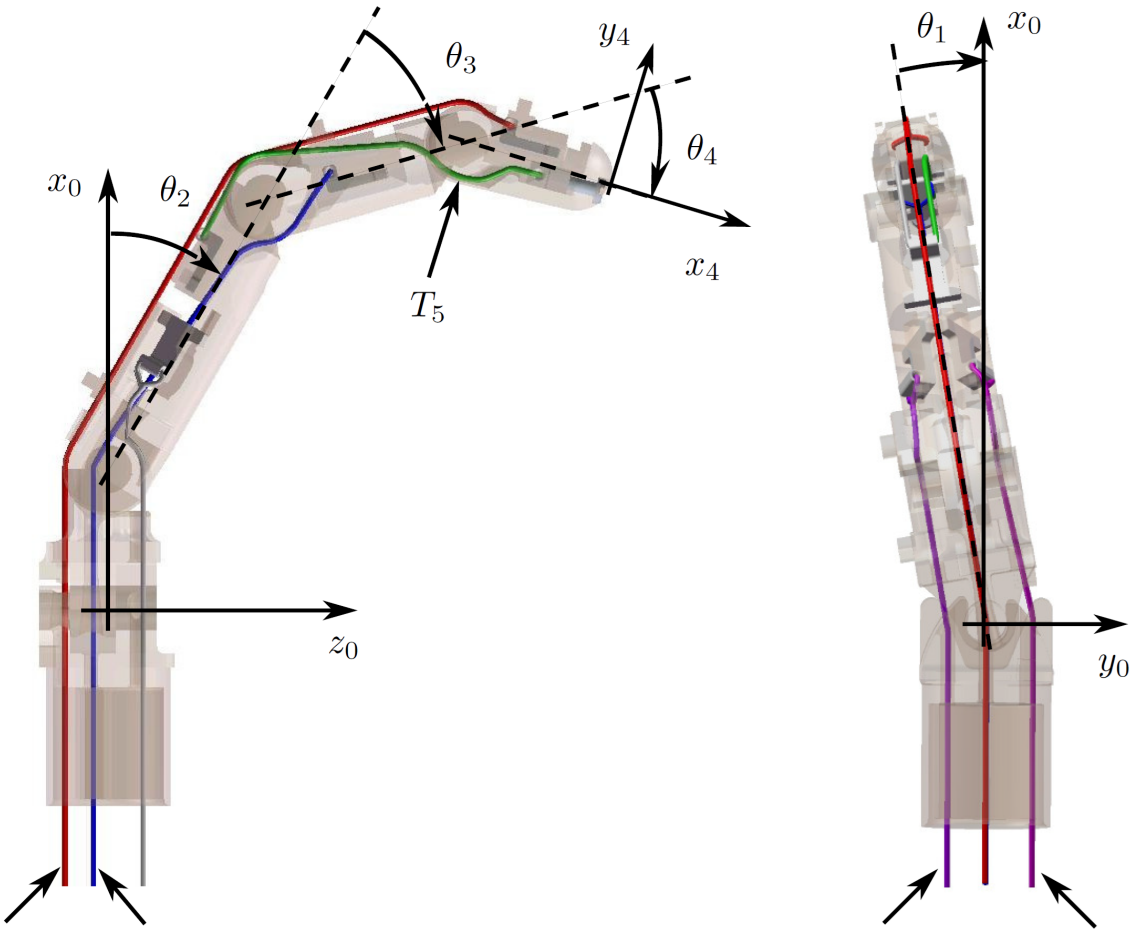
Aristotle University of Thessaloniki

Robotic Finger Modeling Project

Personal Report

Charalampos Inglezos



(Cover image from [1])

Thessaloniki,
June 10, 2024

Contents

List of Tables

List of Figures

1 Introduction

This group project is divided into four main sections: theoretical-mathematical modeling, MATLAB implementation, user interface (UI) interaction, and MQTT server for transmitting data to Team B for visualizing the equivalent of the finger in a Unity environment simulation. My contribution in this project concerns the forward and inverse kinematics, a low-level MATLAB implementation using the results of the first section I found in order for me to verify all the work I had done in the previous step, and a simple user interface creation in the MATLAB App Designer that uses the code I created in MATLAB, which helps visualize the robotic finger into an end-to-end pipeline through a UI, which paves the path for the complete UI implementation by the rest of my team. Everything I describe in the following sections is my own personal work. The diagrams and images were designed manually using Blender and "draw.io".

