MECH0064 MSc Group Design Project

A Compact Continuum Robotics Manipulator for Applications in Therapeutic Ultrasound

Group members:

Zehao Ye (23119333) Yuhao Zhu (23041703)

Zehao Ye (23119333) Zehao Ye (23119333)

Zehao Ye (23119333) Zehao Ye (23119333)

Supervised by **Dr Reza Haqshenas**

Abstract

This is the Abstract of the final report.

AAAAA bbb

test for github

Key Words: Continuum Robotic e.g.

Contents

List of Figures

List of Tables

1 Introduction

1.1 Background

This is the background part.

1.2 Motivation

This is the motivation part.

1.3 Introduction

Continuum robots has emerged and attracted a lot of attention since 2008 [1]. Before that, rigid joint robots were dominating the robotic arm industry. Compared with traditional rigid joint manipulator, continuum robots stand out for their flexible, highly bendable structure and extremely flexible motion performance. The limitations of rigid joint robots have gradually shown up in applications requiring highly detailed operation and in complex or space-limited environments. Thus, the continuum robot was developed and perfected during the years. This new kind of robot not only changes the code of traditional robot design but also demonstrates unprecedented application potential in fields such as exploration industry and medical science [2]. Meanwhile, rigid-flexible-soft coupled continuum robots combine the multiple advantages of the stability of rigid structures, the flexibility of bendable structures, and the compliance of soft structures, and are one of the most promising robots for increasingly complex tasks [3].

With unique bionic structure and motion characteristics, continuum robots provide new possibilities to solve these challenges.

This paper will discuss different types of existing continuum robots and their working principles, advantages, and disadvantages, then propose a proper continuum robot design that can be mainly applied to medical applications.

2 Literature Review

In the last two decades, significant progress in electronic and computer technologies has led to remarkable growth in the field of manipulator robotics. Manipulators developed by various institutions have been integrated into the industrial sector to autonomously or semi-autonomously perform repetitive tasks. Simultaneously, manipulators are utilized for tasks with stringent precision requirements to minimize errors. Additionally, essential tasks are undertook by manipulators to substitute for humans in challenging environments.

2.1 Manipulators in Biomedical Applications Zehao

With a growing emphasis on the field of biology, manipulators have also been introduced to provide assistance. In the field of medicine, manipulators have been utilized since the end of the last century. The AESOP robotic surgical system, proposed by Computer Motion founded by Yulun Wang in 1993, was investigated in laparoscopic surgery in 1997 [4]. Afterward, the ZEUS robotic surgical system endowed with a trilateral manipulator configuration was proposed by Computer Motion in 1998 [5].During the period from 1999 to 2001, the ZEUS system was utilized for a series of clinical surgeries, demonstrating excellent performance [6–8]. At the beginning of the 21st century, a novel robotic system, da Vinci robotic surgery system, was designed to facilitate more intricate surgical procedures [9]. Moreover, manipulators can be leveraged in the field of biological physics as a viable approach for biological experiments. In the HIFU system developed by An et al., the SCARA (self compliant automatic robot assembly) robot was employed as manipulator, incorporating an ultrasound probe for the purpose of scanning biological tissues [10]. The robotic system FUSBOTs (Focal Ultrasound Surgery RoBOTs) was proposed and upgraded to accomplish more precise targeted treatment with multiple DoF (Degrees of Freedom) manipulators [11–13]. Despite the various utilization of manipulator platforms designed for accommodating ultrasonic transducers [14–16], certain limitations persist. Hence, a comparison of different manipulators is necessary in selection of appropriate type manipulator for integrating ultrasonic transducer.

2.2 Types of Manipulator Zehao

From the perspective of geometry, rigid-body manipulators can be categorized into two main types: parallel mechanisms and serial link [17]. The control of parallel mechanisms is relatively complex. Meanwhile, serial link manipulators can be further divided into five types: Cartesian (PPP), articulated (RRR), cylindrical (RPP), spherical (RRP), and SCARA (RRP). Additionally, with the advancement of robotics technology, two other types of manipulators, namely biomimetic and anthropomorphic, have demonstrated their advantages [18, 19]. The comparative analysis will be conducted to highlight the distinctive features of different manipulators presented in Table 1, leading to the identification of the most suitable type for specific applications.

Table 1: The Characteristics of Different Manipulators.

Manipulators	Types	DoF	Features	Applications
Stewart Platform [20]	Series-Parallel	6	series-parallel duality, mature kinematics algorithm.	flight simulation [21]. robocrane [22]
Cartesian	PPP	3	technological maturity, low complexity, cost-effective, high payload.	3D-Printing framework [23] warehousing & hoisting [24] agricultural machinery [25]
Articulated [26]	RRR	6	factory automation, programmable, high precision and payload.	milling operations [27] material handling [28]
Cylindrical [29]	RPP	3	good precision, limited workspace.	assembling industries [30]
Spherical [31]	RRP	3	collision avoidance, low precision, long reach, light weight.	low-precision semi-automated tasks [27, 28]
SCARA [29]	RRP	3	high accuracy and fast operation.	surgical applications [32] medical rehabilitation [33]
Continuum/ Soft [34, 35]	Biomimetic	∞	ability to handle fatigue objects, environmental adaptability.	tendon driven [36–38] fibre-reinforced [39, 40] fluid-elastic drive [41–43] bionics materials [44–46]

Continued on next page

Table 1 – continued from previous page

Manipulators	Types	DoF	Features	Applications
Humanoid	anthropo-morphic	7	high efficiency, flexibility.	humanoid robots driven by pneumatic artificial muscles [47]

The distinctive structure of continuum manipulator imparts to the system with theoretically infinite DoF. The characteristic enables the continuum manipulator to navigate through confined spaces or manipulate objects along specific trajectories within the workspace [2, 34, 35]. Simultaneously, the continuum maniuplator demonstrates suboptimal performance in the context of payload, making it challenging to meet the payload capacities commonly observed in industrial robotic systems. Nevertheless, within the realm of biology applications, the emphasis shifts towards precision and operability for manipulator. In this report, the manipulator platform integrated with the ultrasonic transducer module necessitates a design. The selection of the continuum manipulator was predicated upon its outstanding flexibility and the lightweight attributes of ultrasonic transducer module. The forthcoming discussion will delve into the categorisation of continuum robots, ultimately selecting an appropriate type to serve as the centerpiece for the manipulator platform.

2.3 Types of Continuum Robots Yuantong

Nowadays, a variety of continuum robots exist, each exhibiting unique structures and functions. They serve in different fields such as medicine, construction, and exploration. In this section, some of the most popular continuum robots will be introduced, providing basic insights into their structures as well as discussing their merits and drawbacks.

2.3.1 Tendon-Driven Robots

The arm of the Tendon-Driven robot consists of a backbone, several tendons and disks. The backbone defines the structure and posture of the entire robot arm, while disks

define the diameter and divide the robot arm into segments, and tendons are stretched to create deformation and movements of different directions for the robot arm.

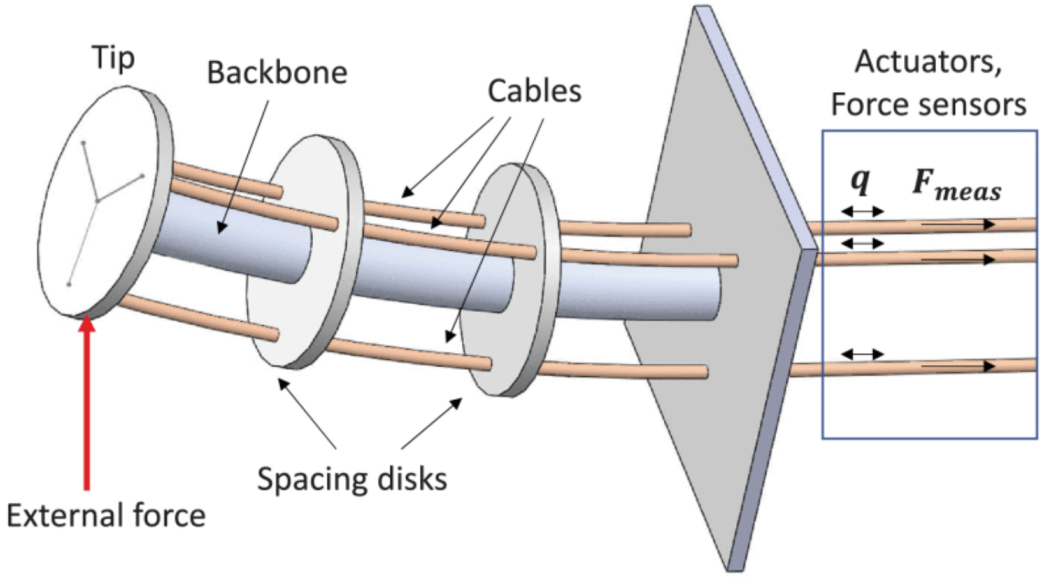


Figure 1: A three-tendons continuum robot with one segment [48].

