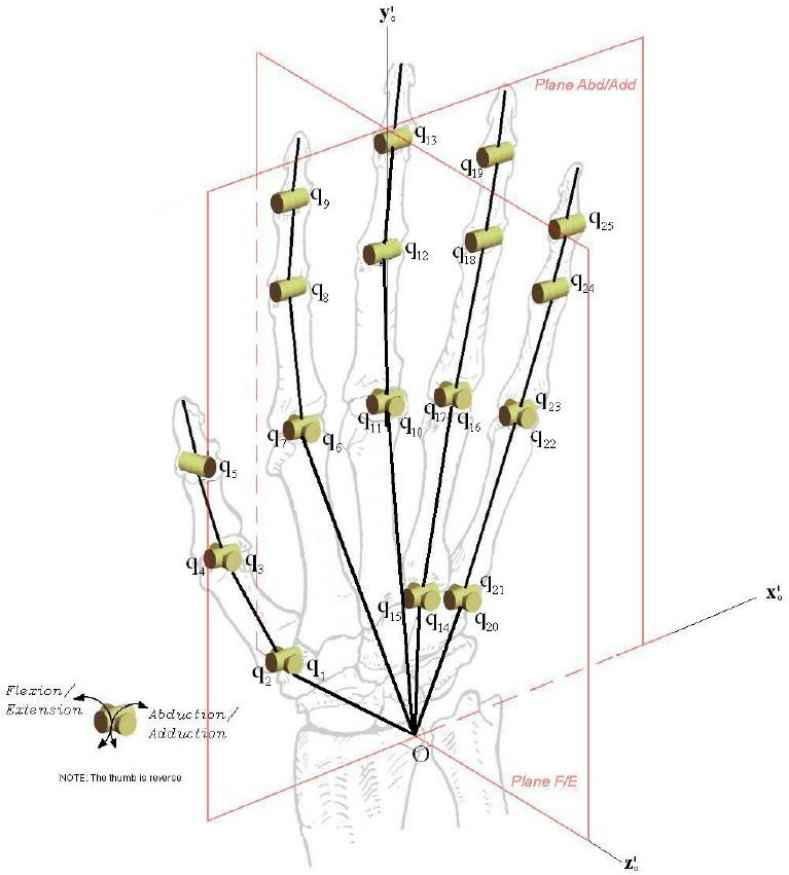

INVERSE KINEMATICS FOR HUMAN FINGERS

Project Report

University College Roosevelt
Middelburg, The Netherlands
Engdata302 - Robotics
May, 2023

Author

Joanikij Chulev
9868046

Instructor

Prof. Dr. Ir. Frank van der Stappen

Contents

Abstract

This project aimed to explore and analyze the capabilities of a 3-link robotic finger and develop an efficient control mechanism for accurately positioning its fingertip. The investigation began by modeling the finger in Mathematica and visualizing its workspace through plots. The unrestricted finger configurations were examined to understand its range of motion. To simplify the problem, the assumption was made that all joint axes were parallel to the z-axis, reducing it to a 2D space. The relationship between the distal inter-phalangeal angle and the proximal inter-phalangeal angle was considered, aligning the finger's movements with real-world constraints. An iterative inverse kinematics solver was then developed to determine the joint angles necessary to position the fingertip precisely at desired coordinates. The solver utilized forward kinematics equations, the inverse Jacobian, and an iterative process to minimize the error between calculated and desired positions. The project was deemed successful, providing a comprehensive understanding of the finger's workspace and capabilities, as well as the solutions for forward and inverse kinematics problems.

1 Introduction

1.1 Purpose of the project

Inverse kinematics is a fundamental concept in robotics that plays a crucial role in solving the complex movements of human fingers. It provides a means to determine the joint angles necessary to achieve a desired fingertip position or orientation in space. By understanding the inverse kinematics of human fingers, we gain insights into the intricate mechanisms underlying finger movement and manipulation.

With a robotic hand that resembles a human hand, the project's goal is to examine the forward and inverse kinematics of a robotic finger. Before going into the specifics of the project, it is crucial to comprehend some finger anatomy. Then we will contrast the finger anatomy with the parameters we need to solve this issue.

