

Homework 2

-- Course: *Intelligent Robotics* – Professor: *Qi Hao*

Homework Submission Instructions Please write up your responses to the following problems clearly and concisely. We require you to write up your responses with A4 paper. You are allowed and encouraged to work together. You may discuss the homework to understand the problem and reach a solution in groups. However, each student must write down the solution independently. You must understand the solution well enough in order to reconstruct it by yourself. (This is for your own benefit: you have to take the examinations alone.)

Written Homeworks. All calculation problems must be written on single-sided A4 paper. The scan version of the paper will also be accepted.

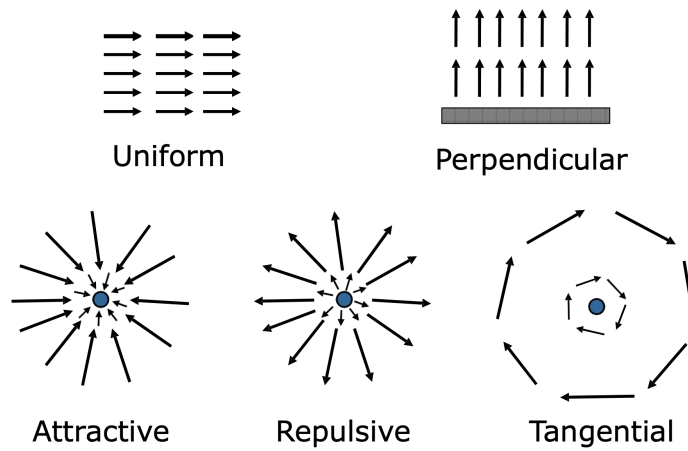
Coding Homeworks. Most of coding assignments will be done by Python(≥ 3.5) under a simple [robotics simulator \(ir_sim 1.1.8\)](#). You can follow the [Coding instruction](#) to use this simulator to complete the coding part in question2 and question3. Your final submission should be a compressed package with the extension .zip, which includes your codes and explanations (you need to know how to write the manuscript with Markdown or LATEX). Your code should be run step-by-step without any error. Real-time animation is strongly encouraged.

Question 1

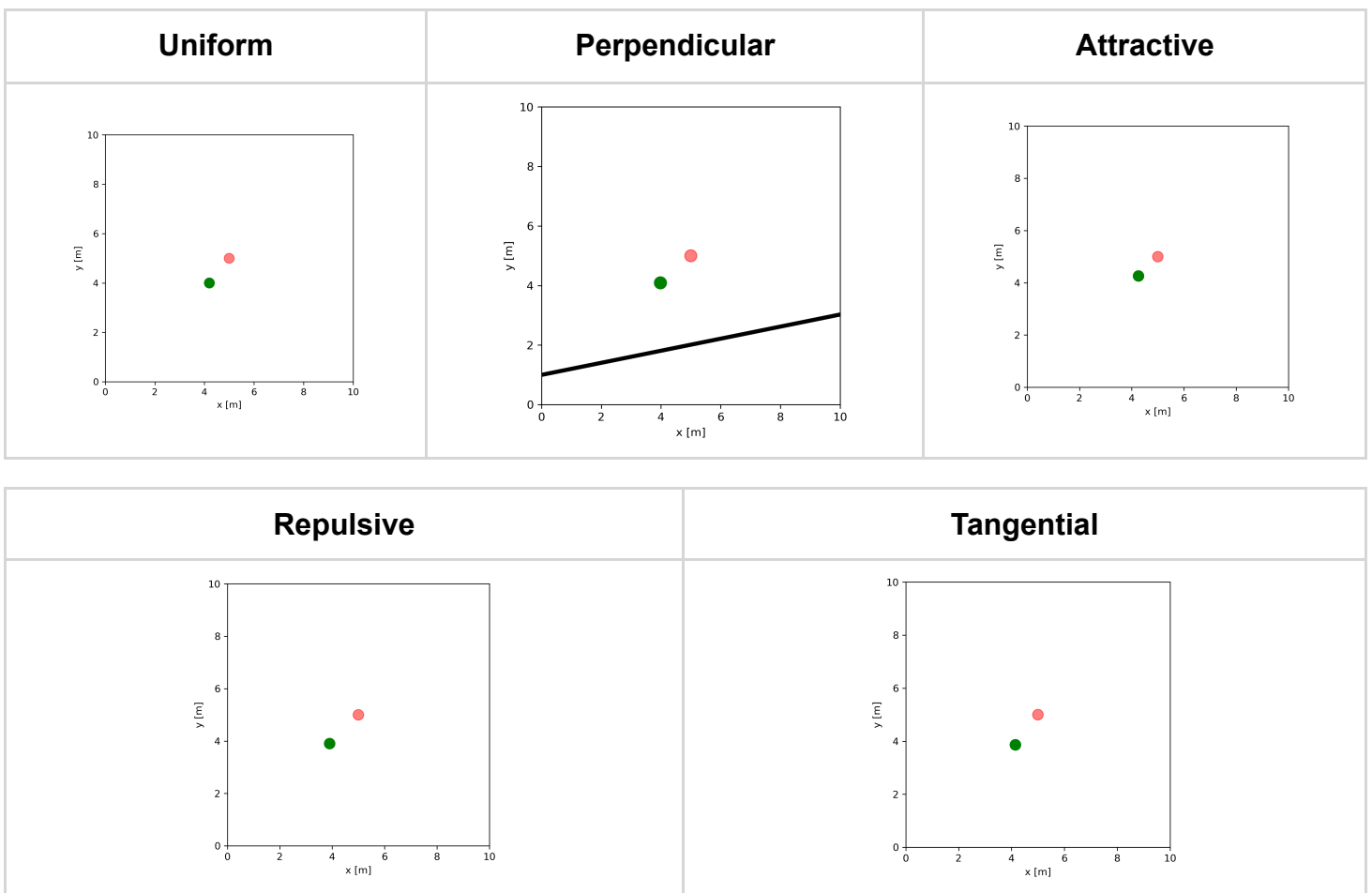
- (a) Imagine you want your robot to perform navigation tasks, which approach would you choose?
- (b) What are the benefits of the behavior based paradigm?
- (c) Which approaches will win in the long run?

