Project 01 - Robot Kinematics

MTRN4230 - UNSW School of Mechanical and Manufacturing Engineering

July 3, 2024

Changelog

- 30/06/24: Initial release
- 03/07/24: Cleared up wording of location to include in report for part C

1 Learning Outcomes

- Demonstrate understanding of forward kinematics, its calculations, and its uses.
- Demonstration understanding of the Jacobian, its calculation, and its uses.
- Demonstrate understanding of singularities that can be experienced by the robot.
- Demonstrate understanding of the inverse kinematics

2 Your task

Within this assignment you must complete the five tasks (Part A-E) below. These will include demonstrations that must be conducted during your allocated tutorial slot during the prescribed week. In addition, you will complete a report that documents the work completed for these tasks and the outcomes and findings. You will also find questions to answer in your report.

2.1 Part A: Dynamic forward kinematics

When working with robotic arms, such as the UR5e, it provides continuous feedback of the state of the arm. In particular, there are readouts for the current joint configuration (the angle of each individual joint) as well as the end effector pose. The former may be directly measured from the physical robot but the same cannot generally be done for the latter. As a result, it is likely that calculations are performed continuously to convert the joint configuration into a pose; relying on forward kinematics.

Your task is to complete the code provided in "`assignment1PartA.m`". This program has the user enter a series of joint configurations that the robot must move to (with all joint angles presented in degrees). These are assumed to be previously recorded from the robot. It then calls a function (which you must implement) that converts these configurations to poses and uses the resulting values to make robot movements.

Your task is to implement the function "`convertJointToPose`" that takes in a joint configuration of a UR5e arm (i.e., an array of six joint angles) and converts this to a robot pose. To complete this you are not able to use the RTDE toolbox and must rather perform the calculation manually and dynamically from first principles.

You must not change any code in the provided file other than the function and the header comments. Your functionality will be assessed by the tutor during an in-class assessment in week 8 (this will be run in your assigned lab class).

In reality a program like this would more likely calculate this dynamically on the physical robot when it is moved around to make conversions as required. However, for simplicity in this assignment the joint configurations are being entered in the terminal.

You do not need to include anything in your report for this practical part of the assessment.

2.2 Part B: UR5e modelling

For this part you must demonstrate the underlying mathematics that you utilised within part A. Manually calculate the forward kinematic solution for a provided joint configurations (home configuration below). This must then be verified using the RVC toolbox and URSim.

Home joint configuration: $[0.00, -75.00, 90.00, -105.00, -90.00, 0.00]$

All intermediate and final results from this should be put into your report.

As part of this, complete the following:

1. Manually calculate the forward kinematic solutions for the joint position (you may use the function you developed in the previous part to complete this).
   (a) Provide the resultant homography matrices as well as the output pose (with the angles provided in rpy configuration (see the "`tr2rpy`" function from the RTB manual to help with this conversion)
   (b) You are required to show your full workings for the calculation (these must be typed out and not as photographs).
   (c) Show also the results of all intermediary matrices (0T1, 0T2, ..., 0T6)
   (d) Explain the meaning of these matrices calculated
2. Construct a model of the UR5e robotic arm using the RVC toolbox in MATLAB. The DH-Table has been provided in Table 1.
   (a) Perform the forward kinematic conversion (using fkine) using this to attain the pose with the angles in RPY configuration
   (b) Insert the results of the matrix results and converted result in your report
3. Validate the calculations from (1) and (2) by moving the robot within the URSim virtual machine to the supplied home configuration
   (a) Take a screenshot showing the pose including the rotation in rpy representation

Table 1: DH table for UR5e robot arm

	theta (rad)	**a (m)**	**d (m)**	**alpha (rad)**
Joint 1	0	0	0.1625	$\pi/2$
Joint 2	0	-0.425	0	0
Joint 3	0	-0.3922	0	0
Joint 4	0	0	0.1333	$\pi/2$
Joint 5	0	0	0.0997	$-\pi/2$
Joint 6	0	0	0.0996	0

2.3 Part C: Robot Speed Limits

You are tasked with performing a safety check on a robot implementation. You have been provided with the output of a series of movej commands (particularly the joint positions and joint velocities). Using this you must calculate the maximum linear velocity achieved during the movement. That is $v = \sqrt{v_x^2 + v_y^2 + v_z^2}$.

