--- PRELIMINARY VERSION ---

Introduction to Vision and Robotics 2014/15

Second Assessed Practical

Autonomous Navigation[1]

1 Introduction

The aim of this practical is to control a robot such that it interacts smoothly with its environment. The tasks are described in the next section below.

You will work in pairs of students and must submit one joint report (only one student of each pair needs to submit the report). The report must contain a short explanation of how the work was shared and how you think the marks should be distributed. You can choose the partner yourself and you will have to choose a different partner from the first practical assignment. Please contact michael.herrmann@ed.ac.uk immediately if you do not have a partner. He will try to pair you up with the nearest student who has sent the same request.

The assignment is estimated to take about 30 hours of work in total, i.e. your share will be about 15 hours. This time includes writing the report, but excludes revision of any material from the lectures or tutorials. Try to distribute the work evenly and such a way that you can contribute what you are good at.

The work will be done using both the Webots robot simulator (and in an advanced extension Khepera robots). Webots (v 6.4.1) runs at the DICE computers and on all machines in the IVR lab (AT 3.01). License numbers are limited, so, please, never stay connected to Webots when you are not using it. The course page at www.inf.ed.ac.uk/teaching/courses/ivr/ contains a link to a project directory that can be used in Webots. (Khepera robots can be lent from the ITO.)

2 The Tasks

There are two tasks which are to be solved using a simulated robot. Please read carefully the hints below before starting to solve the tasks.

A. Distance control

The robot is supposed to move through a narrow corridor, along a wall, or around a convex obstacle (choose any of these set-ups) while keeping the distance to the wall(s) or the obstacle constant. Arrange the configuration such that the distance can be measured by the IR sensors of the robot.

i. Demonstrate in the simulated robot (and optionally in a real Khepera robot) stable wall-following behaviour (i.e. distance remains essentially constant while the robot is moving). Use a control algorithm in order to keep the distance stable. The robot should be able to reach this behaviour from any starting configuration facing the wall in obtuse angle. (DEMO)

ii. Analyse the dependence of the controlled behaviour on the gain parameters.

1 Given out on 05/03/2015

iii. Implement an algorithm for obstacle avoidance. One option for this is the so-called Braitenberg algorithm: Turn to the left if an obstacle is sensed by the IR sensors on the right hand side and vice versa. Test whether the robot is capable of escaping from a position facing a rectangular corner, an acute-angle corner, or a cul-de-sac. (DEMO)

iv. While moving, the robot should keep track of its position relative to a defined starting position by odometry. It must not use the *x* and *y* coordinates of the simulator, but calculate the position based on the actual wheel movements. Use the odometry information to perform “homing”, i.e. to guide the robot to its base position (i.e. the starting position) after the its job is done. The robot is allowed to use information that it can obtain from the environment during its mission. (DEMO)

v. Analyse the dependency of the odometry error on the distance travelled. Design your experiment such that it can be performed also using a real robot.

B: Surveillance robot:

A robot R is supposed to follow a sentry path P around a C-shaped hallway (the top view is given in Figure 1 below) in order to check for any robot intruders. Have the robot moving through a maze similar to the one shown in the figure. Sometimes the path might obstructed by obstacles O which are supposed to be circumvented by the robot. Try to re-use or adapt the code from the previous task.

