COMP0127 Robotic Systems Engineering
Coursework 2: Jacobian, Inverse Kinematics and Path Planning

Dr. Agostino Stilli
Department of Computer Science
University College London

November 14, 2022

To get full credit for an answer, you are *required* to provide a .pdf report, and a fully working coding solution by filling in the provided code templates. These templates provide additional information on how to implement each script. **Do not remove anything from the templates and try to only fill in the code in the specified fields.** For the coding questions, you are also expected to include a simple breakdown of your algorithms in the report. When ready, *upload* your 'cw2' package on Moodle along with your submitted coursework report, in .zip or .rar extension. The necessary ROS packages are available on the course's *GitHub repository*.

Jacobian and Inverse Kinematics

1. How many inverse kinematic solutions exist for a 2D 4R-planar manipulator, if an achievable pose of the end-effector x_e is given? Give a full explanation to support your answer. [report - 5 pts]

2. Suppose that the robot is moving in free space (i.e. there are no obstacles) and that more than one inverse kinematic solution exists for a desired pose of the end-effector x_e, what criteria should you consider when choosing an optimal solution? [report - 5 pts]

3. When is the output of the function atan2(y, x) different from atan$(\frac{y}{x})$? [report - 5 pts]

4. Complete the following tasks by filling in the "cw2q4/youbotKineStudent.py" python code template. A simple code breakdown in the report is required for all subquestions, except subquestion b which is report only. In the cw2q4 folder, you can find three files.

    - youbotKineStudent.py: This is the coding template for the questions below.
    - youbotKineBase.py: This file includes a class implementing common methods you may need to call in order to solve the questions below. You should not edit this file.
    - youbotKineKDL.py: This file includes a class providing implementations to the questions below in KDL in order to check your own solutions. You should not edit this file.

a. Write a script to compute the Jacobian matrix for the YouBot manipulator. [report - 2 pts, code - 10 pts]

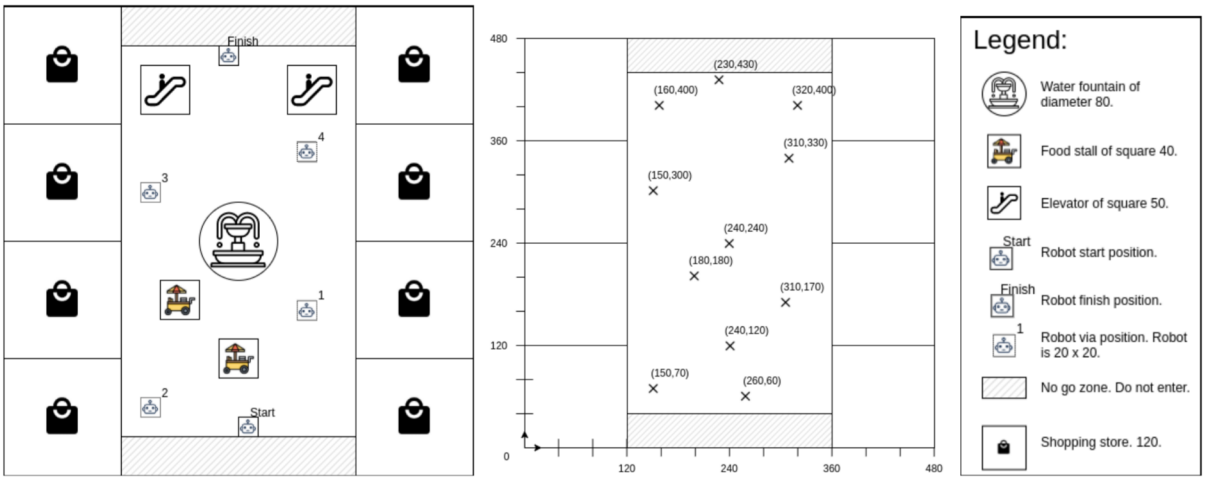
b. Derive the closed-form inverse kinematics solutions for the YouBot manipulator. You can represent any non-zero length parameters as variables. [report - 8 pts]

c. Write a script to detect singularity in any input pose. [report - 1 pts, code - 4 pts]

[25 pts]

Path and Trajectory Planning

5. Assume the following scenario: A shopping mall is testing autonomous cleaning robots to clean the floor of a section of the mall. They have given you the following floor map defining no-go zones and cleaning via points, and obstacles. Cleaning will occur at night, so no dynamic obstacles will be present. Consider only passing through via points, not total floor coverage. They also allow external cameras and tracking, so perfect odometry can be assumed. Present a robotic solution by choosing a drive system to complete the task, addressing path planning and trajectory planning. You can assume the position and orientation of your chosen robot is given as $q = (x, y, \theta)$ where x and y describe the position and θ describes the orientation of the robot. You have a perfect controller that can control your wheel velocities to attain the given position and orientation. When choosing the drive system, consider that the vacuum is located at the back of the robot, while the brush is at the front, therefore, depending on how the robot handles turns and rotations, the vacuum may miss water pickup. Specifically, your solution should address the following. (The recommended answer is up to 50 words per sub-question and the word count for the entire question should not exceed 300 words.)



(a) Map (b) Coordinates (c) Legend

Figure 1: Map, coordinates, and legend of shopping mall cleaning area.

a. Present your drive system configuration and explain your reasoning. Is it holonomic or non-holonomic and why? What is the configuration space? [report - 5 pts]

b. Describe how you would find a path that goes from the starting point to the final point while passing through all the via points in order. How would planning change if the ordering did not matter? [report - 5 pts]

c. Describe a time scaling function you would use when constructing a trajectory. Consider initial and final accelerations as well as at the via points. [report - 5 pts]

d. The robot is designed with a water vacuum at the back of the robot, meaning during turns and rotations, the vacuum may miss water pick-up. What ways can you design your path or trajectory to prevent left-over water? [report - 5 pts]

e. Describe a path-planning approach for full floor coverage while avoiding obstacles. [report - 5 pts]

[25 pts]

6. Complete the following question by filling in the 'cw2q6/cw2q6_node.py' python code templates to perform path planning via the shortest path. A simple code breakdown in the report is required for all subquestions, except subquestion e, which is code only. You are given target joint positions in the bagfile 'data.bag'. Your task is for the Youbot end-effector to reach each target Cartesian check-point associated with each target joint position, via the shortest path in Cartesian space. To solve the question you need to:

   a. Implement the "load_targets()" method that loads the target joint positions from the bagfile and calculates the target end-effector position. [report - 2 pts, code - 3 pts]

   b. Implement the "get_shortest_path()" method that takes the checkpoint transformations and computes the order of checkpoints that results in the shortest overall path. [report - 3 pts, code - 5 pts]

   c. Implement the "decoupled_rot_and_trans()" and "intermediate_tfs()" methods that take the target checkpoint transforms and the desired order based on the shortest path sorting, and create intermediate transformations by decoupling rotation and translation. [report - 2 pts, code - 5 pts]

   d. Implement the "ik_position_only()" and "full_checkpoints_to_joints()" methods that take the full set of checkpoint transformations, including intermediate checkpoints, and compute the associated joint positions with position-only inverse kinematics. [report - 2 pts, code - 8 pts]

   e. Implement the "q6()" method, the main method of this question, where other methods are called in order to perform the path planning task. [code - 5 pts]

[35 pts]

END OF COURSEWORK