2 Mathematical Modeling

In order to comprehend, study and simulate the nature of a robotic finger and how it operates, we have first to model it, both mathematically (as equations at theoretical level) and algorithmically in a more practical way (simulation, visualization). The local reference frames and rotational axes for each joint were selected to derive the modified Denavit-Hartenberg (DH) parameters (r_i, α_i, d_i, θ_i). The modified DH parameters are more intuitive and simpler to work with, since they use the current z-axis of the joint, and there is no need for further transformations. In addition, MCP_{aa} was selected as the first joint and its local frame was matched with the base frame 0, which means that upward is the z-axis and the finger extends its length along the x-axis. The next joint is MCP_{fe} and then PIP and DIP, which all three have the same rotational direction. This selection of reference frames and the order of the angles introduce only once a $\pi/2$ angle into the parameters (in addition to the normal θ_i angles of each joint) and simplify the subsequent equations. Figure 1 shows the simplified robotic finger system with reference frames, axes, lengths and angles, while Table 1 shows the derived modified DH parameters.

Table 1: DH Parameters Table

Joint i	$\boldsymbol{r_i}$	$\boldsymbol{\alpha_i}$	$\boldsymbol{d_i}$	$\boldsymbol{\theta_i}$
1 (MCP_{aa})	0	0	0	$\theta_{MCP_{aa}}$
2 (MCP_{fe})	0	$\pi/2$	0	$\theta_{MCP_{fe}}$
3 (PIP)	L_1	0	0	θ_{PIP}
4 (DIP)	L_2	0	0	θ_{DIP}
5 (*end effector*)	L_3	0	0	0

2.1 Forward Kinematics

Given the lengths L_i of the three phalanges and the angles θ_i of all four joints, the forward kinematics problem is to specify the final position and orientation of the end

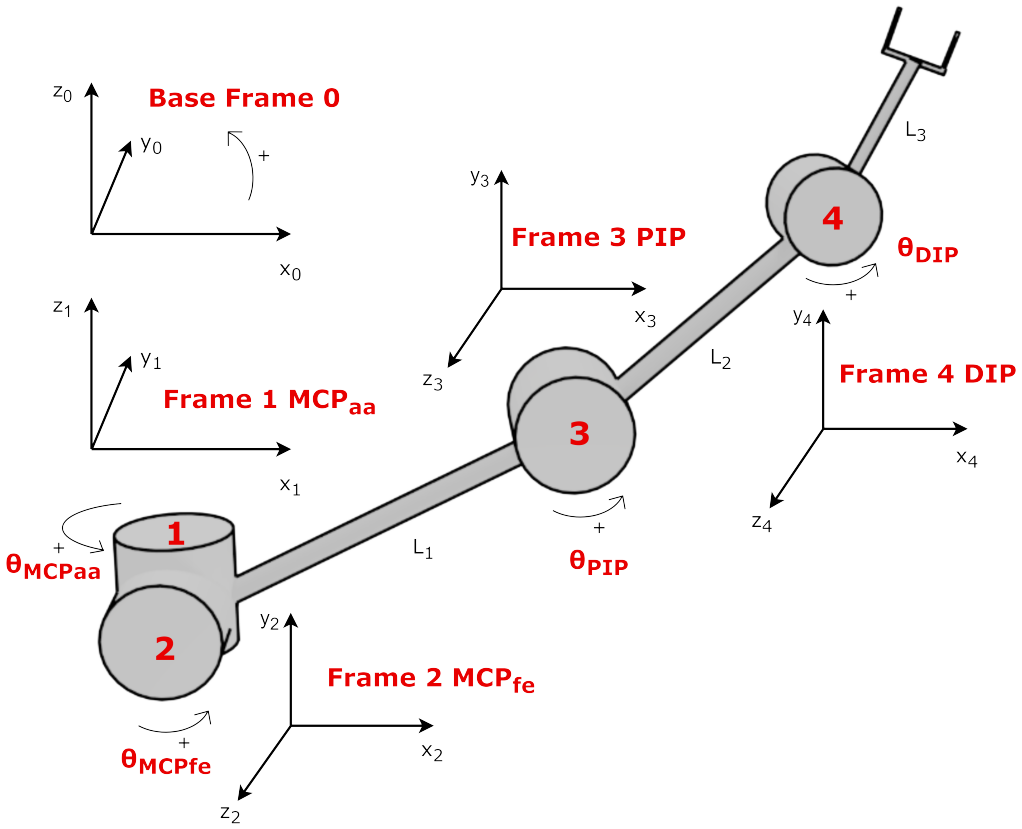


Figure 1: Simplified robotic finger representation with frames, designed in Blender.

effector (fingertip) [1–6].