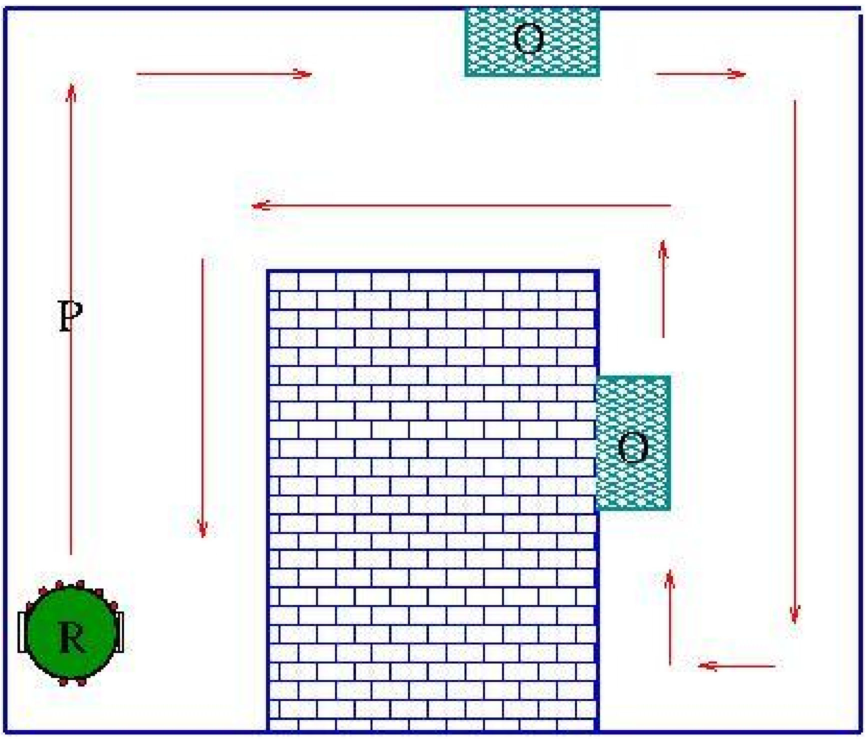

Figure 1.

Choose one of the following options to enable the robot to identify any intruders:

(i) the intruder is recognised as an obstacle in a position where no obstacle was to be expected (For simplicity we will assume that any obstacles are positioned close to the top wall and inner right wall, see Fig. 1),

(ii) by the fact that the intruder can be pushed away from its position while obstacles are immovable,

(iii) by the fact that the intruder moves (for this option you will need to implement a moving intruder),

(iv) using a camera based on the colour of the intruder (for this option you will need to equip the simulated robot with a camera which is available in webots).

If the robot has identified an intruder it should call for reinforcement (simply print a message). If no intruder was found the robot should finish at the starting position after one round through the environment.

Bonus Points:

If all this is successfully completed and you have also finished most of the report, you may investigate transferring your results to a real Khepera robot and carry out the same demonstration and comparisons.

This part is not required to earn full mark for the assignment, but you may receive bonus marks if you can show results. This part is for demonstration only. You do not need to mention this part in your report, unless it contributed to any design decisions in the other parts.

Hints

- Apply control theory! The main ideas of control theory and some reading material were given in the lecture. Although also other solutions are generally acceptable (if they work), control-based solution will typically be more robust and require less coding and less testing.
- If you see several options in solving a task, congratulations! You have understood a very crucial point in robotics. For the assignment, you may not be able to realise all of the option, so try to choose the one that seems to be the most interesting.
- Avoid perfectionism. The behaviour in task A(i) can still be a bit wavy, it is important that the robot stays near the wall and does not touch it. In A(ii) it is not important that the robot can escape from every imaginable configuration of obstacles, etc.
- The tasks can be carried out in Matlab or in C. Note that the communication between Webots and Matlab sometimes is a source of problems.
- It may happen that some basic function of the robot or the simulation is not working. The robot may not respond, Webots does not interact with Matlab or with the real robot etc. Please refer to the guidelines of the supervised practicals for help. Next check the course page for up-to-date information on the issue. If the problem is still not solved, then contact either a tutor or the lecturer (s. Sect. 6 below).
- Ideally, identical programs should control both on the simulated and real robots. Practically, it will be necessary to fix some parameters (e.g. gain factors) in either case.
- An important goal is that the robot move swiftly except where high precision is required. Try to optimize the trade-off between speed and accuracy in a way that is appropriate for the specific task under consideration.
- Behaviours that are described as (e.g.) “essentially straight” may contain small random deviations. Autonomous robots are essentially noisy.
- Required files (Webots environment and a simple controller) can be found in this archive http://www.inf.ed.ac.uk/teaching/courses/ivr/practicals/ivr2015_pract2.tgz, see also the link at the course page. Unpack the archive using “tar xvfoz

ivr2015_pract2.tgz”, then go to the directory ivr2015_pract2 and start webots. More hints can be found in the guidelines for the supervised practicals.

