31385 Autonomous Robots Systems Exercise Facilities and Simulator

(rev 1.0)

1 Objective

The objective of this exercise is to give an introduction to the exercise facilities of the course.

When you have finished this exercise you will be able to:

- Use the different computers of the course
- Use the robot simulator

2 Introduction the computer environments

The exercises of the course take place in two places:

room 017 and room 027 in building 326
room IT017 AND IT025 in building 325

2.1 Building 326

The laboratories in 326 are equipped with computers running the Linux operating system. These are used to write programs for the robots and run the robot simulator. It is also in these rooms that tests with the real robots take place.

Start by choosing the Xubuntu Session in the top right corner of the login screen, then login. Be advised the **no backup** is done, so please make your own on Dropbox, USB key or any other way you prefer.

To login the following is used:

username:	k385hN	where N should be substituted with your team number
password:	smr4ever	

2.2 Building 325

Building 325 is equipped with SIX person tables with three Windows computers on each. There should be one team at each table.

To get a Linux environment on the windows computers we are using Oracle VM VirtualBox. To login use your **campusnet login.**

First time you login you must download the virtual environment file `simenv385_17.ova` from the link `http://aut.elektro.dtu.dk/staff/naa/31385/simenv385_17.ova`

The machine will rename the file to simenv385_17.tar so you must change the .tar to .ova again after download.

To use the environment start Oracle VM VirtualBox click the 'fil' menu and choose

'importér prækonfigureret system'

and browse for the `simenv385_17.ova` file.

The load and configuration will take a couple of **minutes**. When the system is ready you can start the Linux environment by clicking on the simenv. In the Linux system click in the terminal window and enter

`./go N` where N in substituted with your team number

the password is smr4ever

This will mount the same file system as you see on the building 326 computers in the directory k385. (/home/smr/k385 on the VirtualBox) This means that you will work with the same files no matter which computers you are using. Be advised the **no backup** is done, so please make your own on Dropbox, USB key or any other way you prefer.

To logout of the system click the mouse icon in the upper left corner chose the power icon and click shut down.

3 Introduction to the simulator and SMR-CL

The objective of this part is to give a short introduction to programming of mobile robots. As an example vehicle the SMR (Small Mobile Robot) of DTU-Elektro, Automation and Control is used. The robots are programmed with the language SMR-CL that is designed especially for robot programming. The language is text based which means that the commands are written to a text file which is interpreted by the mobile robot.

The exercise gives a step by step introduction to the language using examples demonstrating the basic commands and the basic syntax. At the end of the exercise the participants will be able to program the robot to drive through an exercise track. All the examples should be tested in the simulator first and afterwards on the robot.

Figure 1 shows how the simulator is used.

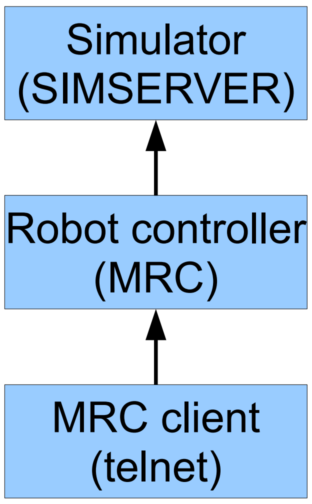


Figure 1: Simulator system.

The simulator simply replaces the robot hardware while the control-software is exactly the same (MRC) as when using the real robot. On the robot the control-software communicates with the hardware via a hardware-server called RHD to which it is connected through a socket. When simulating the control-software simply connects to the simulator instead of the hardware-server.