The wrist, palm, and five fingers make up a human hand. A finger's capacity to bend at different joints is made possible by the presence of numerous bones, known as phalanges, on each finger. While the thumb has just two phalanges, the index, middle, ring, and little fingers all have three. The finger bone that is closest to the palm and is attached to a metacarpal bone inside the palm is referred to as the proximal phalanx, or PP. The middle bone of the finger is referred to as the intermediate (or medial) phalanx, or IP. A middle phalanx is absent from the thumb, however. The finger bone that is farthest from the palm is known as the distal phalanx, sometimes known as DP.

We can see the bone structure represented in Figure 1.

Figure 1

The full anatomy of a hand can get very complicated so we will only consult parts we need. The phalanges of a finger can rotate in relation to one another or to a metacarpal thanks to three joints with one degree of freedom. The rotational axes of every joint on a single finger are parallel. The proximal phalanx and the metacarpal are connected by the metacarpophalangeal (MCP) joint, the intermediate and proximal phalanges are joined by the proximal interphalangeal (PIP), and the distal and intermediate phalanges are joined by the distal interphalangeal (DIP). We can see the joints represented in Figure 2.

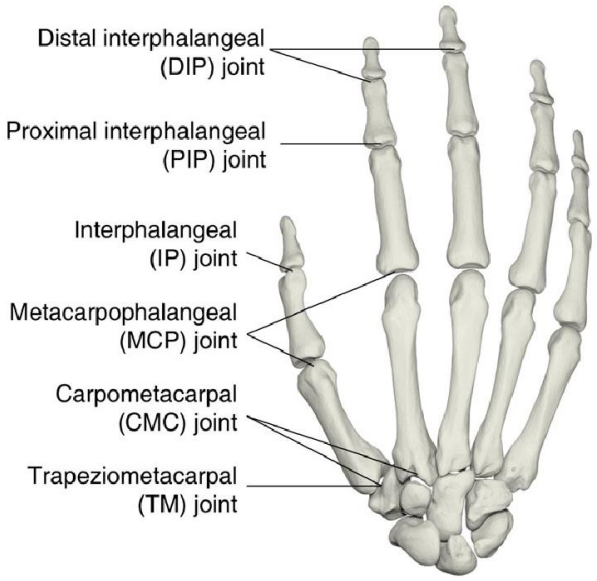

Figure 2

1.2 Defining variables and glossary

The current study does not take the thumb into account. The analysis is made easier by using line segments to represent the phalanges and points to represent the joints. The metacarpophalangeal joint is considered to be fixed at the coordinate frame's origin and all joint axes to be parallel to the z-axis. As a result, the finger's only motions are in the (x, y) - plane.

For the study, the finger has three established joint angles, all of which are measured counterclockwise. We define three joint angles:

θ_m - metacarpophalangeal joint angle (for the MCP joint) - is the counterclockwise angle between the proximal phalanx and the x-axis (metacarpal bone).

θ_p - proximal interphalangeal joint angle (for the PIP joint) - is the counterclockwise angle between the intermediate and the proximal phalanx.

θ_d - distal interphalangeal joint angle (for the DIP joint) - is the counterclockwise angle between the distal and the intermediate phalanx.

These apply to each finger except the thumb, which is comprised of two joints.

Some measurements are needed for the various angles and lengths in order to finish the study. We shall focus just on the index finger in this report to keep things simple. We'll additionally specify three phalanx lengths which are fixed values. The phalanges of a human index finger typically measure the following lengths in millimeters:

L_p - the proximal phalanx with length $L_p = 39.8\, mm$.

L_i - the intermediate phalanx with length $L_i = 22.4\, mm$.

L_d - the distal phalanx with length $L_d = 15.8\, mm$.

Furthermore, the range of feasible angle values for joint angles for a human finger is limited. The ranges of the joint angles are as follows:

For $\theta_m = -\frac{\pi}{3} \leq \theta_m \leq \frac{\pi}{3}$.

For $\theta_p = -\frac{2\pi}{3} \leq \theta_p \leq 0$.

For $\theta_d = -\frac{2\pi}{3} \leq \theta_d \leq 0$.

Additionally, there is an object present denoted by O in the area where the hand is located. The tasks that follow are designed to make contact between the fingertip and an unbounded object (O) in the available space. The object O is specified by:

$$O = \{(x, y, z) \in R^3 | \, y + 18 \leq 0\}$$

1.3 Project solution concept and solvers

Wolfram Mathematica 13.2.0 will be used to complete and set up all of the forward kinematics models and problems. A highly effective solution for the inverse kinematics issue will then be created using Python, based on the equations I derive. First off, it is important to note how challenging it is to model and graphically depict the finger in Python. For this reason, the suggested idea centers around modeling the robotic finger in Mathematica and incorporating sliders for manipulating the joint angles. Also, a fingertip point will be given, represented by a red dot that lights up green when touching object O.