Question 2

- (a) How to generate uniform, perpendicular, attractive, repulse, tangential forces for a robot and obstacles with known positions? Provide related mathematical formulas.
- (b) Please simulate the motions of a robot with the given force fields from the following figure.

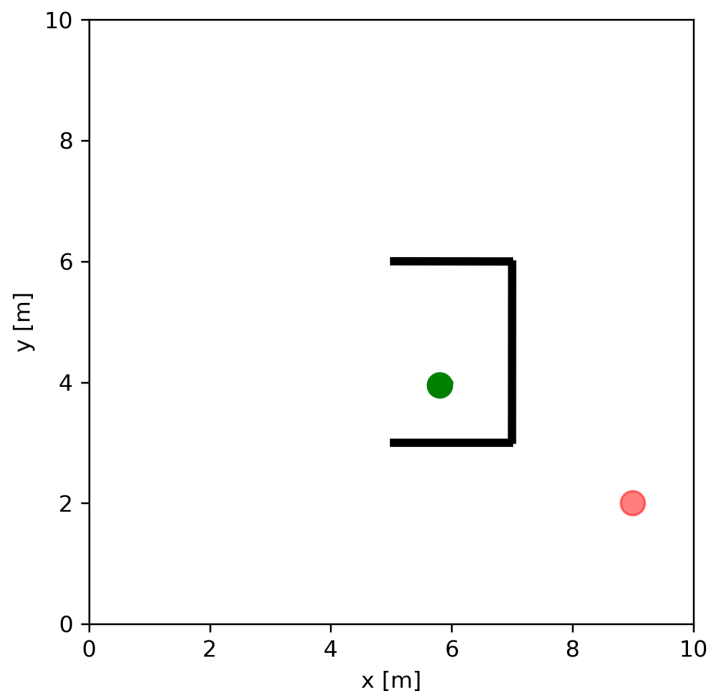


Provide codes and plots of simulation results. You can follow the [instruction](#) to complete the file *potential_fields.py* and run *question2_run.py* in source folder to generate the gif as shown below:



Question 3 - Extra Credit

Simulate a robot can reach the goal without sticking into a local trap. Provide codes and plots of simulation results. You can follow the [instruction](#) to complete the file *question3_run.py* in the source folder to generate the animation as shown below:



Coding instruction

All the assignments are performed based on a simple python based intelligent robotics simulator [ir_sim](#), which provides the common robot motion model and plot functions that we can focus on the robotics algorithm directly.

Install the ir_sim

You can install this repository by pip:

```
pip install ir_sim == 1.1.8
```

Note: Please confirm that the version is 1.1.8.

Common used API

- *env.robot.state*: 2*1 matrix, The position of the robot with x and y
- *env.get_obstacle_list()*: get a list of line obstacles. Each line obstacle includes the position of two points.
- *env.robot.goal*: 2*1 matrix, The goal position of the robot with x, y
- *env.render()*: Show current figure at each step.

Code for question2

There are three files for question2 in the source folder, [question2.yaml](#), [question2_run.py](#), and [potential_fields.py](#).

- [question2.yaml](#) is the parameter set for the simulation environment.
- [question2_run.py](#) is the python file to perform and test the robot and potential field algorithm.
- [potential_fields.py](#) is the potential field class which defines the primitive potential fields as shown in the mentioned figure. You should complete this file.

You should complete the file [potential_fields.py](#) and run [question2_run.py](#) to show the simulation results by the command:

```
python question2_run.py -f FORCE
```

where, 'FORCE' is the one of the force such as: uniform, perpendicular, attractive, repulsive, tangential.

For example, to test the force of uniform, you can run the command:

```
python question2_run.py -f uniform
```

Moreover, you can save the animations by the command:

```
python question2_run.py -f FORCE -a
```

In addition, you can change the number of arguments of coefficient in the [potential_fields.py](#) by your need.

Note1: *we provide the function which calculates the minimum distance between a point and a segment.*

Note2: *The robot is with omnidirectional wheel for simplicity*

Code for question3

There are two files for question3 in the source folder, [question3.yaml](#), [question3_run.py](#). The [potential_fields.py](#) that completed in question2 can also be used for this task. You should complete the file [question3_run.py](#) to simulate a robot to reach the goal without sticking into a local trap. *hint: define the different parameters for the force of repulsive in the two sides of the lines*

You can test your algorithm and save the animations by the command:

```
python question3_run.py -a
```

Note: *The robot is with omnidirectional wheel for simplicity*