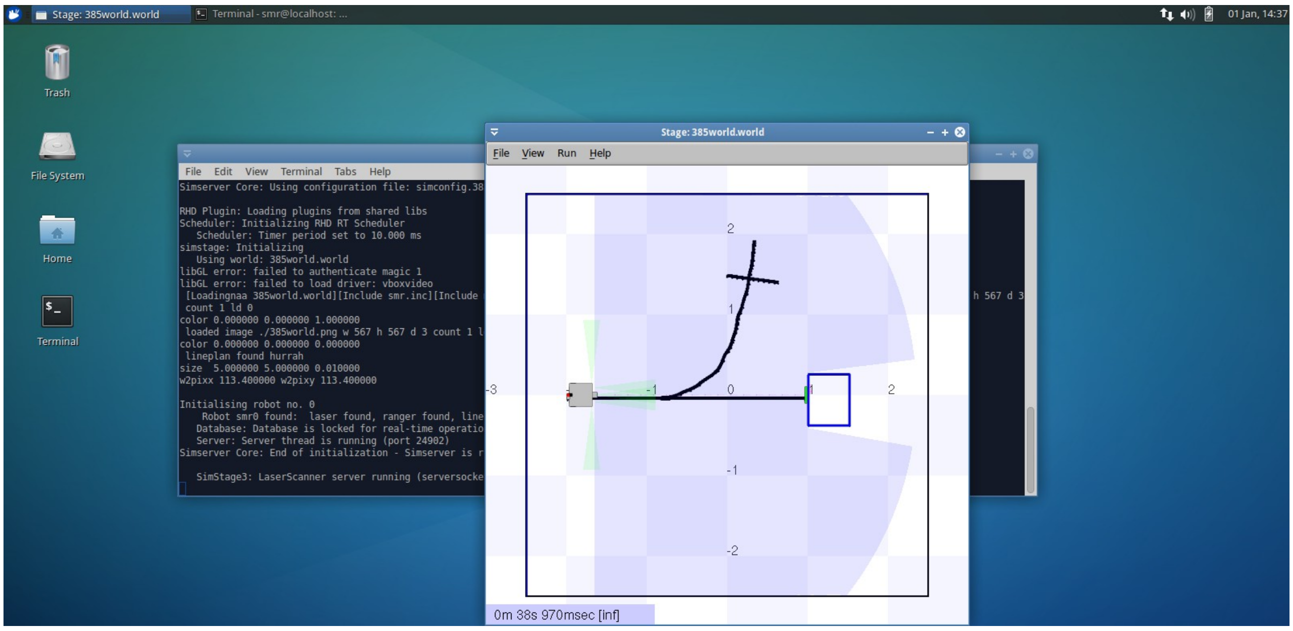
The exercise is run under Linux, so boot up and logon (or start VirtualBox if you are in 325).

3.1 *Simserver stage simulator*

To start the simulator open a terminal and go to the *subdirectory* 'sim', (The pwd command should say /home/smr/k385/sim for virtual box and /misc/shome/31385/hN/sim for 326) and run the command:

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

A window showing the simulated environment should appear.



```
Simserver Core: Using configuration file: simconfig.38

RHD Plugin: Loading plugins from shared libs
Scheduler: Initializing RHD RT Scheduler
   Scheduler: Timer period set to 10.000 ms
simstage: Initializing
   Using world: 385world.world
libGL error: failed to authenticate magic 1
libGL error: failed to load driver: vboxvideo
 [Loadingnaa 385world.world][Include smr.inc][Include
 count 1 ld 0
color 0.000000 0.000000 1.000000
 loaded image ./385world.png w 567 h 567 d 3 count 1 l
color 0.000000 0.000000 0.000000
 lineplan found hurrah
size  5.000000 5.000000 0.010000
w2pixx 113.400000 w2pixy 113.400000

Initialising robot no. 0
    Robot smr0 found:  laser found, ranger found, line
   Database: Database is locked for real-time operatio
   Server: Server thread is running (port 24902)
Simserver Core: End of initialization - Simserver is r

   SimStage3: LaserScanner server running (serversocke
```

3.2 MRC

Open two more terminal windows by clicking the Terminal Icon on the Desktop.

Start MRC in the *mrc* directory (virtual box: cd k385/mrc; 326: cd mrc), as part of the configuration (`calib/robot.conf`) is in this directory. The pwd command should say `/home/smr/k385/mrc (vitual box)` and `/misc/shome/31385/hN/mrc` for 326.

