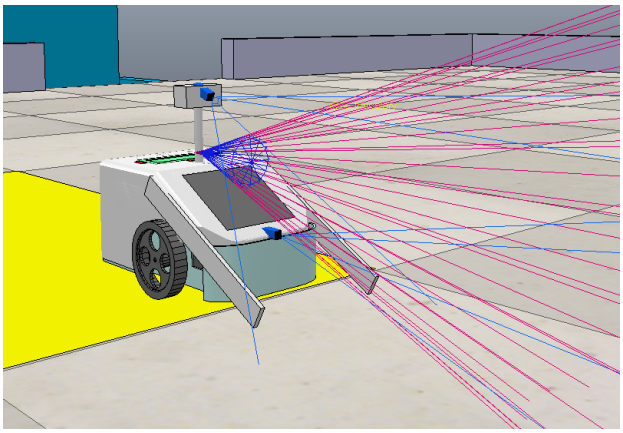
BKI 115A INTRODUCTION ROBOTICS

PRACTICAL ASSIGNMENT



INTRODUCTION

Welcome to the practical assignment for Introduction Robotics! This document gives you all the details you need to get started with the practicals. Primarily, it documents the assignment you have to complete.

As you will see, you will write robot controllers to solve different challenges. There are two types of challenges: a **line following** challenge where you will make use of a **PID controller** and an **object sorting** challenge that should be implemented using a simple **subsumption architecture** in line with reactive paradigms.

Each challenge has two variants – you will solve them both in simulation (using **CoppeliaSim**, install the EDU version from here: https://www.coppeliarobotics.com) and using actual robots (using **Lego Mindstorm Spike**s). The basic idea is that this emulates the workflow in robotics, where solutions are often first designed in a simulator and then ported onto the actual robot. You should thus develop initial code in simulation and then transfer that to the robot (which can, of course, be an iterative process). We will give you starter files for both environments to set up communications between the controllers you will develop and the robots in their respective environments. You will also find instructions and help to get started on the Discogs server.

You are free to work on the simulations whenever you wish, but use of the robots is **restricted to the practical timeslots**. **This means there will be no taking the robots home or elsewhere.** Each group is assigned a robot and a locker. The key to the lockers are kept by the TAs, who will hand them out to the groups at the start of the practical and collect them again at the end. It also means there will be some downtime for the robot, e.g. while it needs to charge, during which you can work on the simulation tasks. **Each group is also responsible for ensuring that the robot is returned as it was given to you at the end of the course**.

Since the robots are Lego kits, you are free to design your own as you wish, although it will help to bear in mind the tasks that need to be solved, which have some implications for e.g. sensor placements.

The point of the practicals is, as the name suggests, to give you some practical experience with robots. This includes the less fun part of working with robots, which is that **things can (and will) go wrong in all sorts of (sometimes frustrating, sometimes amusing) ways** – sensors might misbehave, hardware might fail, bugs can lead to headscratching-inducing misbehaviour of the agent, and so on. Expect this kind of challenge, but **don't let that deter you** – they can be overcome and this is part of the experience. If you are fully stuck, seek the help of your TA.

This also means that the practicals will appear very daunting initially, given the freedom that you have in approaching the tasks and the need to figure out your own ways. Again, don't let that deter you; the flipside is that the payoff at the end is also very satisfying, with a project that will feel like your own rather than something where you just followed step-by-step instructions.

Your grade for this part of the course will be determined by the degree to which your solutions solve the challenges and by a report in which you will document the development of these solutions and reflect on the lessons learned during the practicals. More details can be found at the end of this document.

Good luck!

LINE FOLLOWING CHALLENGE

Figure 1: Max Verstappen wins the Formula 1 race in Barcelona (image from Reuters)

Both the simulation task and the robot task provide you with an approximate recreation of the *Circuit de Barcelona-Catalunya*, a real-world racetrack that hosts amongst others the Spanish Formula 1 GP. Your task is to follow the outline of the circuit and complete a full lap as fast as possible while keeping the behaviour reliable.

For this task, you will need to implement a PID controller. You will need to think, amongst others, about the following:

- What is the error signal? It needs to be something you can evaluate from the sensors your robot uses, and it should contain information about where the robot is headed with respect to the track outline.
- The simulated and real robot, although both wheeled, are rather different in every other aspect, including the behaviour of the sensors. How can you design code such that the core of the controller can be used on both while platform-specific differences need to be implemented too?