- For the purpose of analysis of the robot behaviour you may use the robot coordinates. These can be obtained in the simulator by the controller of the “supervisor” (see world tree or the *.wbt file). For a real robot, good estimates of the coordinates can be obtained using an overhead camera. The control of the robots, however, should only use the on-board sensors, and not the spatial coordinates.
- Testing is not too interesting a task for the larger part of the runtime. Use the fast-forward button in Webots such that you can focus on the more challenging aspects.
- In order to coordinate the tasks (e.g. moving, orienting, obstacle avoidance) it is recommended to prioritize tasks (this could be implemented based on the so-called *subsumption* architecture). In the control program this can be implemented by nested if-statements. Start with a graphical representation of the program structure.
- You should approach the tasks in several stages. Take a sensible approach to the system design: THINK about what components you need; PLAN the tasks to be done, how they will be done, by whom and in what order; IMPLEMENT, and TEST. Your final mark will be based on how well you explain your approach to the task and evaluate the capability of your program. If your control algorithm isn’t reliable, you won’t fail if you can provide a sensible explanation of what you attempted and the limitations and problems in your report.

3 Writing the report

The report should be a concise description of what you did, why, and what happened. The entire report should be between 1000 and 2000 words (excluding appendices and references). Program listings are not required since you are asked to submit your code separately. The report should contain the following sections:

a) Introduction: and overview of the main ideas used in your approach.

b) Methods: Give a functional outline of your code and explain how each part of it is meant to work. Where suitable, justify your decisions, e.g. why you used one method rather than another, what you tried that didn’t work as expected, etc.

c) Results: You should provide some actual quantitative data, from repeated trials on how well your algorithm controls the robot. This can include data on tests that you have performed in preparation of the task, trajectories of the robot(s) in the actual task, comparison between different strategies. Try to represent the data graphically. Well documented failure will get more marks than unsupported claims of success (well-documented success will get even more marks!).

d) Discussion: Assess the success of your program, and explain any limitations, problems or improvements you would make.

4 Live demonstration

You will have to demonstrate your program working in Webots and, optionally, with a real robot.

The demonstration session for this second practical will take place in room AT 3.01 on

Friday 27/03/2015 from 10 am onwards.

The environments where the robots will have to perform its tasks will not be exactly the same as during your preparations. It will have a similar size, the objects will be the same but will be placed in different positions.

5 Submission

Your submission should be a single PDF file and should be submitted electronically by 4pm Thursday 26/03/2015. The command to use for on-line submission is:

submit ivr 2 FILENAME CODE

where FILENAME is the name of your file, e.g. report_s1234567.pdf and CODE is the name of the archive containing your code, e.g. code_s1234567.gz. You are asked to submit your code in a separate zipfile. The code is not subject to marking, but will be referred to in case of ambiguities in the report. The assignment will be marked as follows:

	Marks (to a maximum of 100, worth 10% of coursemark)
Demo: Basic robot control (A)	25 (i: 10, iii: 5, iv 10)
Demo: Surveillance (B)	15
Demo: Real robot demonstration	up to 20 bonus points
Report: Problem analysis, program design	20
Report: Results and evaluation	20
Report: Clarity (including figures, diagrams etc.)	20

6 Demonstrators

Davide Modolo (d.modolo@sms.ed.ac.uk) and Adam Erskine (a.erskine@ed.ac.uk) are the demonstrators for IVR. They will be in the lab at the agreed demonstration sessions, see course page. You can also e-mail them when course equipment or software is not working, but not if your software has bugs (that's your problem!). For any other questions about this practical please contact Michael Herrmann (michael.herrmann@ed.ac.uk, Tel 651 7177)

7 Plagiarism

This assignment is expected to be in your own words and code. Short quotations with proper, explicit attribution are allowed, but the bulk of the submission should be your own work. Use proper citation style for all citations, whether traditional paper resources or web-based materials. Before submitting please acknowledge any additional sources of code that you use and ensure that your submission follows the school policy on plagiarism: http://www.inf.ed.ac.uk/teaching/plagiarism.html.