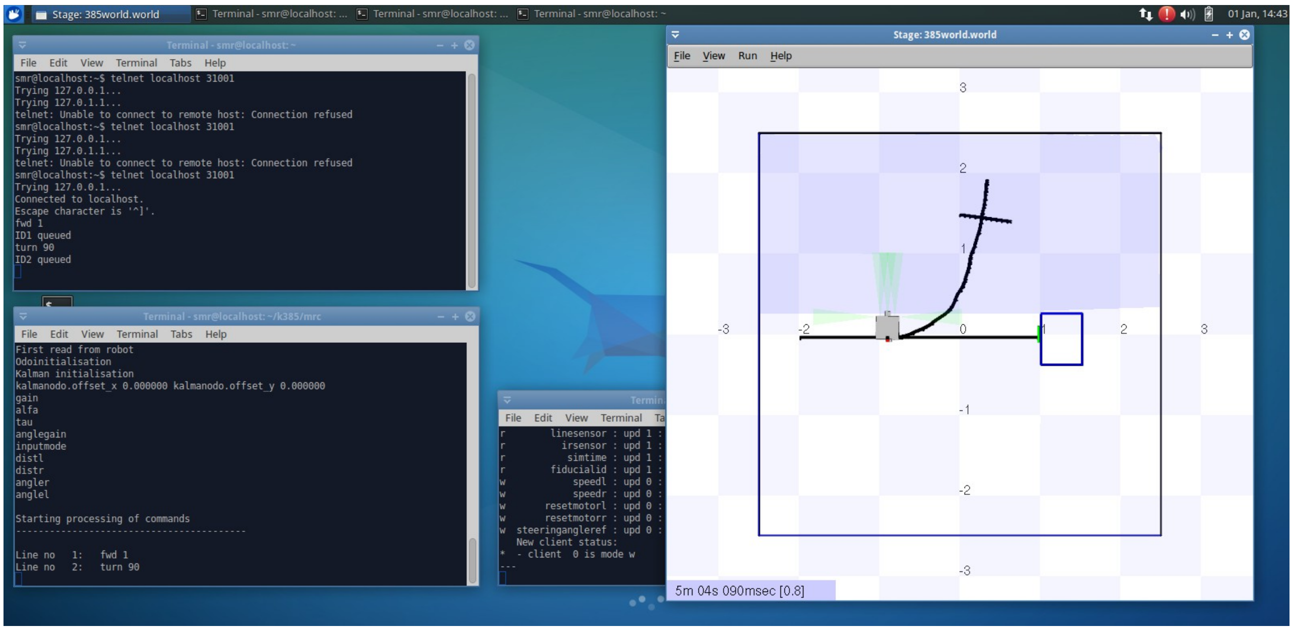
```
    mrc -s0 -t1
```

3.3 Telnet client

Go to another terminal and start a telnet connection to the MRC:

```
  telnet localhost 31001
```

After entering a couple of SMR-CL commands like fwd 1 and turn 90 the simulator should look like this:


Stage: 385world.world
Terminal - smr@localhost: ...
Terminal - smr@localhost: ...
Terminal - smr@localhost: ~
01 Jan, 14:43
Terminal - smr@localhost: ~
File Edit View Terminal Tabs Help
Terminal - smr@localhost: ~/k385/mrc
File View Run Help
5m 04s 090msec [0.8]

```
smr@localhost:~$ telnet localhost 31001
Trying 127.0.0.1...
Trying 127.0.1.1...
telnet: Unable to connect to remote host: Connection refused
smr@localhost:~$ telnet localhost 31001
Trying 127.0.0.1...
Trying 127.0.1.1...
telnet: Unable to connect to remote host: Connection refused
smr@localhost:~$ telnet localhost 31001
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
fwd 1
ID1 queued
turn 90
ID2 queued
```

```
First read from robot
Odoinitialisation
Kalman initialisation
kalmanodo.offset_x 0.000000 kalmanodo.offset_y 0.000000
gain
alfa
tau
anglegain
inputmode
distl
distr
angler
anglel

Starting processing of commands
----------------------------------------

Line no   1:   fwd 1
Line no   2:   turn 90
```

```
r       linesensor : upd 1 :
r         irsensor : upd 1 :
r          simtime : upd 1 :
r       fiducialid : upd 1 :
w           speedl : upd 0 :
w           speedr : upd 0 :
w      resetmotorl : upd 0 :
w      resetmotorr : upd 0 :
w steeringangleref : upd 0 :
   New client status:
*  - client  0 is mode w
---
```

