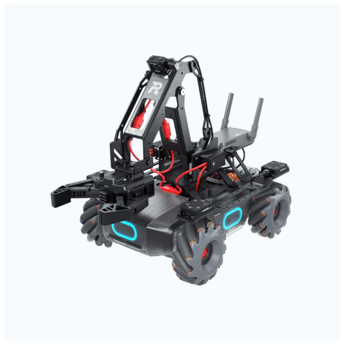
ECE 780 T03 Robot Dynamics & Control
Gennaro Notomista

PROJECT DESCRIPTION

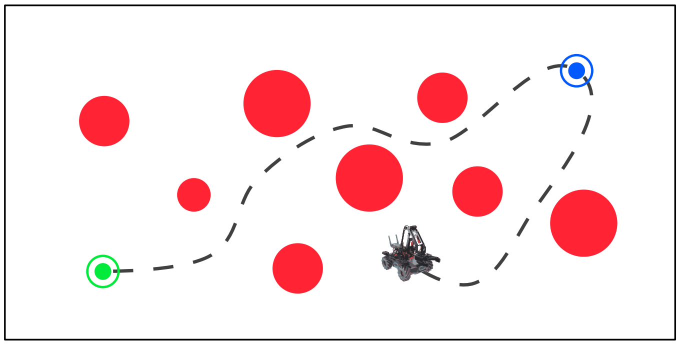
Due date: Jul 28, 2024

Contents

1 Introduction and objective



(a) The mobile manipulator DJI RoboMaster EP used to execute the pick-and-place task.



(b) A pick-and-place scenario with a pick-up location (blue), a drop-off location (green), and obstacles (red).

Figure 1: Mobile manipulator robot and pick-and-place scenario of the project.

Objective

The objective of this project is to program the mobile manipulator DJI RoboMaster EP (Fig. 1a) to perform a pick-and-place task consisting in (see Fig. 1b):

- Navigating to a known pick-up location
- Picking up an object
- Navigating to a known drop-off location
- Droping off the object
- Avoiding obstacles placed in the environment at known locations

The mobile manipulator is comprised of a mobile base controlling using longitudinal and angular velocity inputs, carrying a robotic arm with a gripper mounted on its end-effector. The technical specifications of the robot are available here: `https://www.dji.com/ca/robomaster-ep/specs`.

2 Instructions

The controller must be developed on your PC using Python starting from the files provided on LEARN, under Content/Project/Code. The description of the files is as follows:

- `main.py`: Skeleton script to write project code
- `mobile_manipulator_unicycle.py`: Implementation of the class `MobileManipulatorUnicycle`—inheriting from `Robot` implemented in `robot.py`—to control the base and the arm of the mobile manipulator
- `README.md`: Instructions to get started with the code
- `robot.py`: Implementation of the class `Robot` handling the interface with the backend to receive robot poses from the camera system in the Robohub and to send control inputs to the robot
- `robot_control_comms.py`: Utility classes and functions related to communication with the backend
- `test.py`: Script to test the interface with the backend to send control inputs to the robot and receive its pose from the camera system in the Robohub

Note that you must (and need to) only modify the file `main.py` to solve the project. In the provided `main.py` script, the robot ID has to be set on line 3. For example, if you belong to group 3, then line 3 should be

```
robot = MobileManipulatorUnicycle(robot_id=3, backend_server_ip="192.168.0.2")
```

The object `robot` gives you an interface to the backend—which in turn interfaces with the robot and the tracking camera system of the Robohub—through the following methods:

- `set_mobile_base_speed_and_gripper_power`

Description	Send longitudinal and angular speeds, as well as gripper commands, to the mobile platform
Example of use	`robot.set_mobile_base_speed_and_gripper_power(v=0.1, omega=1.0, gripper_power=1.0)` moves the platform at 0.1 m/s forward, 1.0 rad/s counterclockwise about the local z direction (see Fig. 2 below), and opens the gripper
Example of use	`robot.set_mobile_base_speed_and_gripper_power(v=-0.05, omega=-2.0, gripper_power=-1.0)` moves the platform at 0.05 m/s backward, 2.0 rad/s clockwise about the local z direction (see Fig. 2 below), and closes the gripper

- `set_leds`

Description	Send RGB color to the robot LEDs
Example of use	`robot.set_leds(128, 32, 32)` to set the color of the LEDs to (128, 32, 32) in RGB format

- `set_arm_pose`

Description	The two input arguments control the position of the end effector
Example of use	`robot.set_arm_pose(25.0, 25.0)` moves the arm 25 mm forward and 25 mm upward