The general transformation matrix between two consecutive joints is given by (1):

$$T_{i-1}^{i} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) & 0 & r \\ \cos(\alpha)\,\sin(\theta) & \cos(\alpha)\,\cos(\theta) & -\sin(\alpha) & -d\,\sin(\alpha) \\ \sin(\alpha)\,\sin(\theta) & \sin(\alpha)\,\cos(\theta) & \cos(\alpha) & d\,\cos(\alpha) \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{1}$$

Applying the transformation for all four joints and for the translational end effector as well using equation (2), the final transformation matrix of the end effector is calculated (3) with respect to the base frame 0, which gives us the solution (equations (4)) of the forward kinematics for the end effector (eff).

$$T_0^{eff} = T_0^1 * T_1^2 * T_2^3 * T_3^4 * T_4^{eff} \tag{2}$$

$$T_0^{eff} = \begin{pmatrix} R_{11} & R_{12} & R_{13} & P_{14} \\ R_{21} & R_{22} & R_{23} & P_{24} \\ R_{31} & R_{32} & R_{33} & P_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3}$$

where:

$$
\begin{aligned}
R11 &= \cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\cos\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
R12 &= -\sin\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\cos\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
R13 &= \sin\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
P14 &= \cos\left(\theta_{\mathrm{MCP_{aa}}}\right)\left(L_2\cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}}\right) + L_1\cos\left(\theta_{\mathrm{MCP_{fe}}}\right) + L_3\cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\right)\\
R21 &= \cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\sin\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
R22 &= -\sin\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\sin\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
R23 &= -\cos\left(\theta_{\mathrm{MCP_{aa}}}\right)\\
P24 &= \sin\left(\theta_{\mathrm{MCP_{aa}}}\right)\left(L_2\cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}}\right) + L_1\cos\left(\theta_{\mathrm{MCP_{fe}}}\right) + L_3\cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\right)\\
R31 &= \sin\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\\
R32 &= \cos\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)\\
R33 &= 0\\
P34 &= L_2\sin\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}}\right) + L_1\sin\left(\theta_{\mathrm{MCP_{fe}}}\right) + L_3\sin\left(\theta_{\mathrm{MCP_{fe}}} + \theta_{\mathrm{PIP}} + \theta_{\mathrm{DIP}}\right)
\end{aligned}
$$

$$
\begin{aligned}
x_{eff} &= P14\\
y_{eff} &= P24\\
z_{eff} &= P34
\end{aligned}
\tag{4}
$$

A challenge I faced during forward kinematics was to understand the DH parameters and how the reference frames should be set and what values should be assigned as parameters, which is the first but crucial step in the methodology of robotics. Also, the proper selection of reference frames is very important as a convention, because it affects the simplicity of the derived equations after the transformation matrices calculations.

2.2 Inverse Kinematics

On the contrary, given the end position of the fingertip and its orientation, the inverse kinematics problem seeks to find all four angles that correspond to that end state (position and orientation). Solving the inverse kinematics problem in closed-form algebraic equations is often a difficult task, especially if the joints are too many or the system architecture is complex. Usually, computational numeric methods are applied iteratively to approximate the solutions for the angles. Apart from numeric approaches, there are two ways to solve the inverse problem in closed form equations, either algebraically or geometrically [2, 3, 7–9]. I opted for the geometric approach, which helps visualize the system comprehensively and perhaps find the solution in a simpler way. The following diagrams in Figure 2 show the top-view (x-y) and the side-view (x-z) planes of the finger, always with respect to the base frame of reference, which means that the local rotational z-axes for the last three joints (MCP_{fe}, PIP, DIP) are aligned with the $-y$ axis of the base frame 0. To visualize in space all this system configuration and write down equations describing each joint, equations originating from geometrical analysis, was quite challenging and fascinating, since a clear and comprehensive diagram depicting everything is of utmost importance. Also, apart from the end position, the end orientation of the fingertip is required, which here translates to the cumulative angle on the x-z plane with respect to the x-axis of the base frame 0. This angle, let's denote it as ϵ, is equal to:

$$\epsilon = \epsilon_{234} = \theta_{MCP_{fe}} + \theta_{PIP} + \theta_{DIP} \tag{5}$$