3.4 SMR-CL

A reference manual is found on Campusnet, SMR-CL.

In the first program the following commands are used:

`fwd d @v vel` runs `d` meters ahead with top velocity `vel` m/s

`turn b @v vel` turns `b` degrees counter clockwise with velocity `vel` m/s

If the speed parameter `@v` is omitted, the last entered value will be used

Example

`fwd 0.5 @v0.2` runs 0.5 m ahead with top velocity 0.2 m/s

`turn 90` turns 90 degrees with same top velocity.

After connecting you should be able to give SMR-CL commands to robot using the telnet connection.

Try to give some forward and turn commands to see that the robot reacts as expected. Stop MRC by writing the exit command.

The MRC and the simulator (simserver) should be be restarted to get the robot back to the start position for a new test. Stop the RHD using the x in the simulator window.

3.5 Driving using SMRCL scripts.

Each of the following task should be written as SMR_CL scripts using e.g. the editor kate.

The scripts should be placed in the mrc directory, and run it with the command:

```
mrc -s0 scriptfilename
```

Restart the simulator before you test the script. Each script should be given a new name so that they are ready to be tested on the real robot afterwards,

3.5.1

Make a program that makes the robot run 1.5 meters with a velocity of 0.3 m/s and then turn around and go back to the starting point.

3.5.2

Write a program that makes the robot run in a rectangle with the side length 1m and 0.3 m. Test in the simulator.

In the next program the following command is used

```
drive @v vel :($drivendist > d)
```

The command will make the SMR run straight ahead with the velocity `vel`, `$drivendist` is a variable that contains the driven distance since the command is given.

When the expression in the parentheses is true the program will continue with the next line i.e. the robot will not stop unless told so by the next command.

Example:

```
drive @v 0.3 :($drivendist > 1.0)
stop
```

The SMR runs 1.0m with the velocity 0.3 m/s and stops.

The command:

```
turnr r b @v vel
```

will make the SMR turn `b` degrees counter clockwise with turning radius `r` and velocity `vel`.

3.5.3

Make a program that makes the robot run 0.75 m forward and the run in a square with round corners. The side lengths should be 0.3 m and the turning radius should be 0.3 m. Test the program with the simulator.

In the programs above the movements have been based on odometry i.e. based on measurement of the wheel movements. In the next programs the SMR should run along a black line.

The command

```
followline “bm” @v vel
```

will make the SMR follow a black line (*bm ~black middle*) with the velocity *vel* in the same way the left or right edge is followed with “*bl*” eller “*br*”.

3.5.4

Make a program that makes the SMR follow a black line for 1 m and then stop (c.f. the drive example above). Test the program in the simulator.

3.5.5

Make a program that makes the SMR follow the right branch of a y-fork. The SMR should stop after a suitable distance. Test the program in the simulator.

3.5.6

Make a program that makes the SMR follow the left branch of a y-fork. The SMR should stop after a suitable distance. Test the program in the simulator .

The variable `$crossingblackline` will be 1 when the SMR passes a crossing black line. Otherwise it will be 0.

3.5.7

Make a program that makes the SMR follow a line and stop at a crossing line. Test the program in the simulator.

The variables `$irdistfrontleft`, `$irdistfrontmiddle` and `$irdistfrontright` will hold the distance from the IR-distance sensors in the front of the SMR to the nearest obstacle measured in meters.

3.5.8

Make a program that makes the SMR follow a line and stop if an obstacle is nearer than 0.4 m. Test the program in the simulator.