The first phase is gathering all the mathematical equations we need for the solvers. The second phase entails examining different finger configurations to become familiar with the workspace of the finger. Insights regarding the finger's capabilities and range of motion should be gained throughout this experimental period.

Additionally, leveraging the forward kinematics equations derived during the project, an iterative inverse kinematics solver will be developed. This solver, written in Python, will employ a systematic approach to determine the joint angles required to achieve desired x and y coordinates. It will be based on the iterative inverse kinematics solving approach.

During the solving process, the solver will track and present relevant information, such as whether the desired coordinates were reached and the number of iterations required to attain them. Furthermore, an important aspect of this endeavor will be to assess the efficiency of the solver. This evaluation will involve experimenting with different desired points within the finger's workspace and testing diverse initial guesses for joint angles. By doing so, it will be possible to analyze different scenarios and identify any potential areas for improvement.

2 Unconstrained Joint Angles

2.1 Forward Kinematics

Let's first investigate the case in which there are no dependencies or limitations between the three joint angles of the robotic finger, or in other words, when they are unconstrained. The main goal is to develop the forward kinematics equations, which will allow us to calculate the joint angles to identify the location and orientation of the fingertip. We can deduce these equations using elementary trigonometry because of how the setup is constructed. Firstly, we neglect potential intersections of the phalanges.

Secondly, to derive the forward kinematics equations for the fingertip position and orientation as a function of joint angles, we can use a trigonometric approach. We'll think of the finger as a sequence of linked segments, with each segment representing a phalanx. The x and y coordinates will be calculated to determine the fingertip location. We do not consider the fingertip orientation in our case.

First, we'll calculate the position of the PIP joint (J_2) relative to the MCP joint (J_1):

$$J_2(x) = L_p * cos(\theta_m)$$
$$J_2(y) = L_p * sin(\theta_m)$$

Next, we'll calculate the position of the DIP joint (J_3) relative to the PIP joint (J_2):

$$J_3(x) = J_2(x) + L_i * cos(\theta_m + \theta_p)$$
$$J_3(y) = J_2(y) + L_i * sin(\theta_m + \theta_p)$$

Finally, we'll calculate the position of the fingertip (T) relative to the MCP joint (J_1):

$$T(x) = J_3(x) + L_d * cos(\theta_m + \theta_p + \theta_d)$$
$$T(y) = J_3(y) + L_d * sin(\theta_m + \theta_p + \theta_d)$$

Therefore, the position of the fingertip (T) in terms of joint angles ($\theta_m, \theta_p, \theta_d$) is:

$$x = T(x) = L_p * cos(\theta_m) + L_i * cos(\theta_m + \theta_p) + L_d * cos(\theta_m + \theta_p + \theta_d)$$
$$y = T(y) = L_p * sin(\theta_m) + L_i * sin(\theta_m + \theta_p) + L_d * sin(\theta_m + \theta_p + \theta_d)$$

Where $p = \{x, y\}$ represents the fingertip coordinates, where all notations we use for the angles of θ and the distance of L were defined previously.

The derived forward kinematics equations will provide a clear and concise representation of how the joint angles influence the fingertip's location. These equations will serve as a valuable tool for understanding the finger's behavior and predicting its movements based on the input joint angles.

2.2 Mathematica model

An early step was done to visualize the robotic finger's capabilities without taking into account the existence of any barriers, such as the obstruction O, in order to obtain insight into the workspace of the robotic finger. The matching workspace coordinates were identified and shown by taking into account all conceivable combinations of the three joint angles. To show the locations that the fingertip may reach, a graphing mechanism was created in Mathematica as seen in Figure 3.

