CS 790(X) – Advanced Topics in Robotics

Instructor: Monica Nicolescu

Lab 2 – Handout

1. Introduction

In this lab you will learn how to write your own behaviors and how to deal with cooperative behavior coordination through behavior fusion. For this lab, sample code is available with a set of basic behaviors (avoid, wander, wall attractor). You will use these behaviors as example to write your own behavior.

In this lab we will use a potential field implementation of behaviors, which allows for easy behavior combination by vector summation. In this system, a controller consists of a set of concurrently running behaviors. Thus, for a given task, each behavior brings its own contribution to the overall motor command. These contributions are weighted such that, for example, an obstacle avoidance behavior could have a higher impact than reaching a target, if the obstacles in the field are significantly dangerous to the robot. Alternatively, in a time constrained task, the robot could give a higher contribution to getting to the destination than to obstacles along the way. These weights affect the magnitude of the individual vectors coming from each behavior, thus generating different modalities of execution for the task.

2. Getting Started

- Download the files from `http://www.cse.unr.edu/~monica/Courses/CS790X/Labs/Lab2` into your working directory. You can copy the .tgz file that contains an archive of all the individual files.

- The directory contains the following main files

- `controller.cpp`: main controller loop
- `vec.cpp:` vector operation implementation
- `behaviors.cpp:` behavior file
- `weights.txt:` file that contains the behavior weights.

- To add a new behavior follow the steps below

    - add a new ID for your behavior in behaviors.h
    - increment the number of total behaviors in behavior.h
    - add your behavior code in behaviors.cpp
    - in controller.cpp add a call to your behavior after the call for last behavior
    - in weights.txt add a weight to your behavior

3. Behaviors

Each student will implement the following behaviors:

- **Wall Follow (left/right)**: For a given side of the robot (left/right), the field is a sum of vectors parallel with the surface detected by the laser. Make the behavior flexible in the sense that the robot should follow the side wall that is closest to it.
- **Wall Avoid**: for any laser reading that reaches an obstacle, create a field of vectors perpendicular to the surface at each of those points.
- **Seek Small Objects**: for a contiguous laser reading smaller than a threshold (start with 20 and adjust from there) the robot should return a vector that points to the middle of the contiguous interval. You will need to write an additional function to compute contiguous laser readings. If more "small" contiguous readings exist, the robot should choose to seek the one that is the closest.
- **Avoid Large Objects**: for a contiguous laser reading larger than a threshold (same value as in behavior above) the robot should return a vector that points away from the object, in a similar way as the laser avoid behavior (i.e. have a field of repulsive vectors that add up to a single repulsive vector). If more "large" contiguous readings exist, the robot should avoid the one that is closest (pick the minimum reading from the interval).

4. Testing the Behaviors

Test your behaviors as follows:

- **Wall Follow (left/right)** and **Wall Avoid**: create an environment with curved walls as well as convex and concave walls. Test the behaviors individually (in conjunction with the existing behaviors) – adjust the weights in the weights.txt file, so that you obtain the desired wall following or wall avoiding behavior.
- **Seek Small Objects** and **Avoid Large Objects**: create an environment with several (at least 5) circular objects of similar (small) size (they could be player objects defined in your world file or just walls). Test these behaviors in the absence of the Wall Following behaviors (set their weights to zero), but considering the existing behaviors. Adjust the weights of the behaviors in order to obtain the following outcome: as the robot moves around it gets attracted to objects that are in the distance, because they are small. As the robot approaches the objects, they become larger and the robot should be pushed away from them by the Avoid Large Objects behavior.

5. Submission

These behaviors should be ready for testing by the end of the next laboratory session. We will test them in class, so prepare both types of scenarios in parallel to demonstrate the behaviors.

Appendix

The robot is equipped with a set of primitive behaviors. Among the most important are laser obstacle avoidance (avoid), attraction to unoccupied space (wander), sonar obstacle avoidance (sonarAvoid), distinct from the avoid behavior due to differences in sensor

specifications, and random direction change (random), all implemented using a potential fields approach.

Avoid. The avoid behavior is activated whenever the laser rangefinder returns a value within a distance smaller than 2 meters. In this situation, for each laser reading that reaches an obstacle, the behavior generates a vector whose angle is the bearing to the obstacle plus 180 degrees and whose magnitude is equal to a function of the distance between the robot and the obstacle. We add 180 degrees to the angle to reverse the direction of the vector, such that the robot is repulsed by the obstacle. The function used to determine the magnitude is 2 / d - 1 if d is less than or equal to 2 and 0 if d is greater than 2, where d is the distance to the obstacle and 2 is the maximum range (in meters) at which obstacles can affect the robot's behavior. The resulting vectors are then combined through vector addition to form one vector representing the response of the avoid behavior. For a laser rangefinder with a 180-degree field of view or less, the avoid behavior will always return either a zero vector (if there are no obstacles in range) or a vector that points somewhere behind the robot, as obstacles in front of the robot will generate repulsive vectors that point behind the robot.

Wander. The wander behavior is activated whenever there are unoccupied spaces in the 180-degree front field of view of the robot. For each laser reading that does not reach an obstacle, a vector is generated whose angle is the bearing of that empty space and whose magnitude is 1. These vectors are then combined through vector addition to form one vector representing the response of the wander behavior. For a laser rangefinder with a 180-degree field of view or less, the wander behavior will always return either a zero vector (if the robot is completely surrounded by obstacles in front) or a vector that points somewhere in front of the robot, as open spaces in front of the robot will generate attractive vectors whose directions are within the front field of view.