CSCE 274 – Robotics Design and Applications – Fall 2019

Project 2 (10% over the final grade)

Assigned: October 3, 2019
Due: October 22, 2019

The purpose of this assignment is to give you some practice using serial interface of the Create robots, and some experience with a reactive control architecture.

Instructions

Please read carefully the following tasks and program the robot accordingly. *You should do this assignment in the groups that have been assigned*.

The instructions on how to use the robot and how to submit the project assignment is on the webpage of the class, at the following links

https://drive.google.com/file/d/1Z4sSVl08r5S98r2kKA0xmwTWbPnh8d63/view

https://drive.google.com/file/d/1JRj0926o3-jKDnJP5l_T_xM4pE0sSSgy/view

Tasks

Task 1. Augment the *interface*[1] written for Task 2 in Project 1 including:

a. Adding the possibility to set the robot to Full mode.
b. The reading of the *Bumps and Wheel Drops* sensor data.
c. The reading of all of the *Cliff* (packets 9-13, extremes included).
d. The reading of the Angle and Distance (if not done in Project 1).
e. The use of *Drive Direct*.
f. Play a warning song.

Task 2. Write a *program* that utilizes the augmented interface in the previous task and:

a. Initializes the robot, by setting it in *passive* and *safe* mode (done in Project 1).
b. If the robot is stopped, and none of the Wheel Drops and Cliff are activated, once the *clean/power* button is pressed, it moves according to a random walk: the robot should move forward until it reaches an obstacle, then rotate in place for a 180 degrees plus a small random angle (between -45 and +45 degrees), then move forward again, and repeat. The rotation should be clockwise if the bumper left is pressed, while it should be counterclockwise if the bumper right is pressed. If both of them are pressed, take a random direction of rotation. Note that if the robot starts with a bumper pressed, it should rotate according to the rules described above.
c. If the robot is moving,
    i. when the clean/power button is pressed, stop the robot wherever it is.
    ii. check for the state of the Wheel Drops. In case any of them are activated, the robot should stop and play a warning song.

1 *Interface* here refers to the general concept of Application Programming Interface (API), which is a set of programming instructions (that can be in classes or methods) that can be used to build your own application – i.e., that in Task 2. An example in Python that uses classes can be found in https://python.swaroopch.com/oop.html.

d. Note that the program should **not** terminate and should continue listening for button presses and check the state of the Wheel Drops and Cliff, both in b. and c. cases, until the program is terminated.
e. Note also that when either bumper is pressed or any of the cliff sensors are triggered, the robot should refuse to drive forward, but can still safely rotate in place. On the other hand, if any of the wheeldrop sensors are triggered, neither forward motion nor rotations are safe, because the robot is not resting properly on the ground.
f. Once you are confident enough, you can activate the Full mode.
g. **[Extra Credit]** Log into a file the Distance and Angle, together with unsafe events (namely the wheel drops and/or cliffs) or button press, if happened, over time (timestamp should be included). The format of the file should be as follows:
<timestamp>,<datum>
where datum is distance, or angle, or the string "UNSAFE", in case an unsafe event happened), or "BUTTON", in case button pressed.
h. **[Extra Credit]** Use threads to manage the motion of the robot and the reading of the sensors. Remember that the connection is a shared resource among the different threads, as such you should use a way to synchronize the threads, e.g., `Lock` (https://docs.python.org/2/library/threading.html#lock-objects), as suggested in the Notes on the iRobot Create 2.

Task 3. **[Extra Credit]** Process the log file in order to plot the position of the robot in a 2D graph, e.g., using matplotlib (http://matplotlib.org/users/pyplot_tutorial.html). To plot the position of the robot correctly, remember that the Create 2 is a differential drive robot and that the motion you perform are forward motion and rotation in place. Note that this code should be separate from the robot code and should be executed on your laptop by reading the log file created in step (g).

Comments

Your programs will be evaluated based on both their functionality and their coding style. In the notes for iRobot Create 2 you can find an informal style guide to help give you an idea of what is expected together with the coding style that you should follow.

Evaluation

The solution will be evaluated according to the following.

If the robot does not move there would be -60% penalty.

Task 1. functionality (30):

- Proper message for setting Full mode.
- Correct reading and interpretation of the Bumps and Wheel Drops sensor data.
- Correct reading and interpretation of the Cliff (packets 9-12, extremes included).
- Correct reading and interpretation of Angle and Distance.
- Correct use of the Drive Direct.
- Correct play of a warning song.

Task 2. functionality (40):

- Random walk.
- Correct reaction to bumps.

- Safety conditions properly checked.
- Correct restarting of the robot.
- Full mode tested.

Style (20):

- No duplication of executable code?
- No magic numbers?
- Names match functionality?
- Adequate comments?
- Comments match code?
- Consistent formatting?

Documentation (10):

- Report is complete and clear?
- Required sections exist?
- Free of typos and grammatical errors?

Extra credit (g. and h. of Task 2 and Task 3) (30):

- Use of threads.
- Correct logging of the file.
- Plotting of the data.