e-puck2 Lab Assessment Briefing

ACS6501, last updated on 07.10.2022

1 Overview

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

- to explore a bounded 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 open 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. Two key challenges are

- **to design** and **implement a meaningful control strategy** (as opposed to a bunch of source code lines that happen to work sometimes), potentially drawing on control concepts such as finite state machines, PID control etc.
- **to cope with the inherent uncertainties of *real* robots**. For example, as the laboratory has windows, the level of ambient light depends on weather conditions. Moreover, the hardware of two robots will never behave 100% identically, due to manufacturing differences, thermal noise, wear and tear etc. A good strategy (and implementation) can be adapted reasonably fast to changing levels of ambient light.

There are five laboratory sessions. No access to robots will be provided outside of these sessions. However, you are encouraged to develop your design and code in your own time.

The team allocation (including unique **team IDs**) will be published in Week 3. By default, a **"team"** comprises two students. Where a student does not attend all sessions, their marks will be reduced on a pro-rata basis except for authorised absences that are agreed in advance.

All sessions take place in **202-SHSB**. You are only allowed to work at your assigned desk. Each team is assigned an e-puck2 robot and a Linux PC (**select "Ubuntu" if rebooting**) that must be used to compile and upload the program to the robot. No support will be offered to students that program their robot using their personal devices. You are normally provided with a Logitech c920 camera, which is connected to your computer via USB. If you wish, you can use it, or a smartphone, to take a video of your robot in action. (Please do not record any students or staff though).

The consumption of food and drinks is not allowed within the laboratories.

To facilitate announcements (e.g. including code snippets and fixes), we may use **Blackboard Collaborate** (using the chat to "Everyone").

Prior to attending the first lab session, make yourself familiar with the *e-puck2 Lab Induction* document (at least up to and including Section 4) available on Blackboard (**e-puck2 Lab Induction.pdf).** It explains the robot and how to set up the Linux PC at the beginning of each lab session, and how to share and backup your solutions during the session and before you leave. We encourage you to solve the unassessed tasks in your first session, 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, including the motors and sensors to be used in the assignment. When copying and pasting code from the Cheat sheet, you need to ensure line by line that no mistakes have been introduced (e.g. characters may not copy over correctly; there can be indentation issues).

You are expected to design and implement a solution for both Tasks 1 and 2 using any available routines in the e-puck2 library. You can assume that

- The environment in Task 1 is bounded (a white box or tray) and may include obstacles (e.g. the boxes the robots come in). The obstacles should be detected using the robot's infra-red proximity sensors. No support is provided for detecting obstacles by other means.
- The object to be chased in Task 2 is shown in Figure 1. When conducting a trial for Task 2, the robot may be placed on the desk (to offer more space) provided it is carefully monitored at all times to prevent it from falling onto the floor. At any other time (e.g. while programming) the robot must remain within its bounded environment.

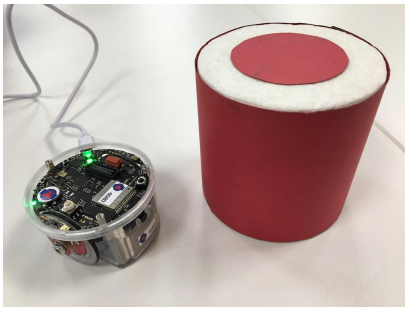

Figure 1. The e-puck2 robot and the object to be chased as part of Task 2. The object has a height of 10 cm and a diameter of 10 cm. Image reprinted from [1].

2 Pre-lab online quiz (penalties may apply if not passed)

The pre-lab quiz will be released on **Wednesday, Week 3 (noon)** via Blackboard. You need to **submit by Friday, Week 3 (23:59pm)**.

You have a single attempt. The questions will be automatically marked. You will receive the feedback on Wednesday, Week 4 (noon).

The pre-lab quiz probes your understanding of the following

- e-puck2 lab essentials (including Health & Safety)

- Basic information about the e-puck2 robot
- Elementary skills of C/C++ programming
- Elementary skills of using a Unix operating system (including the terminal and git)

Not attempting the pre-lab quiz (or obtaining a mark of 0) will result in a 25% penalty on a student's demo mark (other team members are not affected) even where the student is otherwise engaging well with the assignment.

3 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 4 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 5 min in total for both demonstrations.** We will inform you of your assessment time slot at the beginning of Sessions 4 and 5.
- 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**	
10	-	Task 1: The robot moves (i.e., the program does something).
20	-	Task 1: The robot is able to fully explore an environment while avoiding collisions with any obstacles. ● Up to 10 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 10 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.
20		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: significant advancements of abilities, clearly distinct from what is expected; ● 11 to 20 points: outstanding demonstration that is highly original and/or could have lasting impact (e.g. open days demonstrations).

Penalties (Percentages of the demonstration mark)

- If not obtaining a pass mark in their pre-lab quiz, a student receives a penalty on the demonstration mark as outlined in the document *e-puck Lab overview*.

4 Source code submission

One member of your team needs to **submit your source code on Blackboard by Week 8, Monday (noon)**. 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 the submission title.

5 Report (50%)

Each team has to write an **academic report.** To do so, they **will be invited to an overleaf project, which enables them to prepare their report using Latex** using the standard **IEEE conference template** (http://ras.papercept.net/conferences/support/support.php).

You can only 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, noon.** The report must be up to **2 pages** long (excluding appendix) and formatted using the provided standard IEEE conference template.

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, structure (sections and 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 captions (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 report 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).

6 How to Deal with Conflicts with Teams

Conflicts naturally arise when working in groups, and often these are due to misunderstandings and can quickly be resolved. There will be times where a team member contributes more than another, especially if falling behind in their studies, or getting ill. It is suggested that all team members remain supportive of each other and be positive in their communications.

Where a student has concerns about the contribution made by other members of the team, the team should meet to resolve this in a calm and professional manner. It is recommended to do so via a face to face meeting.

Where a team does not manage to resolve a significant conflict on its own (e.g. a member not being reachable for a significant period of time), it is recommended to get in touch with the module leader early on, who will help to resolve the conflict.

Where the module leader determined that there have been significant concerns regarding the engagement of a particular team member, a penalty could be applied to the demo and/or report component of that individual. If this is the case, the module leader will take into account discussions they had with any team member (where applicable), the performance during the pre-lab (if failed), missed lab sessions, reports from GTAs etc.

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