Semester 2 2013
COMP3702/7702 ARTIFICIAL INTELLIGENCE
ASSIGNMENT 1: OIL SPILL CONTAINMENT SYSTEM

Note:
• You can do this assignment individually or in a group of 2-3 students.
• For those who choose to work in a group:
  o All students in the group must be enrolled in the same course code, i.e., all COMP3702 students or all COMP7702 students.
  o Send the group name and members (name and student ID) to comp3702-staff@itee.uq.edu.au with subject group for Assignment1 before Tuesday, August, 13th 11.59am. If we do not receive your e-mail by then, you will need to work on the assignment individually.
• Please submit your source code and report by e-mail to comp3702-staff@itee.uq.edu.au with subject Assignment1 before Tuesday, August, 27th 11.59pm.
  o If your code is more than one file, please put them in 1 folder named assg1_studentID(s) and send the folder as a zipped file.
  o If your code consists of only one file, please name it assg1_studentID(s).[extension].
  o Example for naming the folder/file of your source code:
    For a group of students with IDs 12345 and 98231, the folder/file name would be assg1_12345_98231 (followed by appropriate extension).
  o Please ensure that your program can be compiled from command line, and can be run from command line as :
    [java/python/...] assg1_studentID(s) input_file_name output_file_name
  o For the report, please submit your report in .pdf format.
• Please also submit a hardcopy format of your report during class on Thursday, August 29th.

Congratulations!!! You have been hired as an engineer at Australian Petroleum (AP), and are assigned to AP’s disaster-response team. The team is exploring efficient methods to contain oil spills in marine environments. Around 96% of AP’s oil production comes from its offshore installations, making oil spills in marine environment a real and tangible risk for AP. A commonly known oil spill containment strategy is to use boats carrying booms to sweep the spilled oil to a certain area (Fig. 1). However, this operation requires highly skilled boatmen, who are difficult to find and are very costly. To reduce cost, AP’s disaster-response team is exploring the use of Autonomous Surface Vehicles (ASVs) to carry booms. You have been assigned to propose a solution to enable a group of

Fig. 1. Boats carrying booms.
ASVs to collaborate in carrying booms and sweeping the oil to a pre-specified destination.

As a first step, you simplify the problem and try to solve this simplified problem. The simplifications are:

1. Each ASV is modeled as a point, each boom as a straight line segment, and the environment as a normalized 2D plane (size [0,1]X[0,1]).
2. You only need be concerned about the positions of the ASVs (i.e., velocity, acceleration, momentum, etc. can be ignored).
3. The environment is fully observable, deterministic, static, continuous, and known.
4. All obstacles in the environment have rectangular form, and are axis-aligned.
5. The cost of moving from one configuration to another is the sum of the straight-line distance traveled by each ASV.

The requirements for the containment system are:

1. The length of each boom may be fixed at 0.075 unit length or variable with range [0.05, 0.075] (the difficulty and hence grades are different).
2. The system must form a connected chain at all times. A connected chain means each ASV can be connected to at most two booms and each end of each boom is tied to an ASV.
3. The polygon formed by connecting the two ends of the connected chain must, at all times, be convex and have an area of at least \( \pi r_{\text{min}}^2 \), where \( r_{\text{min}} = 0.007(n-1) \) and \( n \) is the number of ASVs. This geometry is needed to ensure the oil that has been contained does not spill out again.
4. The booms must never intersect with each other.
5. Booms and ASVs must never intersect with obstacles.
6. Booms & ASVs cannot move outside the [0,1]X[0,1] workspace.
7. The primitive step is a distance of at most 0.001 for each individual ASV.
8. Each requirement (1-6) must hold at each primitive step. Since the distances are very small (at most 0.001 unit length for each ASV), it is sufficient to test the requirements only at the end of each primitive step.

Your task is to develop a program that calculates a collision free path for the ASVs to move from a given initial configuration to a given goal configuration, while satisfying the above requirements. In relatively open environments, the path must be found in under 1 minute (average of 30 runs) on a PC with Intel Core i7-2600 3.4GHz and 16GB RAM (e.g., any PC in 78-116). In cluttered environments, the path must be found in under 5 minutes (average of 30 runs) on the same PC.

Steps you can take to develop the solution:

1. Please define the state space, action space, world dynamics, and utility function of the problem.
   
   Hint: This is your chance to make the problem easier to solve.
2. Please define a strategy to generate a state graph representation of the problem definition in 1 and please implement this strategy. If you use a Probabilistic Roadmap (PRM) approach, please be careful on the sampling strategy you choose (esp. if you want to get a 6, 7, or prizes).

3. Please implement a Uniform Cost search on the state graph to solve this simplified problem of containing oil spills in marine environments. Please program your own implementation of step-2 and step-3 (no library is allowed). You are allowed to use data structure library (e.g., implementation of priority queue, stack, hashmap, etc.).

