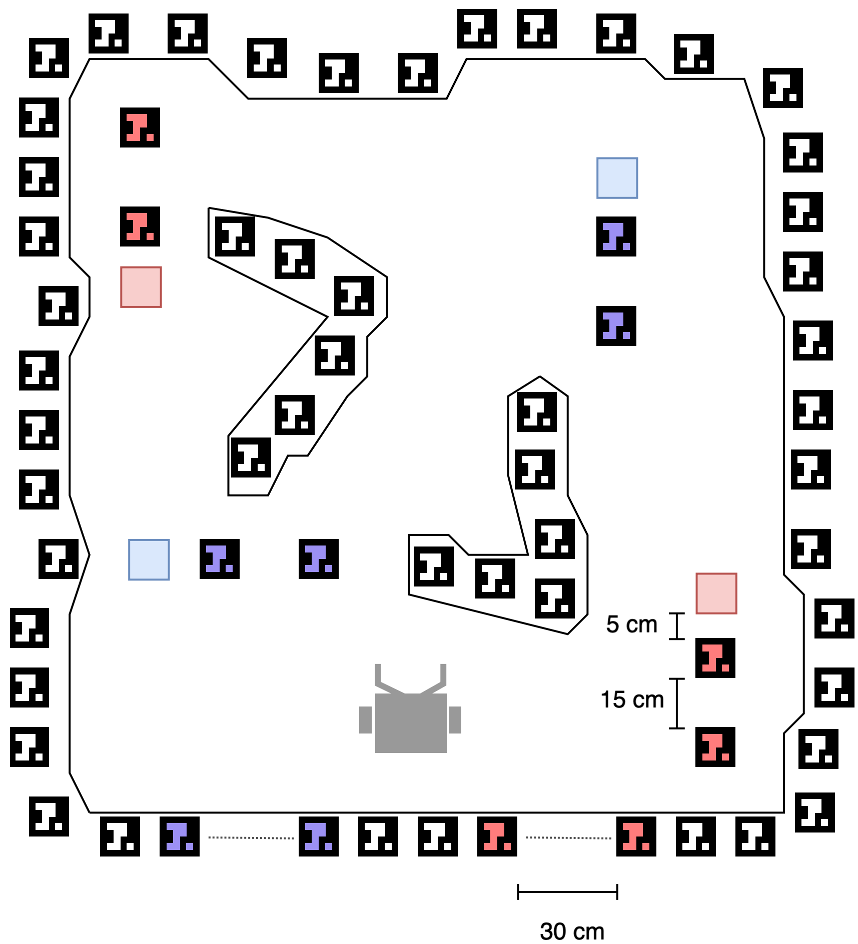
# Practical Course on Computer Vision and Robotics - 2025

Lennart Jahn, Julian Kerl

Winter semester 25/26

## 1 Introduction

This document describes the tasks to complete for successful participation in the practical part of the course *Introduction to Computer Vision and Robotics*. The general task of the practical part is to build and program a robot to fulfil the task given below.




## 2 Task Description

The task is to collect blocks and deliver them to the correct locations. Think of it as a simplified version of many everyday *pic and place* tasks like delivering parcels, or moving materials in a factory from one machine to the next. The task is timed, but there is no strict time limit. We will deduct points if the robot looses time through inefficient behaviour. The blocks have to be collected in an alternating fashion, so if you start with a blue one, the next one has to be red, then blue again, etc. After having picked up a block, the robot has to move them over the line of the respective red and blue gate. Note that pushing the block is fully sufficient, you don't have to lift the block up from the floor.

The task specifications are as follows:

- Aruco markers 0 and lower, and 1000 and higher are not used.
- Aruco markers $\{1, ..., 300\}$ are used for the outside boundary
- Aruco markers $\{301, ..., 400\}$ are used for the obstacles.
- The aruco markers $\{401, ..., 420\}$ are used for the red gate.
- The aruco markers $\{501, ..., 520\}$ are used for the blue gate.
- Blocks are signalled by pairs of markers:
  - The pairs $\{(421, 422), (423, 424), ...(449, 450)\}$ are used for signalling red blocks.
  - The pairs $\{(521, 522), (523, 524), ...(549, 550)\}$ are used for signalling blue blocks.
  - The marker with the lager id in a pair is always closer to the block.
- These definitions apply to the markers on the outer borders and obstacles, respectively:
  - The markers are of size 5 cm
  - The maximum distance between two connected markers is 0.2 m.
- The minimum distance between the border and obstacles is 0.25 m. If it is less, the obstacle connects to the border.
- Crossing the border or obstacles is not allowed.
- The blocks have to be collected in an alternating fashion (if you start with red, the next one has to be blue, then red again, etc. and vice versa). The number of blocks is always even.

## 3 Milestones

To make it easier for you to track your progress throughout the semester, we have divided the project into five milestones. Each milestone is accompanied by Jupyter notebooks, in which you can learn how to implement the necessary sub-tasks for solving the project task.

**Milestone 1**

Deadline: 13th November 2025

- SSH, VScode, robot setup
- (notebook 1) ArUco detection
- (notebook 2) Image (2D) to global coordinate transform (3D)

**Milestone 2**

Deadline: 20th November 2025

- (notebook 3) Basic motor control
- (notebook 4) PID controller

**Milestone 3**

Deadline: 4th December 2025

- (notebook 5) SLAM
- (notebook 6) Driving to a point using the PID controller

**Milestone 4**

Deadline: 8th January 2026

- (notebook 7) Map discretization and optimal paths
- (notebook 8) Planning

**Milestone 5**

Deadline: 29th January 2026

- Apply all these solutions to the task

**Exam**

Exam: TBA

## 4 Examination

The examination consists of a practical part with additional (but related) theory questions from the lecture and a written report. Additionally you have to hand in your code. The requirements for the lecture questions will be set separately.

### 4.1 Practical Exam

The practical exam will consist of one implementation of the general task that your robot will have to solve. The faster/more efficient your robot performs, the better the grade. It is not necessary to complete the full task perfectly to get a passing grade.

The grades are approximately described by the following criteria:

1. Very good: The robot solves the task efficiently, without violating any constraints, or requiring outside intervention.
2. Good: The robot solves the task mostly efficiently. Constraint violations, if they occur, are rare, and can be proven to be the results of extraordinary circumstances and not faulty programming. No intervention is required to solve the task.
3. Fair: The robot manages to solve the task, but with some problems or (small) intervention.
4. Sufficient: The robot requires intervention to complete the task, or frequently violates constraints, e.g. by driving over obstacles or the outer border of the arena.
5. Insufficient (fail): The robot cannot solve the task.

### 4.2 Report

The report should be submitted based on the report template provided. The report has to contain concise information on

- Who did what
- What was exceptionally interesting
- What was hard to do
- What could be changed/done better in the course

### 4.3 Code

Please provide the complete code you wrote for the project. Email the code to us at the end of the course, or invite us to your git repository.