This should be implemented as a function "`calculateMaxLinearVelocity`" in the file "`calculateMaxLinearVelocity.m`" provided. You may make use of the RVC toolbox during the implementation. **DO NOT change the function name or any input or output parameters.**

You have also been provided the file "`assignment1PartC.m`" to test your code. This will not be submitted so you are free to change this as much as you want.

In your report, explain your approach to this problem and the mathematics behind the function you have written. As part of this write out the Jacobian calculation for the first location provided in the MATLAB script.

2.4 Part D: Robot Singularities

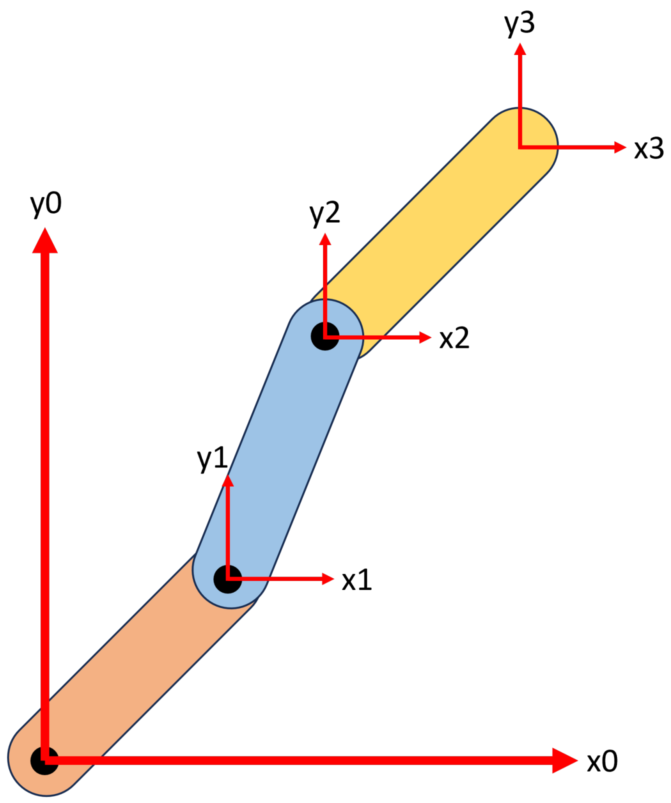

Figure 1: Robotic arm design for singularities

Consider the robotic arm displayed in figure 1. You are required to calculate all the singularity positions that it possesses. To complete this, follow the steps below. All stages and working must be included within your report. Any calculation conducted with Matlab should be indicated and the result shown.

You may assume that all links are of unit length (1 m).

1. Determine the DH matrix.
2. Calculate the Jacobian which relates joint velocities to linear velocities.
3. For what value(s) is the manipulator at a singularity? What motion is restricted at this singularity? What type of singularity is experienced?

2.5 Part E: Inverse Kinematics

Inverse kinematics is the opposite of forward kinematics that we have already looked at. While forward kinematics calculates the end effector pose from a set of joint angles, the inverse kinematics finds the joint configuration given an input pose. For the robot in figure 2, you must calculate the analytical (algebraic) inverse kinematic solution. Ensure you show all steps in this calculation within your report.