Figure 1 shows only the simplest tenon-driven robots. In practical applications, there may be more than one backbone, and the disks are not necessarily parallel.
Compared with other continuum robots, one of the most significant advantages of tendon-driven robots is their flexibility. This advantage makes it more effective in performing tasks in complex and restrictive Spaces. In addition, due to the simple components needed to construct the robot, it is easier to meet lightweight design specifications. Moreover, like the concentric-tube continuum robots, tendon-driven continuum robots can be built designed on a small scale with diameters of below 10mm [49].
However, due to its simple actuating principle, more complex algorithms are needed to control it more accurately. Also, the tendon-driven robots actuate by pulling the tendon, which makes the friction between the tendon and other components inevitable, which will accelerate the wearing speed of the tendon-driven robots.

2.3.2 Fishbone Robots

Fishbone robots, as the name suggests, are inspired by fish bones. It is consists of several "fishbone moduless", which are composed of a number of rigid cross-shaped plates, and a flexible elastic plate backbone embedded in the middle, forming a rigid-soft coupling

structure[3]. When the different modules are connected, the backbones of the bionic fishbone modules are perpendicular to each other, and finally form a complete main frame of the fishbone robot. Like the tendon-driven robot, the fishbone robot is also controlled by cable, but it is worth noting that each module is controlled by two separate cables. This means that for a fishbone robot composed of n fishbone modules, there are a total of 2n cables controlling the motion of the whole robot. In addition, the planes formed by the two ropes that control each module are perpendicular to each other, so that each section rotates along a plane perpendicular to each other.

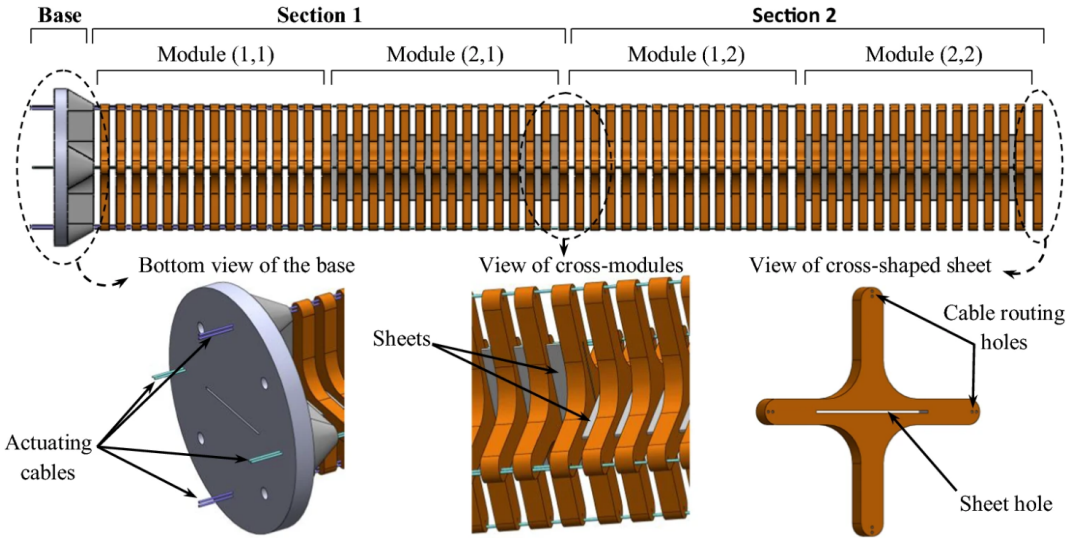


Figure 2: The cable-driven fishbone continuum robot with cable arrangement [50].

From the graph above, it is shown that because of the numbers of rigid cross-shaped sheets formed a large diamater frame, the structural stability of this kind of robot is very strong. Also, each fishbone unit can provide one DoF, making it easy to reach a very high total DoF. However, due to the multiple-curves deformation trajectory, the fishbone robots are not suitable for situations where strict trajectory is required[3].

2.3.3 Concentric Tube Continuum Robots

concentric tube robots, shaped like retractable walking sticks, consist of many tubes with decreasing diameters. Each tube is nested on top of the previous wider tube.

The concentric robots are made of two parts: tubes and coaxial actuation units. The tubes are the main structural element of this robot and act as the backbone. The coaxial

actuation unit consists of two motors which are responsible for rotation and translation movement respectively. Each tube is actuated by an independent coaxial actuation unit.

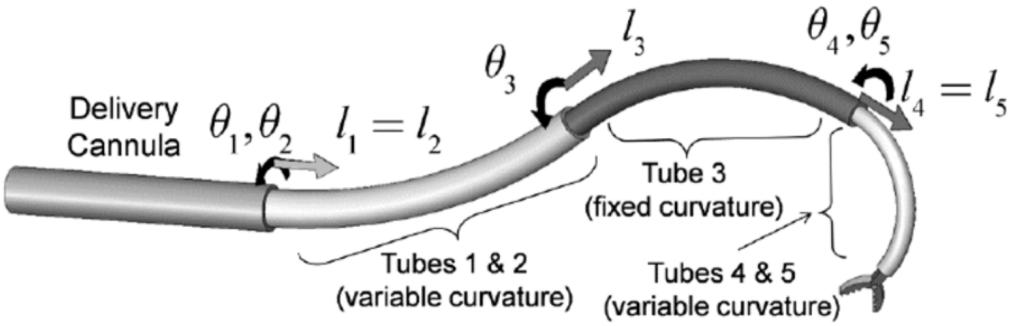


Figure 3: An example concentric tube continuum robot [51].

The most significant advantage of this kind of robots is that of all the continuum robots, concentric tube robots have the smallest possible outer diameter and are best suited to work in confined and narrow Spaces. Therefore, it is the ideal choice for surgical operations.

Their disadvantages, on the other hand, are also very evident. Since each tube requires an independent actuation unit, the overall length of the robot cannot be very long, because longer lengths will lead to more tubes, and will lead to more tip position errors.

3 Design

3.1 Structural Design

The novel Continuous Six-degree-of-freedom Manipulator introduced in this paper, which corresponds to the scheme design shown in Figure 2, belongs to the class of bio-inspired continuum robots [3]. In the design process, a nylon cable drive mechanism is employed as a substitute for the complex musculature of fish, enabling it to maintain stability while undergoing deformation movements, thus achieving effective deformation motion and controllability. As depicted in Figure 4, the rigid cross-shaped sheets is utilized to mimic the spine and ribs, uniformly arranging these sheets on a rubber pure soft sleeve to replace the spinal joints.

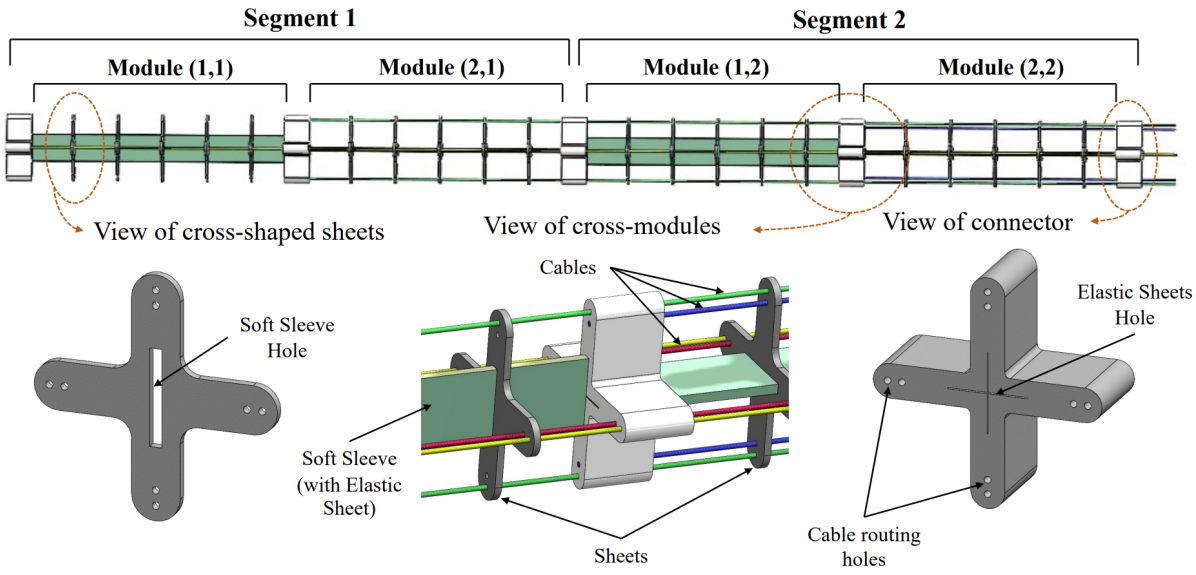


Figure 4: The main components of the proposed manipulator and cables arrangement.

3.2 Designation and Parameter Definition of Manipulator

The manipulator is composed of four identical modules, each of which can be independently controlled to bend within a two-dimensional space. Each module consists of an elastic sheet and five uniformly arranged cross-shaped sheets, where the rubber pure soft sleeve outside the elastic sheet is interference-fitted with the cross-shaped sheets. The connector end of each module features two symmetrically arranged inextensible cables that pass through the cross-shaped sheets and are anchored to the

connector, driving the end of each manipulator module. By applying tension to one of these cables, the elastic sheet can be deflected, thereby achieving a single degree of freedom planar motion at the end of the manipulator module. Since the elastic sheets of the modules are transversely mounted, the deflection of two or more sheets simultaneously allows for spatial motion at the end of the manipulator, as shown in Figure 4. Overall, the proposed manipulator comprises two segments, each maintaining a constant length of 675 mm and a weight of 88 grams. Each module contains five cross-shaped sheets with a circumferential radius of 20 mm and a thickness of 1.5 mm. Both the head and the tail of each segment are connected by connectors, which have the same circumferential radius as the cross-shaped sheet and a thickness of 15 mm.
The definition of manipulator parameters is presented in Table 2. The subsequent sections will adhere to the corresponding naming conventions.