Therefore, given the x, y, z and the angle ϵ, the inverse kinematics problem must be solved. In order for the equations to be more concise, the following notation will be used:

$$\begin{aligned} x &= x_{eff} && \text{(given)} \\ y &= y_{eff} && \text{(given)} \\ z &= z_{eff} && \text{(given)} \\ \theta_1 &= \theta_{MCP_{aa}} \\ \theta_2 &= \theta_{MCP_{fe}} \\ \theta_3 &= \theta_{PIP} \\ \theta_4 &= \theta_{DIP} \\ \theta_{234} &= \theta_2 + \theta_3 + \theta_4 = \epsilon && \text{(given)} \end{aligned} \tag{6}$$

Starting from the top-view diagram, for the triangle formed by r_1, x and y:

$$\tan(\theta_1) = \frac{y}{x} \implies \boxed{\theta_1 = \arctan\left(\frac{y}{x}\right)} \tag{7}$$

$$r_1 = \sqrt{x^2 + y^2} \tag{8}$$

Moving on to the side-view diagram, the following equations apply:

$$\begin{aligned} \phi_2 &= \phi_1 + \phi_4 + \theta_2 \\ r_2 &= z \end{aligned} \tag{9}$$

$$r_3 = \sqrt{r_1^2 + r_2^2} \implies r_3 = \sqrt{x^2 + y^2 + z^2} \tag{10}$$

$$\tan(\phi_2) = \frac{r_2}{r_1} \implies \phi_2 = \arctan\left(\frac{z}{\sqrt{x^2 + y^2}}\right) \tag{11}$$

$$\begin{aligned} r_5 &= L_3 \sin(\epsilon) \\ r_6 &= r_2 - r_5 \\ r_8 &= L_3 \cos(\epsilon) \\ r_7 &= r_1 - r_8 \end{aligned} \tag{12}$$

$$\tan(\phi_{r_4}) = \frac{r_6}{r_7} \implies \phi_{r_4} = \arctan\left(\frac{z - L_3 \cdot \sin(\epsilon)}{\sqrt{x^2 + y^2} - L_3 \cdot \cos(\epsilon)}\right) \tag{13}$$

Applying the cosine rule in the triangle $\overset{\triangle}{ABC}$ and using the properties of $\overset{\triangle}{AEC}$:

$$L_2^2 = L_1^2 + r_4^2 - 2L_1 r_4 \cdot \cos(\phi_1) \tag{14}$$

$$r_4^2 = L_1^2 + L_2^2 - 2L_1 L_2 \cdot \cos(\phi_3) \overset{\phi_3 = \pi - \theta_3}{\implies} r_4^2 = L_1^2 + L_2^2 + 2L_1 L_2 \cdot \cos(\theta_3) \tag{15}$$

$$\begin{aligned} r_4^2 &= r_7^2 + r_6^2 = (r_1 - r_8)^2 + (r_2 - r_5)^2 \\ \implies r_4^2 &= x^2 + y^2 + z^2 + L_3^2 - 2L_3(\sqrt{x^2 + y^2} \cdot \cos(\epsilon) + z \cdot \sin(\epsilon)) \end{aligned} \tag{16}$$