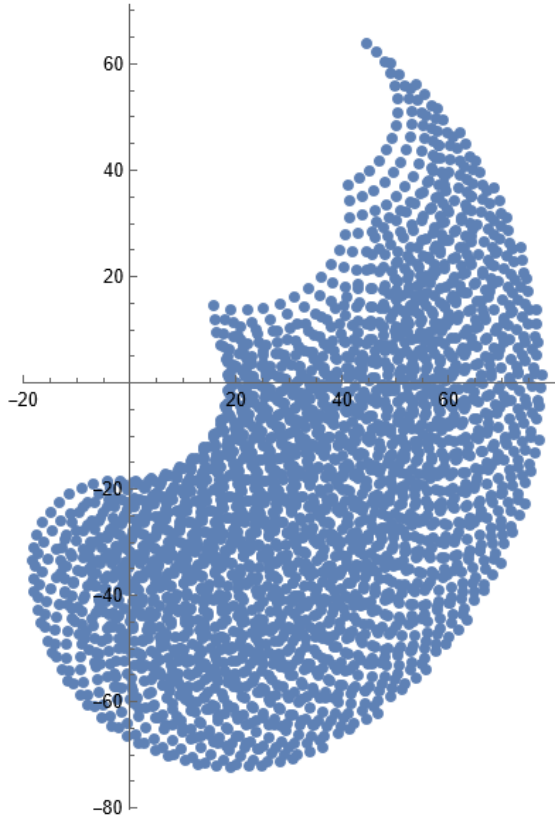

Figure 3

Further, it is also beneficial to represent all the values for the angles and fingertip placements if the fingertip does not interact or penetrate with obstacle O, as seen in Figure 4.

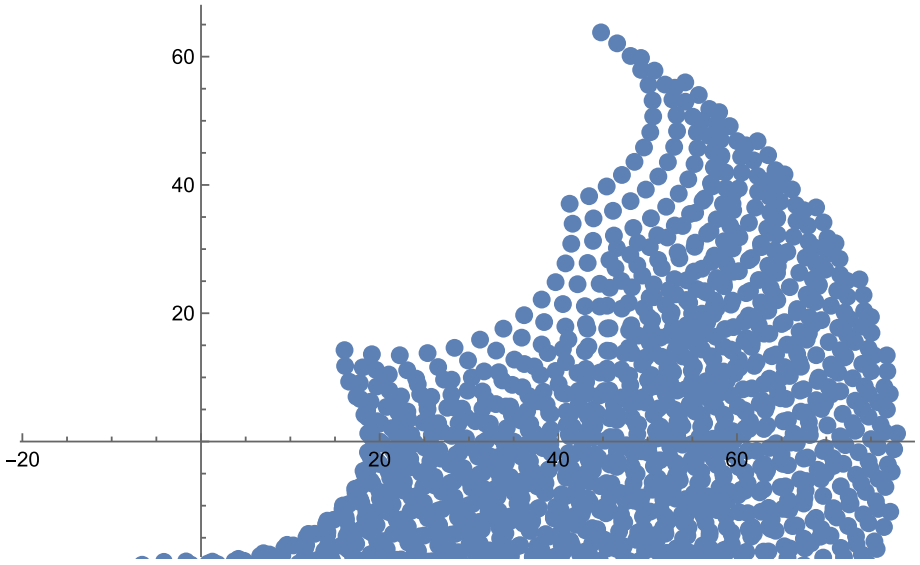

Figure 4

Next, we will examine how the Mathematica model looks and behaves. Hence, we can see a sample configuration for the finger seen in Figure 5. Note that the fingertip point is red because it still has not intersected O.