Table 2: The Parameters of Manipulators.

Paramter	**Definition**	**Value (mm)**
Sr_i	the length of elastic sheet in Module i	$Sr_{1,2,3,4} = 150$
d_i	the thickness of the cross-shaped connector	$d_{1,2,3,4,5} = 15$
N	the number of cross-shaped sheet in a module	$N = 0$ ~ 15
r_i	the distance between the centroid of	$r_{1,2} = 17.5$
	connector and cable routing hole	$r_{3,4} = 15$
ΔS_i	the change volume of cable, cable$_{2i-1}$ and	
	cable $_{2i}$ corresponds to Module i	
$\boldsymbol{\Delta S}$	a series of change volume of cables	$\Delta S_1 \sim \Delta S_8$
α_i	the bending angle of module i	
$\boldsymbol{\alpha}$	a series of bending angles about four modules	$[\alpha_1,\ \alpha_2,\ \alpha_3,\ \alpha_4]$
$\boldsymbol{\theta}$	a series of inverse kinematics solutions	$[\theta_1,\ \theta_2,\ \theta_3,\ \theta_4]$
$\boldsymbol{\epsilon}$	threshold of the error in FABRIKc algorithm	$\boldsymbol{\epsilon} = 0.02$

3.3 Material Selection

The fabrication of the Continuous Six-Degree-of-Freedom Manipulator primarily employs the following methods: Shape Deposition Manufacturing (SDM)[52],

Casting[53, 54], and Multi-material 3D Printing Technologies[55, 56]. By integrating SDM with Casting techniques, it is possible to utilize rigid, flexible, and soft materials, and integrate components such as sensors and circuits into the structure, which is particularly crucial for the integrated manufacturing of sensors and actuators. On the other hand, Multi-material 3D Printing technology supports the unified manufacturing of soft and hard materials, enabling the creation of structures with more complex geometries and adjustable material hardness according to specific requirements. This approach results in structures that are not only more scientifically arranged and exhibit a more rational combination of rigidity and flexibility but also possess higher stability and superior performance. Moreover, Multi-material 3D Printing enhances manufacturing efficiency. Given these advantages, 3D Printing technology emerges as the preferred method for constructing continuum robots.

The materials for the cross-shaped sheets and connectors can be fabricated in an integrated manner using Multi-material 3D Printers, which avoids the impact of imperfect assembly on deformation and achieves a compact design. The material chosen simulates engineering plastics (Mainly digital material with a Poisson's ratio of 0.394, a modulus of elasticity of 2.2 GPa and a Young's modulus of 2.5 GPa). The soft sleeve material is made of silicone rubber, characterized by a Poisson's ratio of 0.47. Furthermore, a 65Mn alloy steel (with a Poisson's ratio of 0.244 and a Young's modulus of 0.2 GPa) elastic sheet (flexible) embedded within the soft sleeve forms a bio-inspired sheath, creating a bio-inspired fishbone structural unit. This structural unit is a hybrid of rigidity, flexibility, and softness, not only exhibiting excellent constant curvature characteristics but also higher structural stability. The bio-inspired fishbone unit also provides a constraint mechanism for the synchronous bending of the driving cables, effectively reducing the impact of external mechanical collisions on the cables [3]. Therefore, this unique structural design is a key foundation for achieving high-precision modelling of continuum robots.

In this design, the elastic sheet is combined with the connector by embedding it into a 0.3 mm wide slot on the connector and utilizing an interference fit, a method that eliminates the need for additional fasteners and ensures the structural integrity of the continuum robot during standard operations. Nonetheless, to prevent the robot from separating under extreme overload conditions, special fixing holes were designed on

both the elastic sheet and the connector. The installation of screws through these holes enhances the robustness of the continuum robot in harsh environments. Leveraging the modular design characteristic of the continuum robot, it can be constructed by serially connecting multiple similar motion modules. The example demonstrated in this study is comprised of two such motion modules, as shown in Figure 4, whereby altering the length of the driving cables enables the continuum robot to achieve multi-degree of freedom movements.

Figure 4 illustrates the cable configuration scheme proposed for the continuum robot in this paper. Each bio-inspired fishbone structural unit is actuated by two symmetrically arranged cables, which traverse through the guide holes of the connectors and cross-shaped sheets, with each module being controlled by two motors, totaling eight motors. The guide holes in the cross-shaped sheets ensure that the cables maintain an arc shape within their bending region. In the actual design process, the maximum number of cross-shaped sheets is determined by the maximum bending angle preset for the bio-inspired fishbone unit. Through simulation analysis, we have concluded that the bending performance of the elastic sheet is optimal when the bending angle reaches 73°. The design proposal offers the following advantages:

- Lightweight: Utilizing cross-shaped sheets results in a lighter weight compared to traditional cylindrical manipulators.
- Stable Deformation Motion: The deformation motion is stable, offering an improvement over traditional cylindrical manipulator.
- Rigidity: The adoption of a stacked skeleton structure effectively eliminates bending deformation.
- Energy Consumption [57]: The proposed robot design facilitates a reduction in the energy required to achieve spatial movement; that is, each module employs two actuation motors instead of three, as is the case with continuum robots possessing a cylindrical backbone.

Like all continuum robots, the sole drawback of this design is the complexity involved in modeling.

3.4 Methodology

3.4.1 Strain Analysis

The objective of the analysis is to determine the mechanical boundaries and failure risks of the manipulator to improve its design for better durability and performance. Using ANSYS, the manipulator was evaluated at its maximum operational angle to understand its behavior under extreme conditions. This revealed specific strength levels of the model, aiding in verifying the arm's structural soundness for its intended use.

As the four modules of the model are identical, it was reasonable to simplify the model to a single module to reduce computational resources and analysis time while maintaining a high level of accuracy in the results. The simplified model is shown in Figure 5.

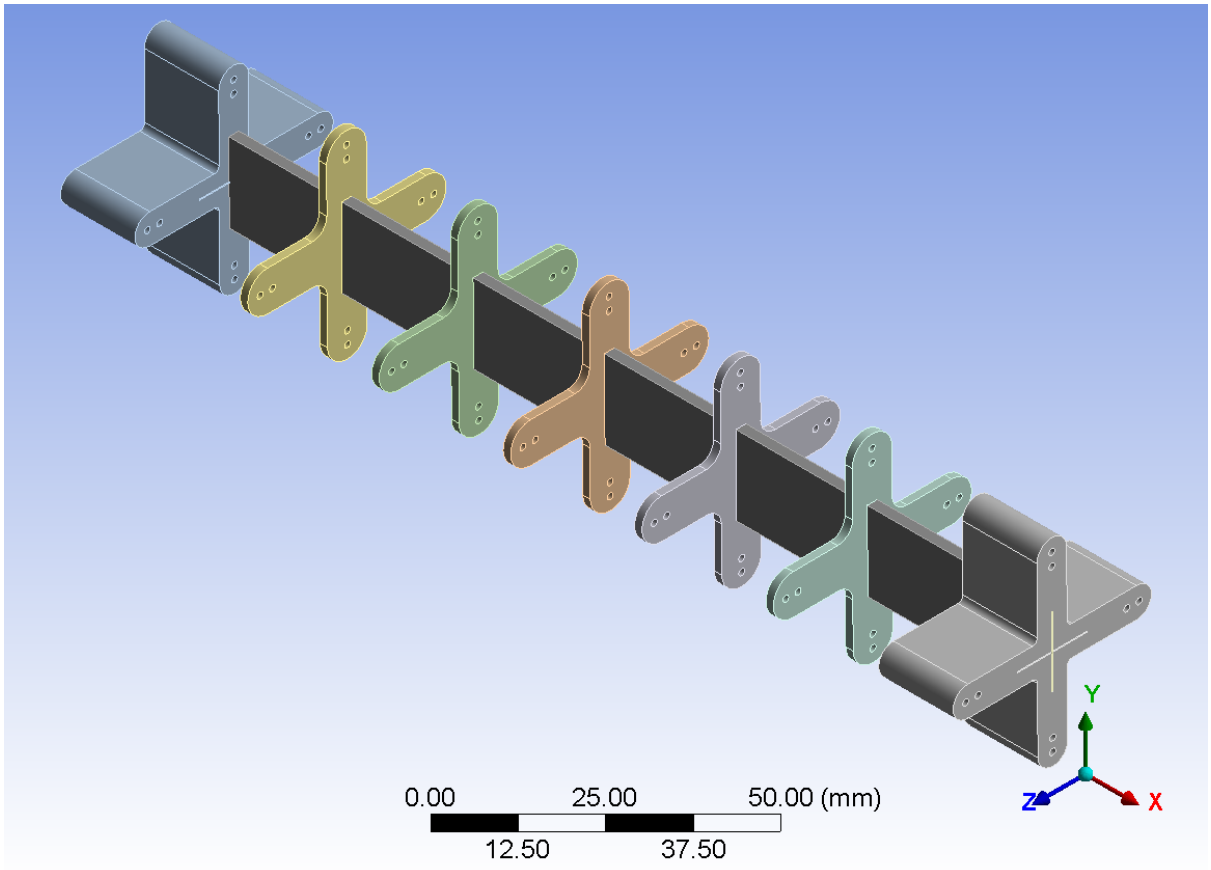


Figure 5: The simplified model of manipulator.

