COMP105P Tasks, Week 2-3

Tasks

After the basic tasks from last week now we move on to more interesting (and challenging) problems. Note that to complete these you will have to build on what you already have in terms of code and logic. You are encouraged to reuse as much of your code as possible in future tasks by creating header (and source) files that group together tasks that you are likely to repeat often. For example bundle all of the logic that deals with driving a fixed distance or turning a fixed angle into one file and expose this functionality via a header file that you will include in subsequent tasks. This makes your code more modular and manageable. In order to prompt you to do this from this week on we will not be awarding credit for work submitted as a single big C file.

The command line for compiling multiple files together starts to get unwieldy and tedious to type each time, especially on the robots, so we encourage you to write a simple `Makefile`. Then all you have to do each time to recompile is to type `make`. See the information sheet "*4. Compiling code for the robots*" on Moodle for details as to how to do this.

- Task 2.0. Make the robot draw a right-angled triangle using the motor encoders, much like before, but have it move faster.
- Task 2.1. Make the robot follow a wall.

Hints and code

To help you along we have a couple of suggestions.

- In order to complete Task 2.0 you will have to re-structure and remove duplication from your previous code for Task 1.3 (since we will not be accepting everything as one big file anymore). The robot behaves well at low speeds, but you will quickly find out (even in simulation) that at high speeds you need to be smart and dynamically manage the voltage of the motors. For example if you want to make the robot go fast exactly one meter you will have to start fast and slow down at some point in order to avoid overshooting.
- Before you take on Task 2.1 make sure you are comfortable reading the sensors. Spend some time playing with them. We do not (yet) pose any restrictions as to how fast you need to follow the wall or how far away from it you need to be so it should be relatively simple to come up with an algorithm to do the following (we are not looking for anything very complicated).
- Follow a wall smoothly does *not* involve a series of straight lines and fixed able turns. You need to continuously calculate and adjust motor speeds based on the readings you take from the sensors.

- The robots have a program called `sensors` installed. You can run this to see what the robots sensors see. This is useful for debugging hardware problems with the robot, or just getting an idea of the sort of noise you get from the sensors. If there's a problem at a particular location following the wall, run `sensors` and see if they give a particular sort of misread at that location. The format of the output is like this:

  ```
  IF: 54 65 IS: 32 8 US: 65 Enc:  973 942 V: 121
  ```

  This indicates the front infrared (IF) can see objects at distances 54cm (left) and 65cm (right), the side infrared (IS) can see objects at distances 32cm and 8cm, the ultrasound sees something at 65cm, the motor encoders have recorded rotations of 973 degrees and 942 degrees, and the current battery voltage is 12.1 volts.