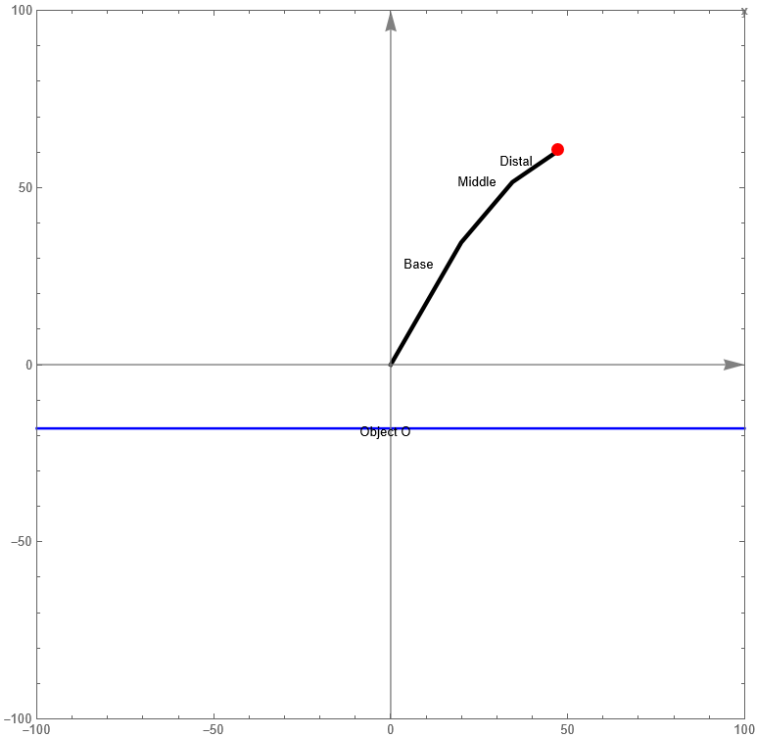

Figure 5

We can see the slider menu and the values for the sliders chosen in Figure 5. This is shown in Figure 6.

Figure 6

Another investigation was if there are ways to place certain finger segments vertically or horizontally and still make the fingertip intersect or interact with O.
Solutions, of which many were found, was to place the proximal phalanx horizontally, to place the intermediate phalanx vertically and to place the distal phalanx vertically. Indeed, using our Mathematica model it was quite simple to get these results and multiple solutions. Figure 7a represents one of the outcomes when the proximal phalanx is placed horizontally. Figure 7b represents one of the outcomes when the intermediate phalanx is placed vertically. Figure 7c represents one of the outcomes when the distal phalanx is placed vertically. Note that the fingertip point is green in all cases because it has indeed intersected O.