Load it into ANSYS and use the Static Analysis Module to evaluate the stresses and strains at a maximum bending angle of 73 degrees.

To ensure that the results of the simulation are independent of the mesh size, a mesh-independent study is required. Finite element analyses were performed with mesh sizes

ranging from 8.5 mm down to 2.5 mm. As illustrated in Figure 6 a significant peak in stress at a mesh size of 6.5 mm suggests possible numerical artifacts or unresolved stress concentrations. Subsequent refinement below this threshold showed the stress values approaching a constant level, indicating the potential achievement of mesh independence.

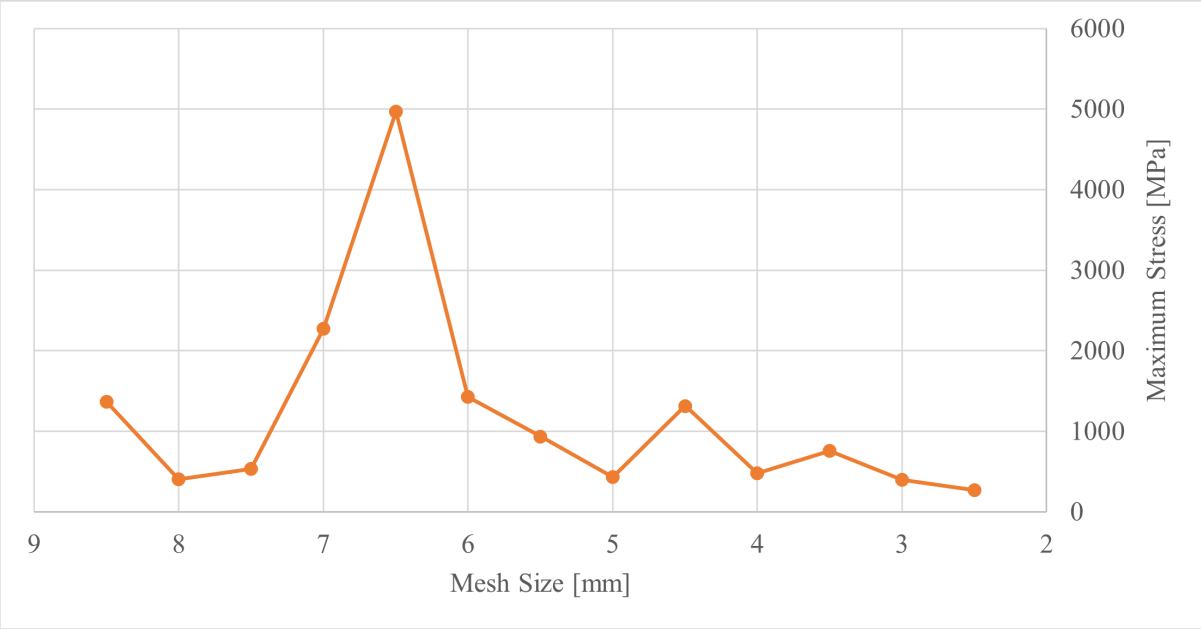


Figure 6: The mesh independenc table.

The trend toward constant stress values for mesh sizes finer than 4 mm suggests that the mesh is sufficiently detailed to capture the stress distribution within the component accurately. This finding establishes that a mesh size of 4 mm provides a balance between computational efficiency and the accuracy of the stress predictions. Based on these outcomes, a mesh size of 4 mm is used for further analysis, ensuring that the results are both computationally economical and reliable for engineering judgments. Figure 7 shows the mesh of the model.

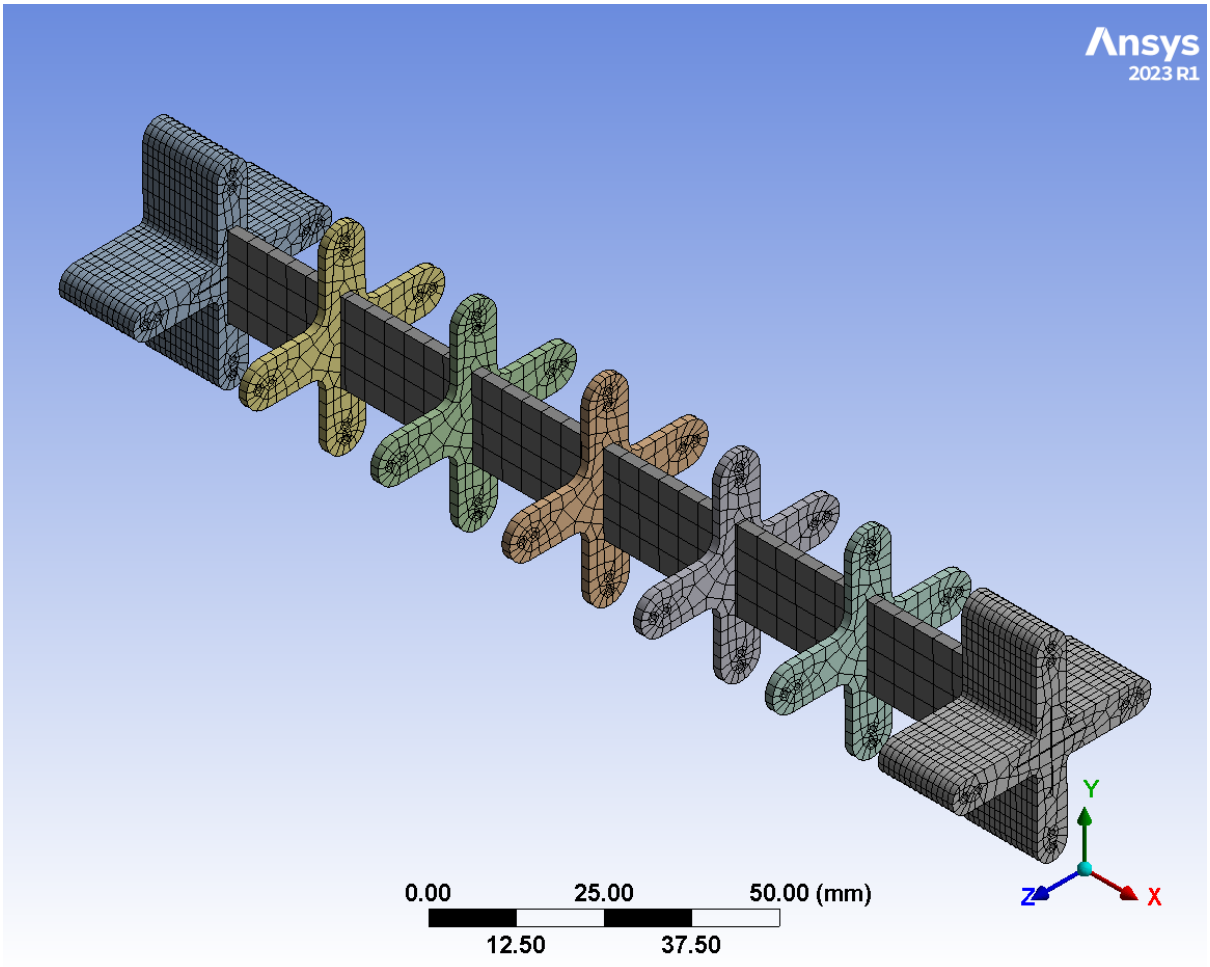


Figure 7: The mesh model of manipulator.

3.4.2 Forward Kinematics

The forward kinematics (FK) provided by Zhou et al. [3] were solely focused on a single segment with two modules. However, utilizing the FK algorithm for calculation became complex while there were four modules in the manipulator. The FK algorithm was rederived to generalize the workspace of the manipulator.

The base coordinate system (CS) was established with the centroid of base connector upper surface serving as the origin. The backbones of Module 1 and 3, which are Module (2,2) and (2,1), were parallel to the X-axis of base CS, while the backbones of Module 2 and 4, which are Modules (1,2) and (1,1), were parallel to the Y-axis of base CS. The upper surface centroids of connectors were designated as $node_1$, $node_2$, $node_3$, $node_4$, and $node_5$. The initial posture with CSs originated at $node_1 \sim node_5$ are shown in Figure 8.