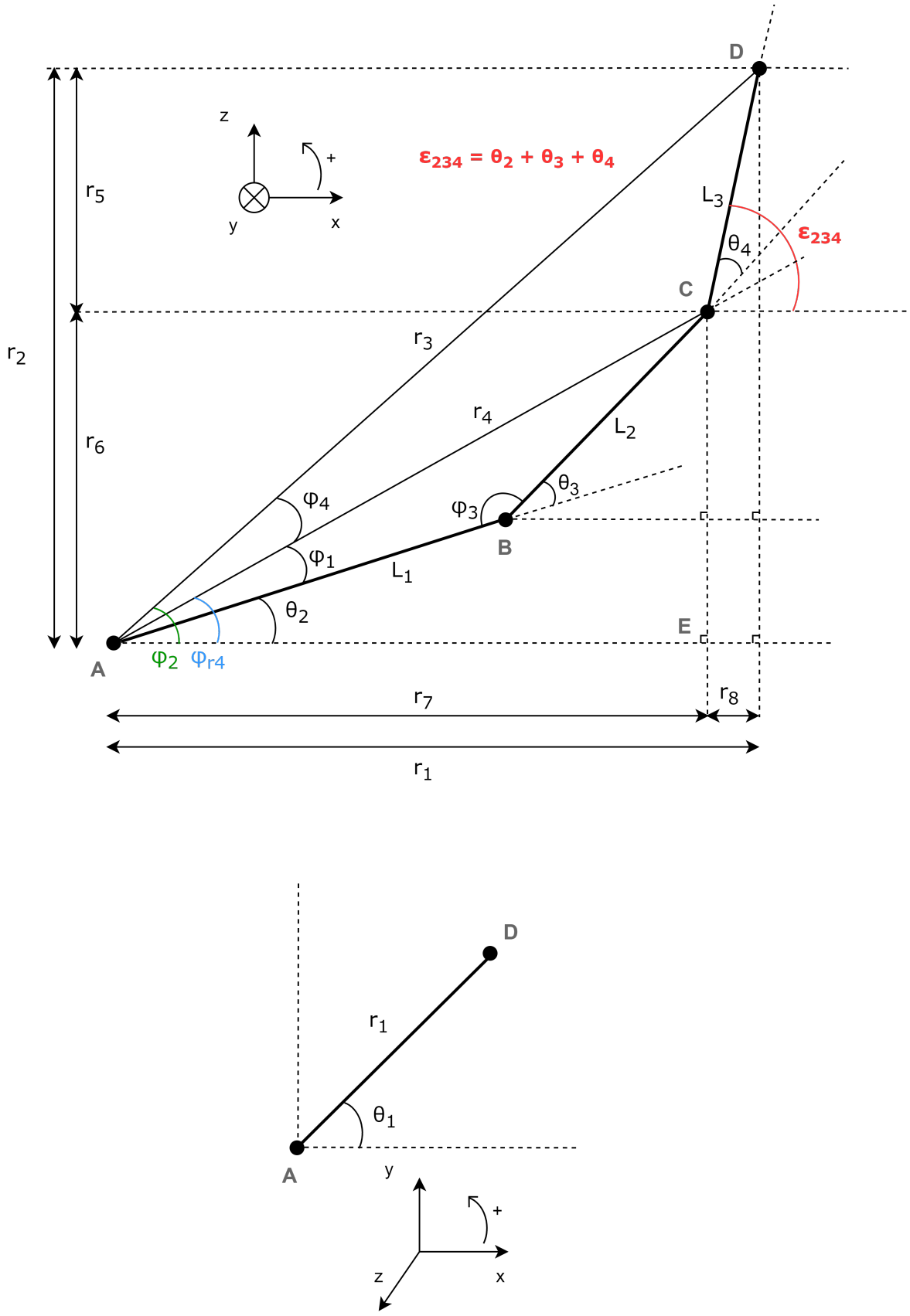


Figure 2: Extensive top-view (x-y plane) and side-view (x-z plane) depiction of the robotic finger, designed in "draw.io", for solving the inverse kinematics problem with geometric approach.

Using the above equation (16), we can now solve for θ_3 in equation (15):

$$\boxed{\theta_3 = \arccos\left(\frac{x^2+y^2+z^2+L_3^2-L_1^2-L_2^2-2L_3(\sqrt{x^2+y^2}\cdot\cos(\epsilon)+z\cdot\sin(\epsilon))}{2L_1L_2}\right)} \tag{17}$$

$$\phi_4 = \phi_2 - \phi_{r_4} = \arctan\left(\frac{z}{\sqrt{x^2+y^2}}\right) - \arctan\left(\frac{z-L_3\cdot\sin(\epsilon)}{\sqrt{x^2+y^2}-L_3\cdot\cos(\epsilon)}\right) \tag{18}$$

From equation (14) we can now solve for ϕ_1, using equation (16):

$$\phi_1 = \arccos\left(\frac{x^2+y^2+z^2+L_1^2+L_3^2-L_2^2-2L_3(\sqrt{x^2+y^2}\cdot\cos(\epsilon)+z\cdot\sin(\epsilon))}{2L_1\sqrt{x^2+y^2+z^2+L_3^2-2L_3(\sqrt{x^2+y^2}\cdot\cos(\epsilon)+z\cdot\sin(\epsilon))}}\right) \tag{19}$$