Note that you **must** use a PID controller for this task, and it needs to behave like an actual PID controller. That means that the control signal you send to the wheels must be proportional to the weighted sum of the error, derivative of the error and integral of the error. You will of course also need to determine appropriate K terms for each part of the controller, and it is permissible to set one or more of them to zero, effectively disabling the corresponding component, if that works for you (for example, you might find that the integral term can be omitted).

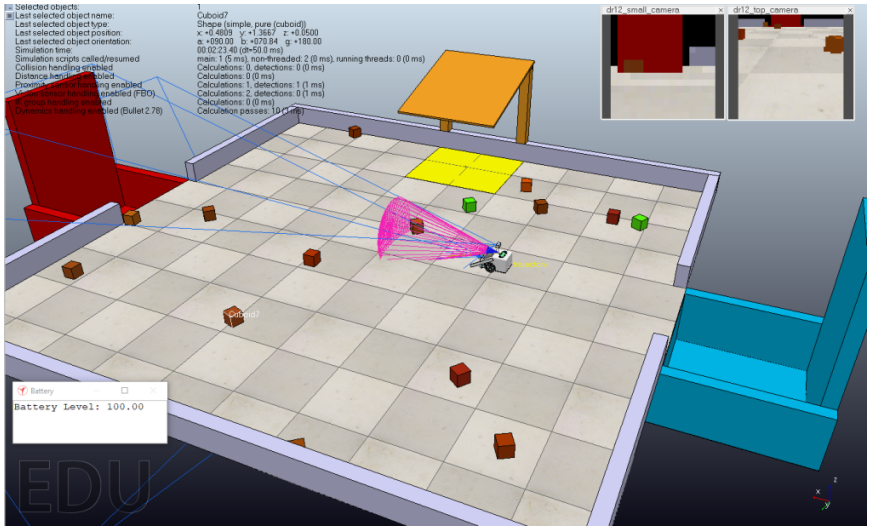


Figure 2: The object sorting arena in CoppeliaSim

In the simulator version of this task, your robot is spawned inside an arena that features two large containers on either side and a charging pad. Small objects spawn over time; these are either plants (green cubes) or thrash (brownish cubes). Plant cubes can be pushed around. Thrash cubes will be compacted by the robot: approaching them from the front will allow you to compact them, resulting in a smaller cube being produced at the back of the robot. In addition, your robot will slowly run out of battery, and the rate at which the battery decays depends on its interactions with objects. You can recharge at any time by entering the charging area. Your objective is to sort all plant boxes into the blue container, compact all thrash boxes, and sort the compacted cubes into the red container.

For the real robot, the task is simplified. The robot will be placed on an arena that is divided into two halves, and on which a number of objects in two different colours are placed. The objects are light enough so they can be pushed by your robot and its task is simply to sort the object such that each half of the arena only contains objects of one colour. Note that objects should remain on the arena, not be pushed off. **Note that this task is hard given the sensors available!**

For this task, you should implement a subsumption architecture from reactive robotics paradigms. You should think, amongst others, about the following:

- What are the necessary “simple” behaviours that need to be implemented? Which of these are common between both versions of the task, and which ones are only needed in one of them? Is there an ordering in terms of priority between them?
- How can you ensure that you can re-use as much of the simulator versions of the behaviours on the robot while the recognizing that the architecture combining them into one controller will obviously need to be different?
- Some behaviours will be the same in both tasks, but pre-conditions might differ. For example, you need to distinguish between different object types in both tasks, but the colours will be different (even if the objects were the same colour, fundamental differences in sensors and lighting conditions mean that would be hard for you to detect). How can you most readily deal with this?

SUBMISSION AND EVALUATION

At the end, you will submit

- The code you have developed
- A report documenting the code, development process, and reflection on the lessons learned (see below). **This is group work, one report and one code base is submitted for each group.**
- A peer review form (available on Brightspace) to self-evaluate the performance of the group. **This form is individual and every member of the group fills out their own copy**.

You will also demonstrate your solution during a **demo day** at the end of term. For this, each team will be assigned specific time slots during which to demonstrate the simulation and the work on the real robot. We evaluate the degree to which the robot solves the tasks, so partial points are possible if some, but not all components work (e.g. the object sorting task requires a few sub-behaviours, some of which are easier to implement than others). The actual schedule for the demo day will be published closer to the time.

