e-puck2 Lab Assessment Briefing

ACS6501, last updated on 18.10.2019

Overview

For **Task 1**, you shall design, implement and test a control strategy for an e-puck2 robot

- to explore an environment with obstacles;
- to avoid any collision with the environment boundary or obstacles.

For **Task 2**, you shall design, implement and test a control strategy for the e-puck2 robot

- to chase an object in an environment that is free of obstacles (see Figure 1);
- to avoid any collision with the object.

Your control strategy must be executed directly on-board the e-puck2 robot. There are 5 laboratory sessions. No access to the robot will be provided outside of these sessions. It is thus vital to arrive well prepared.

During the lab sessions, students will be working in teams (typically 3-4 students per team). Each group is allocated a Linux PC (Ubuntu) that they should use to program the robot. There is space for a laptop, if the group chooses to bring one. The consumption of food or drinks is not permitted inside the laboratory.

Please make yourself familiar with the e-puck 2 Lab Induction document available on Blackboard (**e-puck2 Lab Induction.pdf).** It explains the robot and how to setup the computer at the beginning of your lab session, and how to backup your solutions before you leave. We strongly encourage you to solve the optional tasks, as they help gain a better understanding of the robot's core features.

A cheat sheet of the e-puck2 library is provided on Blackboard (**e-puck2 Library Cheat Sheet.pdf**). It gives example code and explains how to use the various features of the robot.

Your are expected to design and implement a solution for each task using any available routines in the e-puck2 library. You can assume that:

- The environment is bounded (a white tray) and may include obstacles;
- The object to be chased is is shown in Figure 1.

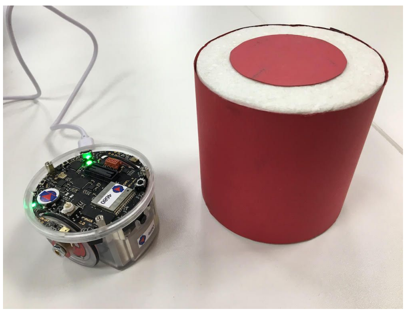

Figure 1. The e-puck2 robot and the object to be chased as part of Task 2. The object has a height of 10 cm and a diameter of 10 cm [1].

2 Demonstration (50%)

You will have to perform demonstrations to a Graduate Teaching Assistant (GTA) in the assessed laboratory sessions, and retain a backup of your solutions.

- You have 2 attempts. They are in Sessions 3 and 5. For each task, you will receive the maximum of the *total marks* awarded in either attempt.
- You must be ready to perform the demonstration no later than 60 min after the start of the session.
- You have to return the e-puck2 no later than 10 min before the end of the session.
- You are allowed to upload a dedicated program to the e-puck2 for each task.
- Note that once the e-puck2 has started you are no longer permitted to interact with it.
- **It is your responsibility to backup your solutions at the end of each session**.
- If you have not demonstrated your program at the end of Session 5, you receive a zero mark.

The detailed marking criteria are listed in Table 1.

Table 1 (Marking criteria for demonstration). The listed points are the maximum points that can be achieved per task.

Points (out of 100)		Robot ability
Task 1	Task 2	
5	-	Task 1: The robot moves (i.e., the program does something).
15	-	Task 1: The robot is able to fully explore an environment while avoiding collisions with any obstacles. • Up to 8 points are awarded for exploration. Ideally we would like to see that the robot eventually ends up in every region of the environment. For example, a robot that does wall-following may never explore the open space in the centre. • Up to 7 points are awarded for avoiding collisions. Ideally the robot does not touch the wall during exploration, even if put in a difficult situation, such as a narrow passage or dead end.
-	10	Task 2: The robot is able to approach the (passive) object. Here it is assumed that the object is not far away, so that using the proximity sensors alone it should be possible for the robot to register the object. The robot needs to approach the object, no matter where it is placed (e.g. front, left, right, back).
-	10	Task 2: The robot is able to chase the object, which is moved by a GTA. Ideally the robot can chase an object that is moved in a straight line, but also one that takes turns to the left or right.
-	10	Task 2: The robot does not collide with the object it is approaching/chasing. Here, the only object near the robot is the object to be chased. So the robot should approach and/or chase the object, but never touch it - no matter in what direction the object moves.

10	10	Tasks 1 & 2: The robot's movements are smooth and efficient (including at the time of interacting with obstacles and the object). Up to 6 points are awarded for smooth movement; distinct turns may be OK, but any form of erratic movement is not. Up to 4 points are awarded for fast movement.
30		These points are only awarded if the robot demonstrates abilities that go **significantly** beyond what is expected above. These abilities need to relate directly to the task the robot is performing. Be creative! ● No points will be awarded for trivial extensions; ● 1 to 10 points: the robots' abilities are advanced in non-trivial ways; ● 11 to 20 points: well thought-off, significant advancements of abilities, clearly distinct from what is expected; ● 21 to 30 points: outstanding demonstration that is highly original and/or could have lasting impact (e.g. open days demonstrations).