From equations (18) and (19), we can now solve for θ_2:

$$\theta_2 = \phi_2 - \phi_1 - \phi_4 \tag{20}$$

$$\boxed{\begin{aligned}\implies \theta_2 = &\arctan\left(\frac{z-L_3\cdot\sin(\epsilon)}{\sqrt{x^2+y^2}-L_3\cdot\cos(\epsilon)}\right)\\ &-\arccos\left(\frac{x^2+y^2+z^2+L_1^2+L_3^2-L_2^2-2L_3(\sqrt{x^2+y^2}\cdot\cos(\epsilon)+z\cdot\sin(\epsilon))}{2L_1\sqrt{x^2+y^2+z^2+L_3^2-2L_3(\sqrt{x^2+y^2}\cdot\cos(\epsilon)+z\cdot\sin(\epsilon))}}\right)\end{aligned}} \tag{21}$$

Finally, after having calculated θ_2 and θ_3 in equations (21) and (17) respectively, and since we know ϵ (given), we can now also solve for θ_4, which will be the last angle to be calculated.

$$\boxed{\theta_4 = \epsilon - \theta_2 - \theta_3} \tag{22}$$

3 MATLAB Code Implementation

To verify all the above, I deemed it prudent to develop MATLAB scripts and functions that implement such example states of the robot finger, mostly the inverse kinematics followed by a confirmatory forward kinematics. All the relative MATLAB code files have been uploaded in my personal Github repository, along with detailed code documentation and comments. Essentially, I have incorporated the transformation matrices equations and the inverse kinematics angles equations into the code for solving both forward and inverse kinematics problem as low-level implementation (while on the contrary, a high-level implementation would use Simulink-Simscape or even ROS).

For example, "forward_kinematics.mlx" script calculates the end-effector's position and orientation for given joint angles and conversely, "inverse_kinematics_equations.mlx" script determines the joint angles needed for a desired end-effector state (position and orientation-angle). Literature suggests [2,10–12] that there are some specific angle ranges for each joint per finger that are considered as valid and are widely accepted, therefore a validity check shall be included for the angles found during inverse solution. Keeping

the minimum and maximum values across all fingers (we did not specify a finger but rather we have considered the general case of "index, middle, ring, little" fingers, except for thumb. As future work, the valid range could be further narrowed down for a specific finger, or even include the thumb as well. These constraints are imposed on the angles using "check_valid_angles.m". The ranges used are shown in Table 2.

Table 2: Generalized valid angle ranges for joints.

Degrees	$\boldsymbol{\theta_{MCP_{aa}}}$	$\boldsymbol{\theta_{MCP_{fe}}}$	$\boldsymbol{\theta_{PIP}}$	$\boldsymbol{\theta_{DIP}}$
Min	-60	-55	-27.5	-31
Max	+60	+90	+135	+97.5

4 Graphical User Interface (GUI) Development

Finally, I created a simple but effective graphical user interface environment to test the above end-to-end pipeline of desired fingertip position, inverse kinematics calculations and angles derivation. The relevant GUI files I created are "GUI_forward_kinematics.m" (calculates forward kinematics for given angles), "GUI_inverse_kinematics.m" (solves inverse kinematics for a user-specified end position and orientation) and "GUI_get_all_positions.m" (returns the positions for the three joints plus for end effector for explicit visualization inside the GUI). The returned angles are validated using "check_valid_angles()" and in case the end position is not valid (due to the limited phalanges lengths or due to out-of-range angles), a proper error pop-up message is displayed to the user.

The user interface is intuitive with a simplified 3D representation of the finger (joints and phalanges). The user can set values for the lengths of each phalange (L_1, L_2, L_3) and can specify the desired end state of the fingertip (position x, y, z and angle ϵ) through dedicated text fields and sliders. The complete code for my GUI version is the "Robotic_Finger_GUI_App.mlapp" file. The app design was challenging, as I had no prior experience with GUI development. My GUI work lays the foundation for an extensive GUI version with even more features and functionalities. The Figures 3 and 4 show an example of my GUI implementation. The calculated values for the inverse kinematics for all four angles are displayed in the "Theta Values" field. These values afterwards can be transmitted to the Team B via the MQTT server.

5 Conclusion

In this personal report, I have described in detail my contributions to three aspects of the project, to the mathematical modeling from scratch, to a low-level MATLAB code implementation, and to the creation of a preliminary graphical user interface environment for manipulating the robotic finger. All this leads to the implementation of an effective, comprehensive, and user-intuitive system with a solid theoretical background and technical analysis, while further refinements and additional functionalities can be explored more systematically as future work.