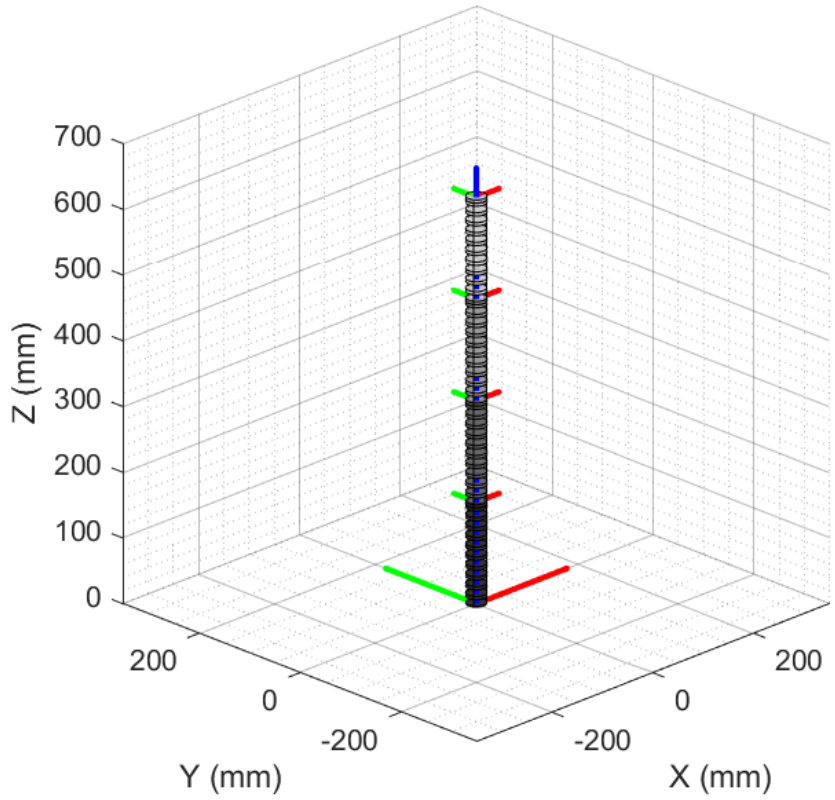
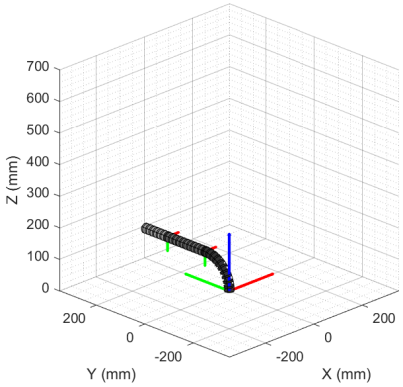
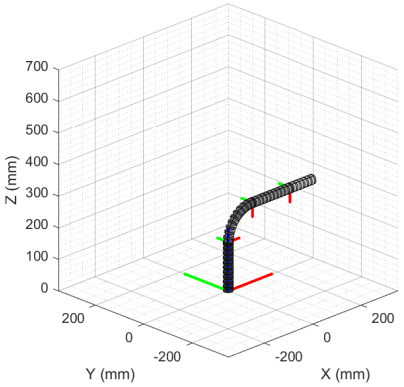


Figure 8: The kinematics model of manipulator at initial position.

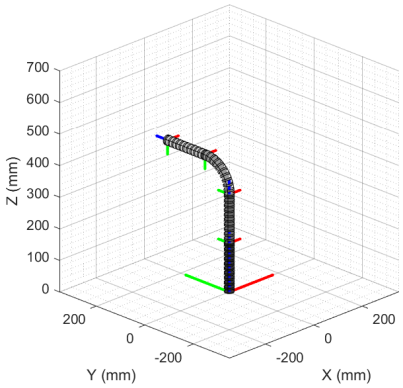
The Module 1 was restricted to bending in Y-Z plane of CS whose origin is $node_1$, while the Module 2 was restricted to bending in X-Z plane of CS whose origin is $node_2$. Similarly, the Module 3 and 4 were subject to the same constraints. The right angle bending postures of single modules are illustrated in Figure 9.



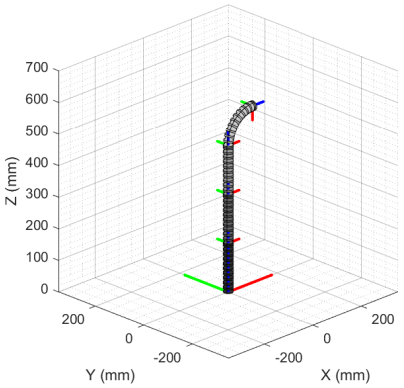
(a) $\alpha_1 = 90°, \alpha_2 = 0, \alpha_3 = 0, \alpha_4 = 0$



(b) $\alpha_1 = 0, \alpha_2 = 90°, \alpha_3 = 0, \alpha_4 = 0$



(c) $\alpha_1 = 0, \alpha_2 = 0, \alpha_3 = 90°, \alpha_4 = 0$



(d) $\alpha_1 = 0, \alpha_2 = 0, \alpha_3 = 0, \alpha_4 = 90°$

Figure 9: The right angle bending postures of single modules.

The bending angle α was defined to be positive while the module bent toward positive direction of the X-axis or Y-axis. Owing to the distinct properties of the four modules, diverse calculation methods were utilized. The Module i has a base node $node_i$ and an end effector node $node_{i+1}$. The position of $node_{i+1}$ in CS whose origin is $node_{i+1}$ was calculated in Equation 1.

$$\mathbf{P}_{i+1}^{base} = \mathbf{R}_i \times \mathbf{P}_{i+1}^{i} + \mathbf{P}_{i}^{base} \tag{1}$$

$\mathbf{P}_{i+1}^{base}$: The absolute position matrix (APM) of $node_{i+1}$ in the base CS.

$\mathbf{R}_i$: The rotational matrix transforms the base CS into CS i whose origin is $node_i$.

$\mathbf{P}_{i+1}^{i}$: The relative position matrix (RPM) of $node_{i+1}$ in CS i.

$\mathbf{P}_i^{base}$: The absolute position matrix (APM) of $node_i$ in the base CS.

- Module 1

  For Module 1, the RPM $\mathbf{P}_2^1$ with α_1 was shown in Equation 2. R_1 was the radius of the Module 1 bending curve. The rotational matrix $\mathbf{R}_1$ and the APM of $node_1$ $\mathbf{P}_1^{base}$ were shown in Equation 3.

$$\mathbf{P}_2^1 = \begin{bmatrix} 0 \\ (R_1 \cdot (1 - \cos(\alpha_1)) + d_2 \cdot \sin(\alpha_1)) \\ (R_1 \cdot \sin(\alpha_1) + d_2 \cdot \cos(\alpha_1)) \end{bmatrix} \tag{2}$$

$$(hint: \ R_1 = Sr_1/\alpha_1)$$

$$\mathbf{R}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathbf{P}_1^{base} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3}$$

  According to the Equations 2 and 3, the APM $\mathbf{P}_2^{base}$ was calculated in Equation 4.

$$\mathbf{P}_2^{base} = \mathbf{R}_1 \times \mathbf{P}_2^1 + \mathbf{P}_1^{base} \tag{4}$$

- Module 2

  For Module 2, the RPM $\mathbf{P}_3^2$ with α_2 was shown in Equation 5. R_2 was the radius of the Module 2 bending curve. The rotational matrix $\mathbf{R}_2$ and the APM $\mathbf{P}_2^{base}$ were shown in Equations 6 and 4.

$$\mathbf{P}_3^2 = \begin{bmatrix} (R_2 \cdot (1 - \cos(\alpha_2)) + d_3 \cdot \sin(\alpha_2)) \\ 0 \\ (R_2 \cdot \sin(\alpha_2) + d_3 \cdot \cos(\alpha_2)) \end{bmatrix} \tag{5}$$

$$(hint: \ R_2 = Sr_2/\alpha_2)$$

$$\mathbf{R}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha_1) & \sin(\alpha_1) \\ 0 & -\sin(\alpha_1) & \cos(\alpha_1) \end{bmatrix} \tag{6}$$

According to the Equations 4, 5, and 6, the APM $\mathbf{P}_3^{base}$ was calculated in Equation 7.

$$\mathbf{P}_3^{base} = \mathbf{R}_1 \times \mathbf{R}_2 \times \mathbf{P}_3^2 + \mathbf{P}_2^{base} \tag{7}$$

- Module 3

  For Module 3, the RPM $\mathbf{P}_4^3$ with α_3 was shown in Equation 8. R_3 was the radius of the Module 3 bending curve. The rotational matrix $\mathbf{R}_3$ and the APM $\mathbf{P}_3^{base}$ were shown in Equations 9 and 7.

$$\mathbf{P}_4^3 = \begin{bmatrix} 0 \\ (R_3 \cdot (1 - \cos(\alpha_3)) + d_4 \cdot \sin(\alpha_3)) \\ (R_3 \cdot \sin(\alpha_3) + d_4 \cdot \cos(\alpha_3)) \end{bmatrix} \tag{8}$$

$$(hint: \ R_3 = Sr_3/\alpha_3)$$

$$\mathbf{R}_3 = \begin{bmatrix} \cos(\alpha_2) & 0 & \sin(\alpha_2) \\ 0 & 1 & 0 \\ -\sin(\alpha_2) & 0 & \cos(\alpha_2) \end{bmatrix} \tag{9}$$

  According to the Equations 7, 8, and 9, the APM $\mathbf{P}_4^{base}$ was calculated in Equation 10.

