.
  - $[25, 30]$: Solve the scenario(s) where the robot consists of only the base (i.e., 0 number of joint and link). There will be 3 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[32.5, 35]$: Solve the scenario(s) where the robot consists of the base and a single joint and link. There will be 2 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[37.5, 45]$: Solve the scenario(s) where the robot consists of the base and 2–4 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[47.5, 55]$: Solve the scenario(s) where the robot consists of the base and 5–8 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[57.5, 65]$: Solve the scenario(s) where the robot consists of the base and 9–12 joints and links. There will be 4 scenarios of this type, and each scenario will be worth 2.5 points.
  - $[67.5, 70]$: Solve the scenario(s) where the robot consists of the base and 13–15 joints and links. There will be 2 scenarios of this type, and each scenario will be worth 2.5 points.

The marking breakdown for the report for both 3702 and 7702 students are as follows.

- A maximum of 7.5 point for problem formulation. We will check for completeness and correctness.
- A maximum of 7.5 point for the description of the method. We will check for readability of the idea and concepts behind your solution. This means, you need to explain what your method does in a high level manner accessible to third year computing students, without explaining the details.
- A maximum of 7.5 point for the conceptual arguments on why your solution will solve the problem up to the level you want to solve. We will check for logical argument and understanding of the problem and method.
- A maximum of 7.5 point for the experimental arguments. We will check for how reasonable the test cases for supporting certain claims are.

Although the maximum points for 3702 and 7702 on each component of the report are the same, the marking for 7702 student will be stricter than 3702 students.

oOo THE END oOo

"I don't know why I was suspended from doing surgery! I'm not the one who loaded the Appendix app rather than the GallBladder app!"

Source:http://webserver.computoredge.com/online.mvc?zone=sd&issue=3119&article=cover&session=