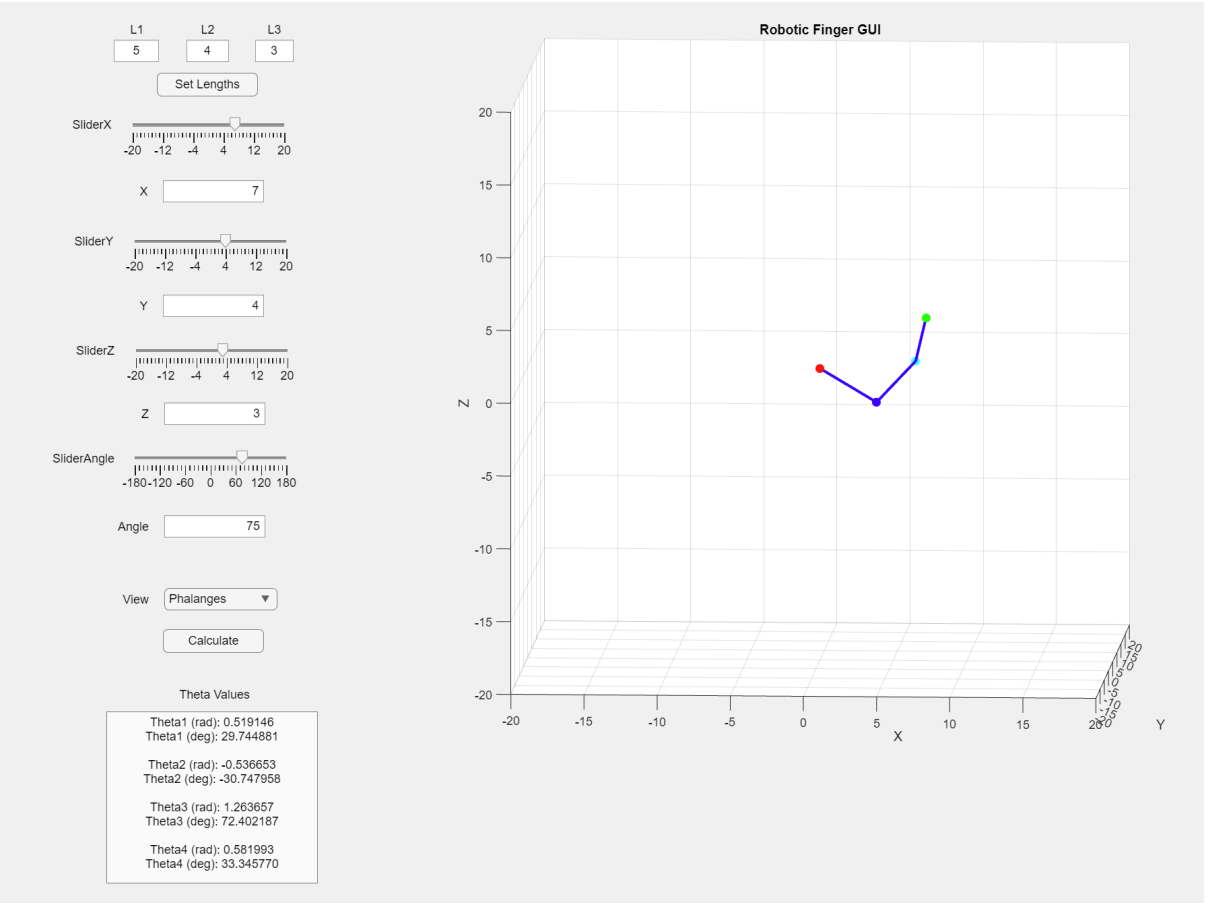


Figure 3: An intuitive user interface environment for controlling the robotic finger. On the left, through the configuration fields and sliders, the user can set the length of each phalange, and manipulate the end position in 3D space (x, y, z) and the end orientation (angle). On the right, the finger's visualization of the user-specified end position is shown.