$$\mathbf{P}_4^{base} = \mathbf{R}_1 \times \mathbf{R}_2 \times \mathbf{R}_3 \times \mathbf{P}_4^3 + \mathbf{P}_3^{base} \tag{10}$$

- Module 4

  In summary, the APM P_5^{base} was calculated in Equation 11. The calculation of $\times\mathbf{R}_4$ and $\mathbf{P}_5^4$ was similar to Equations 6 and 4.

$$\mathbf{P}_5^{base} = \mathbf{R}_1 \times \mathbf{R}_2 \times \mathbf{R}_3 \times \mathbf{R}_4 \times \mathbf{P}_5^4 + \mathbf{P}_4^{base} \tag{11}$$

Applying the corresponding transformations to the Equation 1 revealed the patterns

shown in Equations 12 and 13.

$$\mathbf{P}_{i+1}^{base} - \mathbf{P}_{i}^{base} = \prod_{n=1}^{i} \mathbf{R}_n \times \mathbf{P}_{i+1}^{i} \tag{12}$$

$$\mathbf{P}_{i+1}^{base} = \sum_{m=1}^{i} \left[\prod_{n=1}^{m} \mathbf{R}_n \times \mathbf{P}_{m+1}^{m} \right] \quad (\mathbf{P}_{1}^{base} = 0) \tag{13}$$

3.4.3 Inverse Kinematics

The inverse kinematics (IK) algorithm was inspired by a rigid manipulator IK solver called FABRIK [58]. W. Zhang et al.[59] expanded this solver to the continuum manipulator domain. Considering the distinctive nature of the manipulator, adaptations were implemented to the FABRIKc solver, transforming it into a tailored IK algorithm for this particular application. For further elucidation, the IK derivation was demonstrated using a target whose angles $\boldsymbol{target}$ = [0,0,90,0] °. The comprehensive process of the first epoch will be elaborated as follows. Simultaneously, the flowchart of IK algorithm was shown in Figure 10.

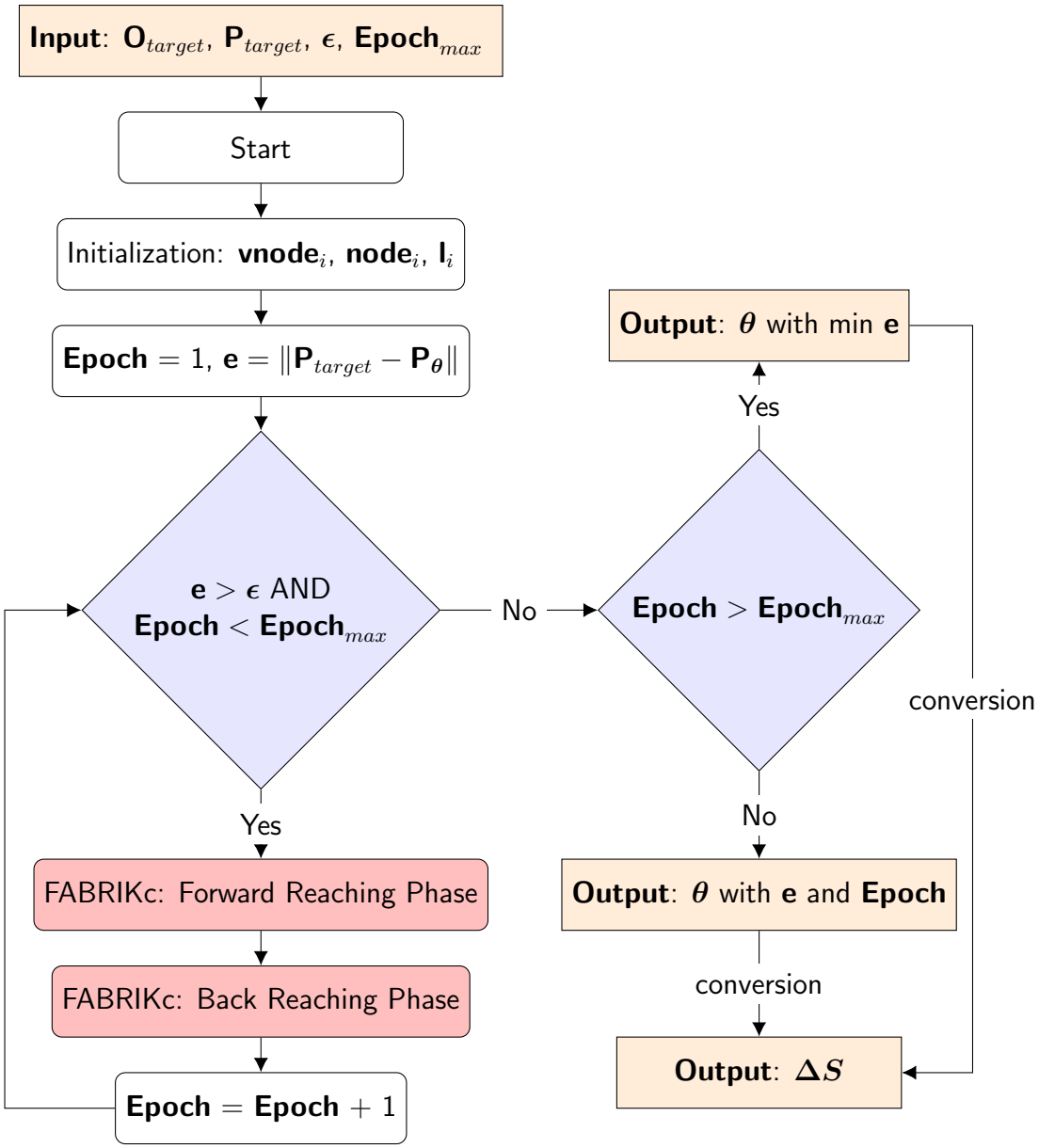


Figure 10: The flow chart of the IK algorithm.

The orientation and position matrices of the target were derived in Equation 14. These matrices were utilized as the input of the IK algorithm. The output of the algorithm was a series of bending angles $\boldsymbol{\theta}$.

$$\mathbf{O}_{target} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \mathbf{P}_{target} = \begin{bmatrix} 0 \\ 245.493 \\ 395.493 \end{bmatrix} \tag{14}$$

Additionally, it is essential to introduce a novel concept called "virtual node". This raised the fact that FABRIK algorithm was originally designed for rigid manipulators, which

implied that continuum manipulators need to be abstracted for modeling purpose. In this context, the module of manipulator were abstracted as a rigid manipulator with two prismatic joints and one revolute joint. The location of the revolute joint served as the virtual node.

Subsequently, the forward reaching phase of the FABRIKc algorithm can commence. The calculation which involves four iterations was specified as follows.

- Initialization

  The manipulator was estimated to stay at the initial position, which is shown in Figure 8. Under this circumstance, the nodes $\mathbf{node}_i$ and virtual nodes $\mathbf{vnode}_i$ of the manipulator were listed in Equations 15 and 16. The connector thickness of manipulator was not taken into account in this demonstration, despite being considered in the programme. The virtual length $\mathbf{l}_i$ for Module i, which is the distance from $\mathbf{vnode}_i$ to $\mathbf{node}_i$, was derived in Equation 17.

$$\mathbf{node}_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \mathbf{node}_2 = \begin{bmatrix} 0 \\ 0 \\ 150 \end{bmatrix} \mathbf{node}_3 = \begin{bmatrix} 0 \\ 0 \\ 300 \end{bmatrix} \mathbf{node}_4 = \begin{bmatrix} 0 \\ 0 \\ 450 \end{bmatrix} \mathbf{node}_5 = \begin{bmatrix} 0 \\ 0 \\ 600 \end{bmatrix} \tag{15}$$

$$\mathbf{vnode}_1 = \begin{bmatrix} 0 \\ 0 \\ 75 \end{bmatrix} \mathbf{vnode}_2 = \begin{bmatrix} 0 \\ 0 \\ 225 \end{bmatrix} \mathbf{vnode}_3 = \begin{bmatrix} 0 \\ 0 \\ 375 \end{bmatrix} \mathbf{vnode}_4 = \begin{bmatrix} 0 \\ 0 \\ 525 \end{bmatrix} \tag{16}$$

$$\mathbf{l}_i = \frac{Sr_i}{\theta_i} \cdot \tan(\theta_i) \quad (hint: \ \mathbf{l}_i = Sr_i/2 \ while \ \theta_i = 0) \tag{17}$$

$$\mathbf{l}_1 = \mathbf{l}_2 = \mathbf{l}_3 = \mathbf{l}_4 = 75 \ (unit: \ mm)$$

- Iteration 1

  The virtual node of Module 4 was derived by $\mathbf{O}_{target}$ and $\mathbf{P}_{target}$ in Equation 14. The virtual node $\mathbf{vnode}_4$ was derived by Equation 18.

$$\mathbf{vnode}_4^{'} = \mathbf{P}_{target} - \mathbf{l}_4 \cdot \mathbf{O}_{target}^{z} \quad (hint: \ ^{z} \ is \ z-axis \ orientation) \tag{18}$$

Afterwards, the vector from virtual node of Module 3 to virtual node of Module 4 was derived based on the positions of $\mathbf{vnode}_3$ and $\mathbf{vnode}_4^{'}$. The coordinate of the vector in the orientation of $\mathbf{vnode}_4^{'}$ was calculated in Equation 19.