Submission information

You need to submit your solution in the form of source codes and a report. The report is at most 3 A4 pages, and should contain:

- A description of the proposed method.
- An explanation why you think the proposed method is a good solution.
- An explanation on the limitations of the proposed method.

The code should take a text file as input, and output the path in a text file. The format of the input and output files are as follows.

**Input format:**

a. The file consists of $k+4$ lines, where $k$ is the number of obstacles. The first line is $n$, the number of ASVs. The next two lines are the initial and goal configurations. The next line is $k$, the number of obstacles. Each line in the last $k$ lines represents the vertices of each rectangular obstacle.

b. The initial & goal configurations are written as x-y positions of each ASV, followed by the length of each boom. Boom-1 is attached to ASV-1 and ASV-2, ..., boom-$(n-1)$ is attached to ASV-$(n-1)$ and ASV-$n$.

c. Each rectangular obstacle is written as a quadruple of the x-y coordinates of its vertices, in counter-clockwise order.

d. Example of an input file:

```
2
0.3 0.5 0.375 0.5 0.075
0.6 0.5 0.665 0.5 0.065
2
0 0.2 0.2 0.2 0.4 0 0.4
0.5 0.6 0.7 0.6 0.7 0.9 0 0.9
```

The input file means:

- The system consists of 2 ASVs (and 1 boom).
- The initial positions for ASV-1 and ASV-2 are $(0.3, 0.5)$ and $(0.375, 0.5)$. Boom-1 (attached to ASV-1 and ASV-2) has length 0.075.
- The goal positions for ASV-1 and ASV-2 are $(0.6, 0.5)$ and $(0.665, 0.5)$. Boom-1 (attached to ASV-1 and ASV-2) has length 0.065.
- There are two rectangular obstacles. The vertices of the first obstacle are $(0, 0.2), (0.2, 0.2), (0.2, 0.4), (0, 0.4)$. The vertices of the second obstacle are $(0.5, 0.6), (0.7, 0.6), (0.7, 0.9), (0.5, 0.9)$.


**Output format:**
The file consists of \( k+2 \) lines. The first line is the number of primitive steps and the total length of the path, separated by a white space. The next line is the initial configuration, and each line in the next \( k \) lines contains the position of each ASV at the end of every primitive step. For example:

```
390 0.59
0.3 0.5 0.375 0.5
0.301 0.5 0.376 0.5
0.302 0.5 0.377 0.5
```

... (more lines)

```
0.4 0.5 0.475 0.5
0.401 0.5 0.476 0.5
0.402 0.5 0.477 0.5
```

... (more lines)

```
0.5 0.5 0.575 0.5
0.501 0.5 0.575 0.5
0.502 0.5 0.575 0.5
```

... (more lines)

```
0.599 0.5 0.575 0.5
0.6 0.5 0.575 0.5
0.6 0.5 0.574 0.5
0.6 0.5 0.573 0.5
```

... (more lines)

```
0.6 0.5 0.566 0.5
0.6 0.5 0.665 0.5
```

The above output means the solution path consists of four nodes (including the initial and goal nodes). ASV-1 moves from \((0.3, 0.5)\) to \((0.301, 0.5)\) to ... to \((0.6, 0.5)\), ASV-2 moves from \((0.375, 0.5)\) to \((0.376, 0.5)\) to ... to \((0.665, 0.5)\).

We will provide a Java program to visualize the input and output files (this program will be downloadable from the class website).

**Grading**

Solution & program: 15%.
Report: 5%.

Marking scheme for solution & software:

**COMP3702:**
- 4: 3 ASVs, rigid booms with fixed length.
- 5: 3 ASVs, rigid booms with variable length.
- 6: Up to 7 ASVs, rigid booms with variable length.
- 7: Up to 7 ASVs, rigid booms with variable length, in cluttered environments.

**COMP7702:**
- 4: 3 ASVs, rigid booms with variable length.
• 5: Up to 7 ASVs, rigid booms with variable length.
• 6: Up to 7 ASVs, rigid booms with variable length, in cluttered environments.
• 7: Up to 15 ASVs, rigid booms with variable length, in cluttered environments.

Marking scheme for report:

**COMP3702:**
• 4: A “core dump” of code documentation.
• 5: Clear explanation on the concepts of the proposed solution.
• 6: Requirements for getting a 5 + clear explanation on why you think your solution is suitable for the problem.
• 7: Requirements for getting a 6 + clear explanation on what are the limitations of your solution.

**COMP7702:**
• 4: Clear explanation on the concepts of the proposed solution.
• 5: Requirements for getting a 4 + clear explanation on why you think your solution is suitable for the problem.
• 6: Requirements for getting a 5 + clear explanation on what are the limitations of your solution.
• 7: Requirements for getting a 6 + convincing arguments that the proposed solution is the most suitable, compared to other existing solutions.