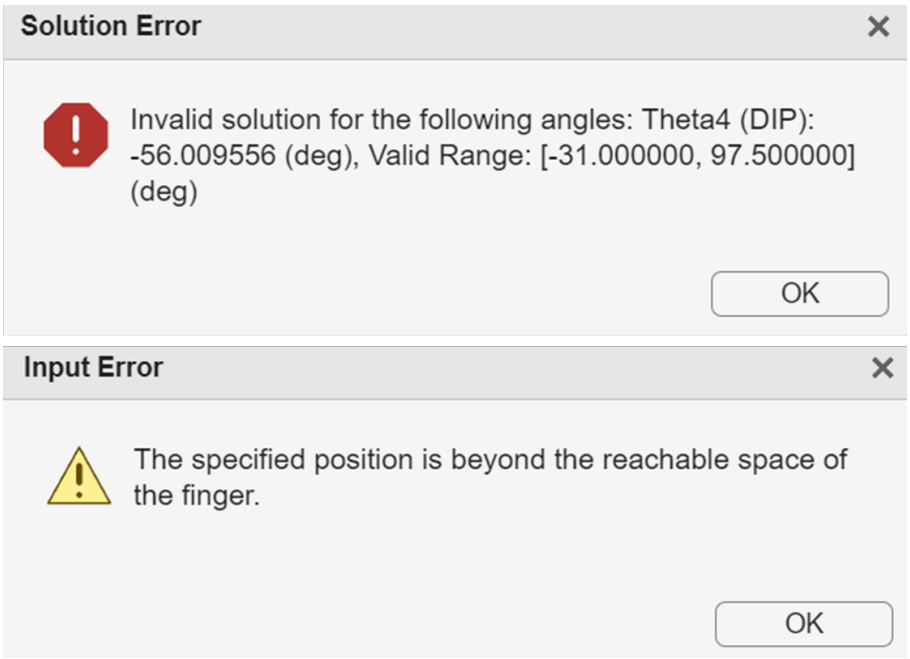


Figure 4: GUI error messages in case the user has specified an invalid end position.

References

[1] G. Borghesan, G. Palli, and C. Melchiorri, "Design of tendon-driven robotic fingers: Modeling and control issues," in 2010 IEEE International Conference on Robotics and Automation, 2010, pp. 793–798.

[2] F. Chen Chen, S. Appendino, A. Battezzato, A. Favetto, M. Mousavi, and F. Pescarmona, "Constraint study for a hand exoskeleton: Human hand kinematics and dynamics," Journal of Robotics, pp. 910 961, 17 pages, 2013. [Online]. Available: https://doi.org/10.1155/2013/910961

[3] Y. Li and L. Wang, "Kinematic model and redundant space analysis of 4-dof redundant robot," Mathematics, vol. 10, no. 4, 2022. [Online]. Available: https://www.mdpi.com/2227-7390/10/4/574

[4] F. Chen, S. Appendino, A. Battezzato, A. Favetto, M. Mousavi, and F. Pescarmona, Human Finger Kinematics and Dynamics. Springer Netherlands, 01 2014, pp. 115–122.

[5] P. Tsakonas, E. Neil, J. Hardwicke, and M. Chappell, "Parameter estimation of a model describing the human fingers," Healthcare Technology Letters, vol. 11, pp. 1–15, 2024.

[6] K. Ueda and S. Katsura, "Development of 4-dof tendon-driven robot finger," in 2024 IEEE 18th International Conference on Advanced Motion Control (AMC), 02 2024, pp. 1–6.

[7] J. Craig, Introduction To Robotics: Mechanics And Control, 3rd Edition. Pearson Education, 2004.

[8] I. Chavdarov and B. Naydenov, "Algorithm for determining the types of inverse kinematics solutions for sequential planar robots and their representation in the configuration space," Algorithms, vol. 15, no. 12, 2022. [Online]. Available: https://www.mdpi.com/1999-4893/15/12/469

[9] A. Nuñez, "Design of a 4-dof robot manipulator with optimized algorithm for inverse kinematics," World Academy of Science, Engineering and Technology International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering, 2015.

[10] S. Cobos, M. Ferre, M. Sanchez Uran, J. Ortego, and C. Pena, "Efficient human hand kinematics for manipulation tasks," in 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2008, pp. 2246–2251.

[11] K.-S. Lee and M.-C. Jung, "Flexion and extension angles of resting fingers and wrist," International Journal of Occupational Safety and Ergonomics, vol. 20, no. 1, pp. 91–101, 2014.

[12] J. Ngeo, T. Tamei, and T. Shibata, "Continuous and simultaneous estimation of finger kinematics using inputs from an emg-to-muscle activation model," Journal of Neuroengineering and Rehabilitation, vol. 11, p. 122, 08 2014.