$$\mathbf{vector}_4 = \mathbf{O}_{target} \times (\mathbf{vnode}_4^{'} - \mathbf{vnode}_3) \tag{19}$$

The bending angle of Module 4 θ_4 was derived in Equation 20. The Y-axis directional component was neglected because Module 4 can only bend in the X-Z plane. The updated length $\mathbf{l}_4^{update}$ was calculated according to Equation 21. Meanwhile, the updated $\mathbf{vnode}_4^{update}$ was rederive using $\mathbf{l}_4^{update}$ in Equation 22. The $\mathbf{node}_4$ was determined based on the orientation of $\mathbf{vnode}_4^{update}$, which is $\mathbf{O}_{vnode_4}$, in Equation 24.

$$\theta_4 = -arctan2(\mathbf{vector}_4^x, \mathbf{vector}_4^z) \tag{20}$$

$$\mathbf{l}_4^{update} = \frac{Sr_4}{\theta_4} \cdot \tan(\theta_4) \tag{21}$$

$$\mathbf{vnode}_4^{update} = \mathbf{P}_{target} - \mathbf{l}_4^{update} \cdot \mathbf{O}_{target}^z \tag{22}$$

$$\mathbf{O}_{vnode_4} = \begin{bmatrix} cos(\theta_4) & 0 & sin(\theta_4) \\ 0 & 1 & 0 \\ -sin(\theta_4) & 0 & sin(\theta_4) \end{bmatrix} \times \mathbf{O}_{target} \tag{23}$$

$$\mathbf{node}_4 = \mathbf{vnode}_4^{update} - \mathbf{l}_4^{update} \cdot \mathbf{O}_{vnode_4}^z \tag{24}$$

- Iteration 2

  The virtual node of Module 3 was derived by $\mathbf{O}_{vnode_4}$ and $\mathbf{node}_4$ in Equations 23 and 24. The virtual node $\mathbf{vnode}_3$ was derived by Equation 25.

$$\mathbf{vnode}_3^{'} = \mathbf{node}_4 - \mathbf{l}_3 \cdot \mathbf{O}_{vnode_4}^z \tag{25}$$

Afterwards, the vector from virtual node of Module 2 to virtual node of Module 3 was derived based on the positions of $\mathbf{vnode}_2$ and $\mathbf{vnode}_3^{'}$. The coordinate of the vector in the orientation of $\mathbf{vnode}_3^{'}$ was calculated in Equation 26.

$$\mathbf{vector}_3 = \mathbf{O}_{vnode_3} \times (\mathbf{vnode}_3^{'} - \mathbf{vnode}_2) \tag{26}$$

The bending angle of Module 3 θ_3 can be derived in Equation 27. The X-axis directional component was neglected because Module 3 can only bend in the Y-Z plane. The updated length $\mathbf{l}_3^{update}$ was calculated according to Equation 28. Meanwhile, the updated $\mathbf{vnode}_3^{update}$ was rederived using $\mathbf{l}_3^{update}$ in Equation 29. The $\mathbf{node}_3$ was determined based on the orientation of $\mathbf{vnode}_3^{update}$, which is $\mathbf{O}_{vnode_3}$, in Equation 31.

$$\theta_3 = -arctan2(\mathbf{vector}_3^y, \mathbf{vector}_3^z) \tag{27}$$

$$\mathbf{l}_3^{update} = \frac{Sr_3}{\theta_3} \cdot \tan(\theta_3) \tag{28}$$

$$\mathbf{vnode}_3^{update} = \mathbf{node}_4 - \mathbf{l}_3^{update} \cdot \mathbf{O}_{vnode_4}^z \tag{29}$$

$$\mathbf{O}_{vnode_3} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_3) & sin(\theta_3) \\ 0 & -sin(\theta_3) & sin(\theta_3) \end{bmatrix} \times \mathbf{O}_{vnode_4} \tag{30}$$

$$\mathbf{node}_3 = \mathbf{vnode}_3^{update} - \mathbf{l}_3^{update} \cdot \mathbf{O}_{vnode_3}^z \tag{31}$$

- Iteration 3

  The virtual node of Module 2 was derived by $\mathbf{O}_{vnode_3}$ and $\mathbf{node}_3$ in Equations 30 and 31. The virtual node $\mathbf{vnode}_2$ was derived by Equation 32.

$$\mathbf{vnode}_2^{'} = \mathbf{node}_3 - \mathbf{l}_2 \cdot \mathbf{O}_{vnode_3}^z \tag{32}$$

Afterwards, the vector from virtual node of Module 1 to virtual node of Module 2 was derived based on the positions of $\mathbf{vnode}_1$ and $\mathbf{vnode}_2^{'}$. The coordinate of the vector in the orientation of $\mathbf{vnode}_2^{'}$ was calculated in Equation 33.

$$\mathbf{vector}_2 = \mathbf{O}_{vnode_2} \times (\mathbf{vnode}_2^{'} - \mathbf{vnode}_1) \tag{33}$$

The bending angle of Module 2 θ_2 was derived in Equation 34. The Y-axis directional component was neglected because Module 2 can only bend in the X-Z plane. The updated length $\mathbf{l}_2^{update}$ was calculated according to Equation 35. Meanwhile, the updated $\mathbf{vnode}_2^{update}$ was rederived using $\mathbf{l}_2^{update}$ in Equation 36. The $\mathbf{node}_2$ was determined based on the orientation of $\mathbf{vnode}_2^{update}$, which is

$\mathbf{O}_{vnode_2}$, in Equation 38.

$$\theta_2 = -arctan2(\mathbf{vector}_2^x, \mathbf{vector}_2^z) \tag{34}$$

$$\mathbf{l}_2^{update} = \frac{Sr_2}{\theta_2} \cdot \tan(\theta_2) \tag{35}$$

$$\mathbf{vnode}_2^{update} = \mathbf{node}_3 - \mathbf{l}_2^{update} \cdot \mathbf{O}_{vnode_3}^z \tag{36}$$

$$\mathbf{O}_{vnode_2} = \begin{bmatrix} cos(\theta_2) & 0 & sin(\theta_2) \\ 0 & 1 & 0 \\ -sin(\theta_2) & 0 & sin(\theta_2) \end{bmatrix} \times \mathbf{O}_{vnode_3} \tag{37}$$

$$\mathbf{node}_2 = \mathbf{vnode}_2^{update} - \mathbf{l}_2^{update} \cdot \mathbf{O}_{vnode_2}^z \tag{38}$$

- Iteration 4

  The virtual node of Module 1 was derived by $\mathbf{O}_{vnode_2}$ and $\mathbf{node}_2$ in Equations 37 and 38. The virtual node $\mathbf{vnode}_1$ was derived by Equation 39.

$$\mathbf{vnode}_1^{'} = \mathbf{node}_2 - \mathbf{l}_1 \cdot \mathbf{O}_{vnode_2}^z \tag{39}$$

  Afterwards, the coordinate of the vector in the orientation of $\mathbf{vnode}_1^{'}$ was directly calculated in Equation 40. This was because the $\mathbf{O}_{node_1}^z$ must be oriented vertically upward.

$$\mathbf{vector}_2 = \mathbf{O}_{vnode_1} \times \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \tag{40}$$

  The bending angle of Module 1 θ_1 was derived in Equation 41. The X-axis directional component was neglected because Module 1 can only bend in the Y-Z plane. The updated length $\mathbf{l}_1^{update}$ was calculated according to Equation 42. Meanwhile, the updated $\mathbf{vnode}_1^{update}$ was rederived using $\mathbf{l}_1^{update}$ in Equation 43. The $\mathbf{node}_1$ was determined based on the orientation of $\mathbf{vnode}_1^{update}$, which is $\mathbf{O}_{vnode_1}$, in Equation 45.

$$\theta_1 = -arctan2(\mathbf{vector}_1^y, \mathbf{vector}_1^z) \tag{41}$$

$$\mathbf{l}_1^{update} = \frac{Sr_1}{\theta_1} \cdot \tan(\theta_1) \tag{42}$$

$$\mathbf{vnode}_1^{update} = \mathbf{node}_2 - \mathbf{l}_1^{update} \cdot \mathbf{O}_{vnode_2}^z \tag{43}$$

$$\mathbf{O}_{vnode_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta_1) & sin(\theta_1) \\ 0 & -sin(\theta_1) & sin(\theta_1) \end{bmatrix} \times \mathbf{O}_{vnode_2} \tag{44}$$

$$\mathbf{node}_1 = \mathbf{vnode}_1^{update} - \mathbf{l}_1^{update} \cdot \mathbf{O}_{vnode_1}^{z} \tag{45}$$

Ultimately, the FABRIKc algorithm initiated the backward reaching phase, which is the FK algorithm. The FABRIKc algorithm inherently guaranteed the orientation of manipulator's end effector, with the necessity to account solely for positional errors. The error was calculated by Equation 46.

$$\mathbf{e} = \|\mathbf{P}_{target} - \mathbf{P}_{\boldsymbol{theta}}\| \tag{46}$$

For the first epoch, the results of IK algorithm were $\boldsymbol{\theta} = [6.71,\ -0.0,\ 83.29,\ -0.0]^{\circ}$, The error of the first epoch was 33.77506. The results about IK algorithm with the target angle $\boldsymbol{\alpha} = [0,\ 0,\ 90,\ 0]$ are shown in Table A.1 in Appendix A. The algorithm was configured to execute a maximum of 200 epochs, with the provision to pause when the error reaches a sufficiently low level, thereby conserving computational resources. The maximum epoch number was defined as $Epoch_{max}$.