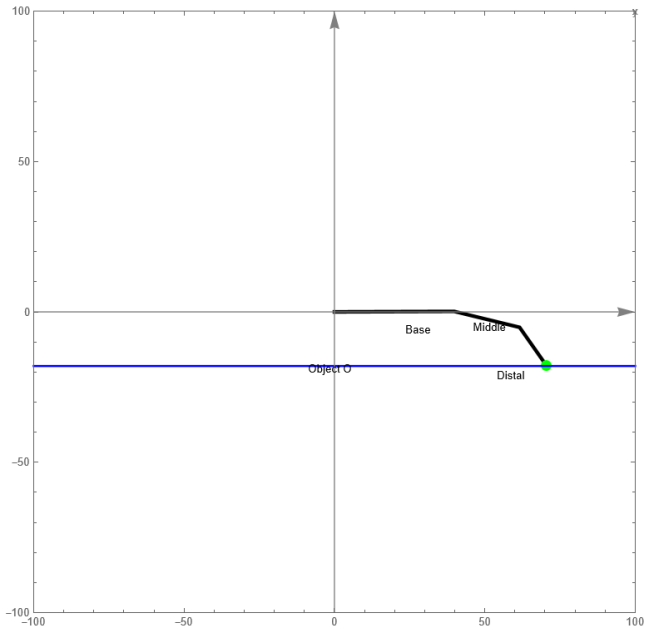

Figure 7a

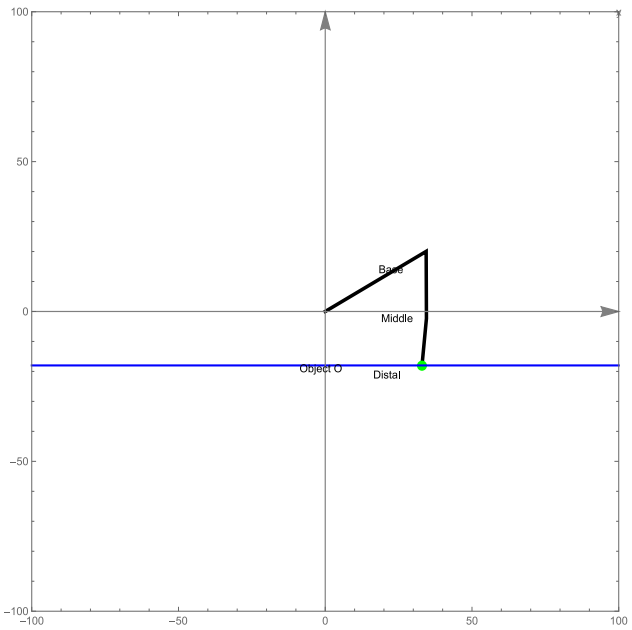

Figure 7b

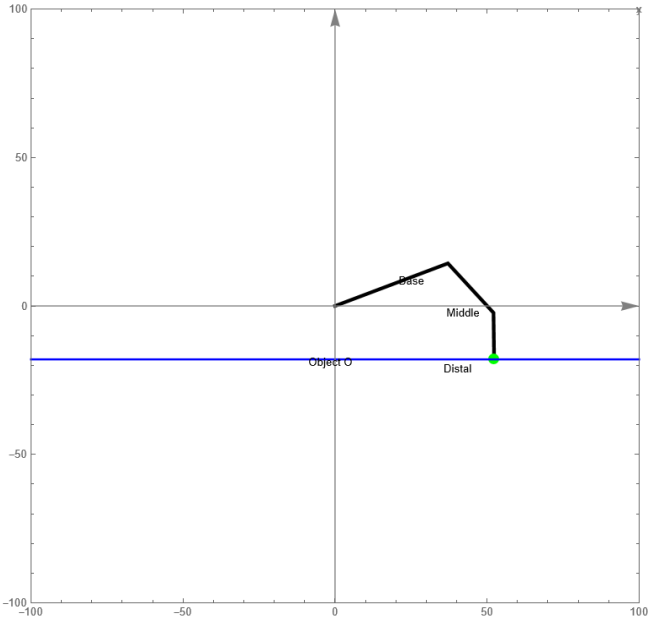

Figure 7c

3 Constrained Joint Angles

3.1 Inverse Kinematics

The issue must be made simpler to make use of the information acquired and place the fingertip of the robotic hand on obstacle O. The initial simplification makes the assumption that all of the finger's joint axes are parallel to the z-axis. With this reduction, the issue is essentially reduced from 3D space to 2D space, making analysis and computation simpler. By assuming parallelism between the joint axes and the z-axis, the finger's movement is restricted to a plane. This simplification enables us to focus solely on the x and y coordinates, disregarding the z-coordinate.

Additionally, it is important to note there is a relationship between two specific joint angles: the distal inter-phalangeal angle θ_d and the proximal inter-phalangeal angle θ_p. In reality, these two angles exhibit a particular relationship that needs to be taken into account when determining the finger's configuration. Given by:

$$\theta_d = \frac{2\theta_p}{3}$$

With this new relation we have, we can substitute and get a simpler equation. As follows:

$$x = T(x) = L_p * cos(\theta_m) + L_i * cos(\theta_m + \theta_p) + L_d * cos(\theta_m + \theta_p + \frac{2\theta_p}{3})$$

$$y = T(y) = L_p * sin(\theta_m) + L_i * sin(\theta_m + \theta_p) + L_d * sin(\theta_m + \theta_p + \frac{2\theta_p}{3})$$

Thus,

$$x = L_p * cos(\theta_m) + L_i * cos(\theta_m + \theta_p) + L_d * cos(\theta_m + \frac{5\theta_p}{3})$$

$$y = L_p * sin(\theta_m) + L_i * sin(\theta_m + \theta_p) + L_d * sin(\theta_m + \frac{5\theta_p}{3})$$

Given the desired x and y coordinates for the fingertip and the known lengths of the finger's phalanges, we now have a system of two equations with two unknowns. To solve for these angles, one effective approach is to utilize the equations employed in the iterative inverse kinematics solution. The equations are given by:

$$q^{(i+1)} = q^{(i)} + J^{-1}\left(q^{(i)}\right)\ \delta T\left(q^{(i)}\right)$$

The technique involves using the above equation to iteratively decrease the error until it is below a predetermined threshold, starting with an initial estimation of the angles. Where $q^{(i)}$ refers to a vector of initial approximations for the angles, a guess. $J^{-1}(q^{(i)})$ is the inverse of the Jacobian of the coordinates, which is evaluated at $q^{(i)}$. $\delta T(q^{(i)})$ is defined as the error in the initial approximation which is the difference between the desired coordinates and the current coordinates. $q^{(i+1)}$ is the next guess for the joint angles, with the purpose to reduce error. This was all implemented in the Python code, in which you only specify the desired coordinates and an initial guess, and you get the output. Additionally, since the Jacobian involves derivatives, they were calculated prior on paper and implemented in the code as a finished result.

3.2 Python solver

As mentioned before the code was written using the logic of the $q^{(i+1)}$ equation established before. The program is set up to run iterations until it reaches the desired result with a threshold value for the error (program is stopped once this error is passed, and result is printed). The threshold value chosen was error > 0.0000001. This value proved to be efficient and a very close estimation to the desired coordinates. Notice that the desired value for the y coordinate will always be -18 to establish an intersection with O. And for the desired value for coordinate x, we will pick arbitrary values that would still make sense for the intersection to happen. We can choose our x values by going back to the Mathematica model and determining where the fingertip intersects O for values of x.