REPORT

In addition to the code, you are asked to produce a report. It documents your solutions for each of the challenges. You should, in particular, describe the controller you implemented, focussing on the design of the mandatory aspects (either a PID controller or a subsumption architecture depending on the task).

You should also write a reflection on what you have learned during these practicals. For this, consider the following two statements:

"All robotic work can be done in simulation; a real robot only costs money and adds no benefit."

"Simulators cannot replace robots, in fact, they are so far removed from reality that no meaningful robotic work can be done in them."

As you can see, they are two (overly) extreme statements arguing against any utility of one the two approaches you have used. The reality is, of course, more nuanced. **Using your experience working in these practicals**, write a short essay that reflects on the advantages and disadvantages of each approach, and how they can complement each other. **You must use examples from your own group work experience** to substantiate your reflection.

NOTES ON PLAGIARISM

The assignment is by its nature collaborative and you are free to seek help from others and outside sources. **However, you must clearly indicate such instances in your report and give credit to *each and every* person and source you have used**. Plagiarism is presenting work that is not yours as your own and failing to give appropriate credit leads to suspicions of plagiarism that will be reported to the examination board. Also note that the programme's regulations on fraud and plagiarism apply, so the use of tools like chatGPT is not allowed.

GRADING SUMMARY

Line follower challenge (6p)

Controller and PID design	2p
Performance in simulation	2p
Performance on the robot	2p

Object sorting challenge (9p)

Controller and subsumption architecture design	5p
Performance in simulation	2p
Performance on the robot	2p

Report (15p)

Documentation of the implementations	5p
Reflective essay	10p
Maximum grade:	**30p**

CHECKS DURING THE DEMO DAY (SIMULATION AND ROBOT)

During the Demo day, you will show your implementation in simulation and on the real robot to graders. We check for the following specific aspects to determine overall performance, and partial points are possible if only some of these are covered.

Line follower task

- Does the robot stay on the line?
- Is it able to complete the task?
- Is the task implemented using a PID controller?

Object sorting task

- Can the robot distinguish the objects?
- Can the robot push objects in a goal-directed manner?
- Does the robot complete the task?
- Is the task implemented using a subsumption architecture?

HOW TO APPROACH THE PRACTICALS

A large educational aspect of the practicals is learning self-organisation and independence in taking ownership of a project. You will have noticed that the assignment splits into several components that can be carried out either in parallel or in an arbitrary sequence while there are also some dependencies. For example, the object sorting task and the line following task can be worked on independently. Meanwhile, the implementation of the line following task in either simulation or robot requires a basic PID controller.

Your common starting point is 1) familiarizing yourself with CoppeliaSim and the starter files that are provided and 2) building your Lego robot. Beyond this, any team can work as they wish; however, there are aspects you need to take into account:

- There are seven practicals in total, so 28 hours. You can expect to spend 6 hours familiarizing yourself with various aspects, 8 hours on the line follower task, and 14 hours on the object sorting task. These numbers are obviously for guidance only, your mileage might vary.
- Working on the robots is restricted to the practicals. You can, of course, work on your code whenever you wish, but you will only have access to the robot during the practicals.
- This year, we are split over four locations throughout the Maria Montessori building. This means that not all tasks for the physical robots will be available in all locations but at the same time there is not enough room for all groups to be in the same place at the same time. You will need to liaise with the other groups and ensure that you are distributed across the locations.

It is recommended that you spend some initial time as a group reading the assignments here, sketching your own solutions (break the task down into individual components – what depends on what else, who can work on what, how will it all come together) to them, and then making a plan for the implementation given the constraints above.

How you divide the work in your team is up to you. However, **make sure every member contributes to the assignment**. You will need to document everyone's role on the peer review form. It is not mandatory to always be present at every practical, or to stay until the end, but it is clearly advantageous. If you cannot make it to a specific practical due to some constraints, talk to your team members and organise accordingly.

If you wish to make use of your robot, you need to collect the key from a TA in the atrium at the start of the practical and *you must return the robot to the locker and the key to the TA* at the end.

Each location has two TAs to help you. Additional help, communication, announcements, and organisational matters take place on the Discord server. Note that there are guides for frequently encountered problems on the Discord server. Remember to also check those when you run into trouble – you are most likely not the first nor the last to encounter a specific issue, so help can also be found there.

Good luck!