3 Source code submission

In order for your demonstration mark to count, one member of your group needs to **submit your source code on Blackboard on the day of your final demonstration**. For Task 1, rename your corresponding main.c to **main_task1.txt** and then upload this file. Do the same for Task 2 (**main_task2.txt**). If you used the same file for both tasks (e.g. using the selector switch to choose which task to perform), then call it **main_both.txt** but upload it only once (for Task 1). If you created further files (e.g. Robot.h, Robot.cpp), put all their content into a single file, with proper subheadings (indicating the respective file names).

Use your full team name (e.g. **Team 2.D**) as submission title.

4 Report (50%)

Each student has to write and submit their own report. You can use the source code that was developed by your own team. The report has to be submitted as a **PDF file** via turn-it-in, no later than **Monday, Week 8, 23:59 pm.** The report must be up to **2 pages** long (excluding appendix) and formatted using the standard **IEEE conference template**. The template for LaTeX or Word/OpenOffice can be found here: http://ras.papercept.net/conferences/support/support.php

Your report should be structured as follows (see Table 2, for additional information):

- Title: Come up with a meaningful title, but include "(ACS6501)" at its end.
- Author: Your name
- Affiliation: "[Your surname] is with the Department of Automatic Control and Systems Engineering, The University of Sheffield, UK" (placed on page 1, bottom of the left column, according to IEEE conference template).

- Abstract: Describe in up to 100 words the content of your report (do **not** include any irrelevant details like “ACS6501”, your degree programme, etc).
- Do **not** include any introduction section for this lab report.
- **Strategies** (section): describe your two strategies at a high level.
- **Implementation** (section): describe how you implemented your strategy (referring to all key elements of your source code, which is provided in the appendix).
- **Results and Discussions** (section): Describe what situations were examined and the results obtained. Provide a critical analysis and discuss possible improvements.
- Do **not** include any conclusions section for this lab report.
- Appendix (section): Your source code properly formatted as text (no image), with syntax highlighting and line numbers. For the appendix, use single-column format.

Further guidance on writing reports can be found on Blackboard. This is an academic report, avoid using, for instance, “I”, “my”, “us”, and “we”. An example report in IEEE format (though longer than 2 pages and including sections that you do not need) can be found here: http://dx.doi.org/10.1109/IROS.2014.6943114

The detailed marking criteria are listed in Table 2.

Table 2 (Marking criteria for report). The listed points are the maximum points that can be achieved.

Points (out of 100)	**Criterion**
25	**General writing quality & formatting (10)** Presentation quality of report (excluding appendix). Organisation (title, author name, affiliation, abstract, meaningful (sub-)section structure, paragraphs, figures or tables where relevant); writing quality and clarity (clear and concise language, grammar, no typos, no ambiguous/informal expressions, no inconsistencies etc); adhering to standard conventions (figures/tables with captions and all cited in text [e.g. “Figure 1 shows… ”], abbreviations defined when first used); use of external images acknowledged in caption (e.g. “Image reprinted from [x]”). **Abstract (5)** Motivating sentence providing some context. Stating the problem that is addressed. Stating what (method) is proposed as the solution. Stating what results were obtained. And stating what is concluded from this. **Appendix - source code (10)** Presentation quality of appendix (e.g. appropriate commenting, as well as syntax highlighting, line numbers, consistent indentation).
30	**Quality of the two *strategies*** Explain the strategies you have chosen - *how* do they enable the robot to solve the task(s)? Choose an appropriate level of abstraction. For example, presenting source code or specific parameter values would be too low level (this would be *implementation*).

	You can get up to **20** points for the quality of the strategies themselves, and up to **10** points for presenting them using suitable abstractions (e.g. a correct finite state machine diagram, pseudocode, equations).
30	**Quality of the *implementation*, including source code.** You can get up to **10** points for describing *how* the strategies have been implemented. Avoid repeating what the strategies are - just explain how their different elements are implemented. Do not put source code in the implementation section, rather point the reader to the appropriate lines of source code in the appendix. The implementation section should help the reader understand how the source code (in the appendix) relates to the strategy (in the previous section). You can get up to **20** points for source code that is a correct implementation of the strategy, well structured, and efficiently implemented. Think about whether the code implements the strategy in a correct and efficient manner. Think about whether all lines of source code are needed. Think about using built-in commands.
15	**Quality of *results and discussions* (including critical analysis)** You can get up to **5** points for describing what situations were tested (your experimental setups for Tasks 1 and 2), and the results from your experiments in the lab - what worked, and what did not. Try to do so using unbiased, formal language, where possible based on facts (don't write "The robot did a great job."). You can get up to **10** points for providing a critical analysis of the findings obtained, and for discussing possible future improvements. Where possible ground your discussions, for example, in observations made in the lab.

Penalties (Percentages of the overall mark)

- 10% penalty if source code not provided as text (but image)
- 10% penalty if appendix includes anything other than source code
- 10% penalty if report excluding appendix exceeds 2 pages
- 10% penalty if not using the IEEE template
- Standard penalties for late submissions
- Further penalties (up to 100%) are applied for students failing to attend laboratory sessions (excluding absences that were authorised in advance).

References

[1] Gauci M., Chen J., Li W., Dodd T.J., Gross R. "*Clustering objects with robots that do not compute*," Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems (AAMAS 2014). IFAAMAS (2014) 421-428