We will investigate a few tests and see what output results we get.

Firstly, let us see what difference the initial guess makes. This will be done by setting the desired coordinates to $\{x, y\} = \{50, -18\}$. Then we will set the initial guess to $\{0, -1\}$ (which would make sense) and contrast those results to setting the initial guess to $\{-25, -13\}$, which is very off. Even though this is the case, the solver is still efficient in solving the problem. With 5 iterations in the 1st case and 8 in the 2nd, 3 iterations being a very small difference. This can be seen in figure 8a and 8b.

```
Joint angles for the metacarpophalangeal (MCP) and proximal
interphalangeal (PIP) joints are: [ 0.37641848 -1.23268268]
Number of iterations it took: 5
Errors for the estimates for x and y are: [ 8.59188276e-11
-5.31649391e-10]
Final coordinates x,y of the fingertip are:
(49.99999999999999, -18.0)
```

Figure 8a

```
Joint angles for the metacarpophalangeal (MCP) and proximal
interphalangeal (PIP) joints are: [ 0.37641848 -1.23268268]
Number of iterations it took: 8
Errors for the estimates for x and y are: [3.37507799e-12
9.87654403e-13]
Final coordinates x,y of the fingertip are:
(50.00000000000001, -18.0)
```

Figure 8b

Secondly, let us see what difference the initial guess makes. This will be done by setting the initial guess to $\{0, -1\}$. Then we would set the desired coordinates to $\{x, y\} = \{70, -18\}$ and contrast those results to setting the desired coordinates to $\{x, y\} = \{20, -18\}$. Both coordinates are a valid value for the fingertip intersection with O, thus both are solved with the same logic with the 1st case being faster for 1 iteration. This can be seen in figure 9a and 9b.

```
Joint angles for the metacarpophalangeal (MCP) and proximal
interphalangeal (PIP) joints are: [ 0.10087461 -0.57052766]
Number of iterations it took: 6
Errors for the estimates for x and y are: [ 3.08375547e-12
-1.32516220e-12]
Final coordinates x,y of the fingertip are:
(69.99999999999999, -18.0)
```

Figure 9a

```
Joint angles for the metacarpophalangeal (MCP) and proximal
interphalangeal (PIP) joints are: [ 0.16056433 -1.89507766]
Number of iterations it took: 7
Errors for the estimates for x and y are: [-3.55271368e-15
-3.55271368e-15]
Final coordinates x,y of the fingertip are:
(19.999999999999996, -17.999999999999996)
```

Figure 9b

4 Conclusions

A thorough grasp of the robotic finger and its workspace has been provided, making the project a well-done study. In order to see and evaluate the finger's capabilities, Mathematica was used to plot the finger's locations. The plots produced by Mathematica have not only been effective but have also been very useful in understanding the dexterity and range of motion of the finger.

The development of an inverse kinematics solver has also been a resounding success. The solver's ability to accurately determine the joint angles required to achieve desired coordinates showcases its effectiveness. The solver's performance has been exceptional, consistently producing satisfactory results and successfully positioning the fingertip as intended. Overall, the project's success stems from a systematic and comprehensive approach, utilizing mathematical modeling, computational tools, and problem-solving techniques.

References

Anaconda. (2023, May 15). *Anaconda | The World's Most Popular Data Science Platform*. https://www.anaconda.com/

Elnady, A. O. (n.d.). *Iterative Technique for Solving the Inverse Kinematics Problem of Serial Manipulator*. Copyright? 2012 Scientific & Academic Publishing. All Rights Reserved. http://article.sapub.org/10.5923.j.jmea.20211001.02.html

Inverse kinematics for the index finger and thumb. (n.d.). ResearchGate. https://www.researchgate.net/figure/Inverse-kinematics-for-the-index-finger-and-thumb_fig3_257143429

Wolfram Mathematica: Modern technical computing.

What Is Inverse Kinematics? (n.d.). MATLAB & Simulink. https://www.mathworks.com/discovery/inverse-kinematics.html#:~:text=Inverse%20kinematics%20is%20the%20use,between%20bins%20and%20manufacturing%20machines.