31385 Autonomous Robot Systems
Robot Programming in C

(rev 1.0)

1 Objective

The objective of this exercise is to give an introduction to programming of basic functions for mobile robots in C.

When you have finished this exercise you will have learned about:

- RHD robot hardware interface
- odometry
- basic motion functions

2 Robot hardware interface RHD.

The input/output data of the robot is managed by RHD (Realtime Hardware Daemon). The available variables can be inspected using the utility *rhdtest* from the terminal.

Start the simulator in the sim directory

```
simserver simconfig.385world.xml
```

Start *rhdtest* in a new terminal and enter *connect write*, you can now see the variables. Enter the command *set speedr 30*, the right motor should now run and the encoderticks *encr* should be counting. Try the same with the left motor. Remember to send a 0 to both motors when finishing this point.

The SMR can also be controlled from a C program using the server RHD which handles the communication to the modules of the SMR. RHD is started automatically when the SMR boots (or the simulator is started) and the C program calls the functions of RHD using a library (API).

Go to the 'square' directory and explore the 'square.c' program with an editor('kate' or any that you prefer can be used), and 'make' the program:

```
make square
```

this will compile square.c and make the executable file square that will drive the robot in a square.

Start the simulator in the sim directory

```
simserver simconfig.385world.xml
```

run the square program in the square directory

```
./square
```

If you look carefully you will see that the robot turns around the left back wheel.

Modify the square program so that the SMR turns around the centre of the vehicle instead of one wheel. Only the case *mot_turn* in the *update_motcon* function should be changed. Test the modified code with the simulator.

3 Basic Motion Control and Odometry.

The objective of this part of the exercise is to implement odometry and basic motion commands based on the odometry and test everything with the simulator. The code should be based on the square program.

3.1

Extend the square program to enable data logging in a big array. At each sample of the mission state machine the mission.time , mot.motorspeed_l and mot.motorspeed_r should be logged. When the program terminates the array should be written to a ASCII file, one line for each sample. Make the file have the extension .dat to enable processing by MATLAB. Why should we use a statically allocated array? Think about what happens if the end of the array is reached. Run the square program with logging and make a matlab plot of the motorspeeds versus time.

3.2

Implement odometry (i.e. find x,y, θ) in the square program. The implemen-tation should be done only modifying the functions reset_odo and update_odo and the struct odotype. Change the logging to log the time and odometry. Test with the simu-lator with both clockwise and counter clockwise squares. Make xy-plots of the odometry tests.

3.3

Modify the program square the make the SMR move 2 m forward and stop. Make the experiment at the speeds 0.2 , 0.4 and 0.6 m/s (speed command). Plot x,y and θ versus time in MATLAB on the home page.

3.4

Limit the acceleration to 0.5 m/s^2 . This could be done by limiting the speed command increment ΔV for each sample to a suitable value. Make the experiment at the speeds 0.2 , 0.4 and 0.6 m/s (speed command). Plot x,y and θ versus time in MATLAB on the home page.

3.5

Limit the deceleration to $0.5\, m/s^2$ This could be done by calculating the maximum allowed speed considering the remaining distance to target d and the allowable acceleration using the formula: $V_{max}=\sqrt{2\, a_{max}\, d}$

Make the experiment at the speeds 0.2 , 0.4 and 0.6 m/s (speed command). Plot x,y and θ versus time in MATLAB on the home page.

3.6

Implement acceleration and deceleration limits for the turn command and base the turned angle on the odometry angle

Test with the simulator and plot x,y and θ versus time in MATLAB on the home page.