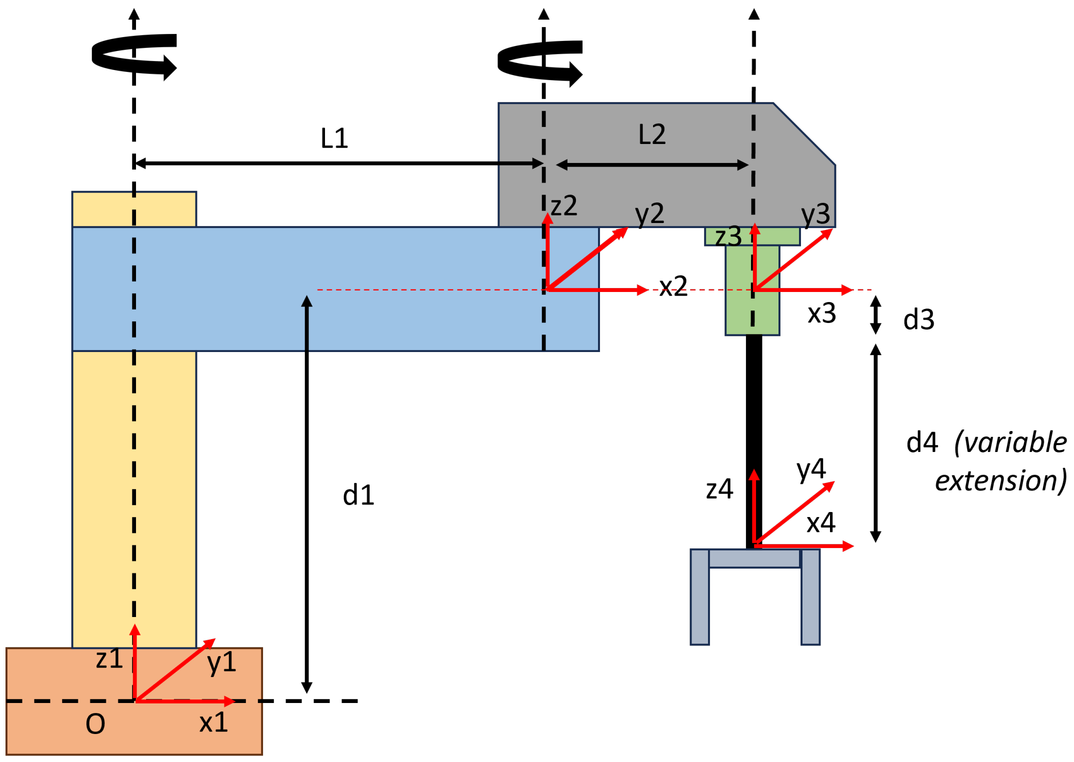

Figure 2: Robotic arm design for Inverse Kinematics

This method of solving the inverse kinematics analytically (algebraic) is often not possible for more complex robot arms (such as the UR5e). Explain in detail (you do not need to provide any mathematics here) two alternative methods that could be undertaken to find an inverse kinematic solution.

3 Marking Criteria

This assignment is worth 20 % of the total course mark. The breakdown of marks for this assignment is described in table 2.

Table 2: Mark allocation for assignment

Criteria	Weighting	Description
Part A	25 %	Practical demonstration of functioning code during your lab session.
Part B	25 %	Correct calculations of forward kinematics for the provided joint configuration. This includes appropriate validation of results using the RVC toolbox and URSim (including screenshots where appropriate). You must provide all intermediary steps for full marks.
Part C	25 %	Code correctly indicates safety (correct maximum velocity) for variety of tested inputs. Report also includes required calculation.
Part D	15 %	Complete mathematical solution provided in report for the calculation of the singularity of the provided robotic arm. This must include all intermediary calculations and solutions. Additional questions posed in this part must also be addressed.
Part E	10 %	Complete algebraic calculations provided for inverse kinematics of prescribed SCARA arm. This must include all intermediary calculations and solutions. Additional questions posed in this part must also be addressed.

4 Submission

This assignment (the report) is due by 11:59 PM on Monday of Week 09 (22/07/24). It should be submitted via the submission box provided on Moodle under the "Assessments Hub" section. This will require you to submit your report as a PDF document along with code files for both part A and part C (which must follow the naming as prescribed in the spec).

5 Plagiarism

If you are unclear about the definition of plagiarism, please refer to What is Plagiarism? — UNSW Current Students. You could get zero marks for the assignment if you were found:

- Knowingly providing your work to anyone and it was subsequently submitted (by anyone), or
- Copying or submitting any other persons' work, including code from previous students of this course (except general public open-source libraries/code). Please cite the source if you refer to open source code.

You will be notified and allowed to justify your case before such a penalty is applied.