- `get_poses`

Description	Get poses of the robot, pick-up and drop-off locations, as well as of obstacles, in the global reference frame of the Robohub (see Fig. 2 below). Each pose is a list of x, y position coordinates, and orientation θ (e.g. `robot_pose` is the list containing the position and orientation of the robot, $[x_\mathrm{R}, y_\mathrm{R}, \theta_\mathrm{R}]$)
Example of use	`robot_pose, pickup_location, dropoff_location, obstacle_1_pose, obstacle_2_pose, obstacle_3_pose = robot.get_poses()`

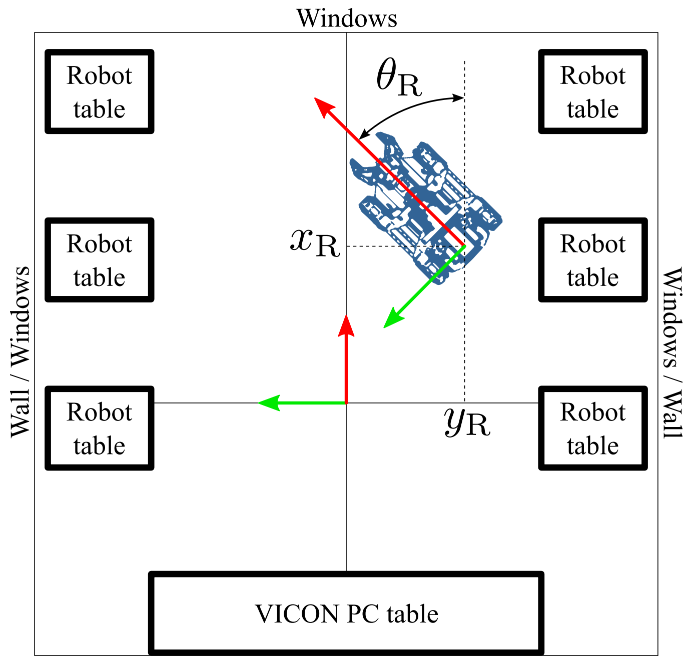


Figure 2: Global reference frame in the Robohub (red and green arrows are x and y directions) and local reference frame of the robot (red and green arrows centered at the robot).

Tasks

Using the provided code, complete the following steps to achieve the pick-and-place task:

(T1) Assuming the mobile manipulator is at a known pick-up (resp. drop-off) location, write a function to pick up (resp. drop off) an object at that location

(T2) Design control Lyapunov functions (CLFs) to drive the robot to known pick-up and drop-off locations

(T3) Design control barrier functions (CBFs) to avoid obstacles placed at known locations in the environment

(T4) Synthesize the control input for the mobile platform using a quadratic program (QP) formulation to achieve safe navigation

(T5) Combine the controllers for the manipulator arm and the mobile platform developed above to realize the desired pick-and-place task

3 Performance evaluation

The robot should be able to

- Navigate to a known pick-up location
- Pick up an object
- Navigate to a known drop-off location
- Drop off the object
- Avoid obstacles placed in the environment at known locations

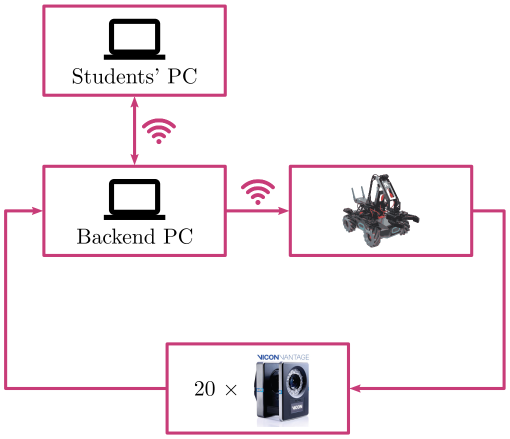
Recall that the deliverables related to the project are as follows (including percentage of the overall course grade):

- Midterm project report: 10%
- Final project report: 20%
- Project code: 10%

More details on the format and the structure of the reports can be found on the syllabus.

4 Robohub schedule

(a) A humanoid, a mobile manipulator, and the ceiling camera system in the Robohub.



(b) Block diagram of the closed-loop control of the mobile manipulators in the Robohub.

Figure 3: The Robohub at University of Waterloo.

The Waterloo Robohub (Fig. 3a) is a collaborative robotics research facility located on the ground floor of Engineering 7. It hosts a diverse fleet of robots (humanoids, manipulators, ground and aerial mobile platforms), and it is equipped with an indoor positioning system comprised of a set of 20 Vicon Vantage V5 cameras. The RoboMaster EP robots are controlled according to the feedback control loop shown in Fig. 3b.
The group schedule to perform project-related activities in the Robohub is reported in the table below (as well as in the course syllabus).

Date	**Time**	**Activity (suggested)**
Jun 5	11:30–13:30	Intro to RoboMaster EP
Jun 19	13:30–15:30	Object pick-up and drop-off
Jul 17	11:30–13:30	Optimization-based control (navigation)
Jul 24	11:30–13:30	Optimization-based control (safety)