3.4.4 Angular Conversion

The Arduino programme utilized $\boldsymbol{\Delta S}$ as input to control the motors. However, the solutions of IK algorithm are $\boldsymbol{\theta}$, which are angles. It is vital to convert angles $\boldsymbol{\theta}$ into $\boldsymbol{\Delta S}$. The variables R_1, R_2, and R_3 were mentioned in Equations 2, 5, and 8, while $R_4 = Sr_4/\alpha_4$. The relationship between $\boldsymbol{\theta}$ and $\boldsymbol{\Delta S}$ was shown in Equations 47, 48, 49, 50. The relationship can be further generalized into matrix in Equation 51 for Python programme.

$$\Delta S_1 = (R_1 + r_1) \cdot \alpha_1 - Sr_1 = \alpha_1 \cdot r_1 \quad \Delta S_2 = (R_1 - r_1) \cdot \alpha_1 - Sr_1 = -\alpha_1 \cdot r_1 \tag{47}$$

$$\Delta S_3 = (R_2 + r_2) \cdot \alpha_2 - Sr_2 = \alpha_2 \cdot r_2 \quad \Delta S_4 = (R_2 - r_2) \cdot \alpha_2 - Sr_2 = -\alpha_2 \cdot r_2 \tag{48}$$

$$\Delta S_5 = \Delta S_1 + (R_3 + r_3) \cdot \alpha_3 - Sr_3 = \Delta S_1 + \alpha_3 \cdot r_3$$

$$\Delta S_6 = \Delta S_2 + (R_3 - r_3) \cdot \alpha_3 - Sr_3 = \Delta S_2 - \alpha_3 \cdot r_3 \tag{49}$$

$$\Delta S_7 = \Delta S_3 + (R_4 + r_4) \cdot \alpha_4 - Sr_4 = \Delta S_3 + \alpha_4 \cdot r_4$$
$$\Delta S_8 = \Delta S_4 + (R_4 - r_4) \cdot \alpha_4 - Sr_4 = \Delta S_4 - \alpha_4 \cdot r_4 \tag{50}$$

$$\boldsymbol{\Delta S} = \begin{bmatrix} r_1 & 0 & 0 & 0 \\ -r_1 & 0 & 0 & 0 \\ 0 & r_2 & 0 & 0 \\ 0 & -r_2 & 0 & 0 \\ r_1 & 0 & r_3 & 0 \\ -r_1 & 0 & -r_3 & 0 \\ 0 & r_2 & 0 & r_4 \\ 0 & -r_2 & 0 & -r_4 \end{bmatrix} \times \boldsymbol{\alpha} \tag{51}$$

3.5 Electronic Control

3.5.1 Actuation Control

As mentioned above, the manipulator is divided into four units in total, with each unit being controlled for its bending movements by two cables. This means that the entire manipulator is controlled by a total of 8 cables. How to control the extension and retraction of these 8 cable ropes is the key to the Control part.

In this project, 8 stepper motors are employed to control the 8 individual cables. The planned design involves wrapping the cables around the rotor of each motor, and the stepper motors are programmed to extend or retract the ropes by stepping a certain number of steps in either a counterclockwise or clockwise direction. These 8 motors are connected to an Arduino board together, and the user can control the programme by a keyboard, and a serial monitor.

- Actuation Principle

  In this programme, the user inputs the desired length changes for 8 cables ΔS_1 ΔS_8 ($\Delta S_{1,2,3,4}$ ranging from -36.65 mm to 36.65 mm, $\Delta S_{5,6,7,8}$ ranging from -68.07 to 68.07 mm). After conversion, it outputs the corresponding step numbers $Step_1$ $Step_8$ for the eight stepper motors. Based on this, the 8-stepper motors will step

correspondingly. The relationship between the changes in cable length and the motor steps can be defined by Equation 52.

$$Step_i = \frac{(\Delta S_i - \Delta S_{i(prev)}) \cdot N}{\pi \cdot d},\ i \in [1, 8] \tag{52}$$

$\Delta S_{i(prev)}$: The length change for i^{th} cable in the previous location.

d: The diameter of the rotor. A rotor of diameter 10 mm is used currently.

N: Number of steps/rev. 2048 for 28YBG-48 motor.

- Component Selection

  To achieve a successful and precise control functionality, the first thing to do is to select appropriate components. Below are the main electronic components chosen:

  - Arduino Mega 2560 board:

    Because the project requires connecting 8 motors at the same time, the board should have more pins on the board as well as stronger processing capabilities compared to ordinary boards. Therefore, the Arduino Mega 2560 has been selected. It boasts up to 54 I/O pins and a more powerful processor to allow for better performance in multitasking operations [60].

  - 28YBJ-48 stepper motor:

    The 28YBJ-48 stepper motor is currently one of the most commonly used stepper motors on the market. It operates in two modes: half-step and full-step, corresponding to 4096 steps/rev and 2048 steps/rev respectively. This high-resolution configuration allows the manipulator to perform high-precision operations. Additionally, at its rated voltage, it can provide a torque of up to 34Nm, which is more than sufficient for pulling the cables of a lightweight mechanical arm.

  - ULN 2003A motor control chip:

    The motor control chip acts as the brain of the motor, responsible for translating instructions from the Arduino board into operations that the motor can execute. The ULN2003A chip is specifically matched with the 28YBJ-48 motor, ensuring seamless coordination and operation between the

chip and the motor.

– 9V power supply:

The rated operating voltage of the stepper motor is 5-12V. Here, a 9V power supply is chosen to allow losses due to component resistance.

- Circuit Schematics

The schematic of the overall circuit for the manipulator actuation control part is shown in Figure 11.

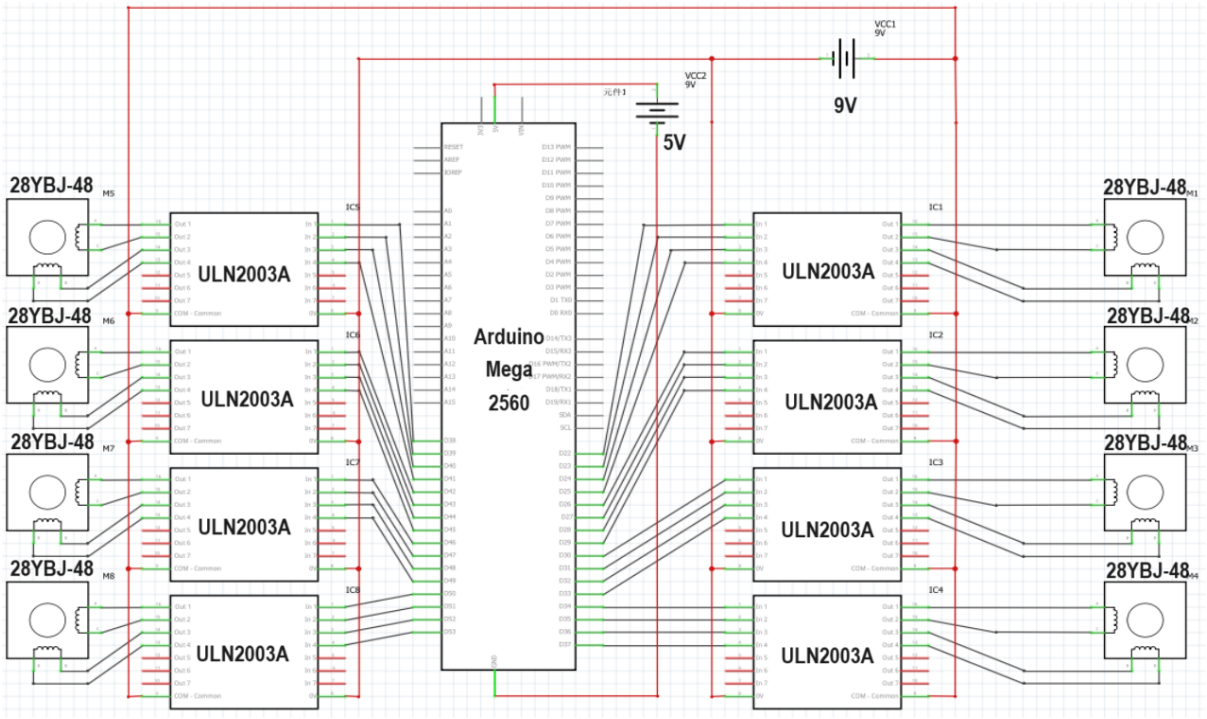


Figure 11: The circuit layout of Arduino step motor control system.

The 8 motors use pin 23-52 on the Arduino Mega 2560 board, and are powered by a 9V battery. The power supply is in a parallel configuration on a breadboard. The Arduino board is connected to a serial monitor and a keyboard for state inspection and manual input.

- Programme development

The main logic framework of the programme has been illustrated by the flowchart shown in Figure 12.

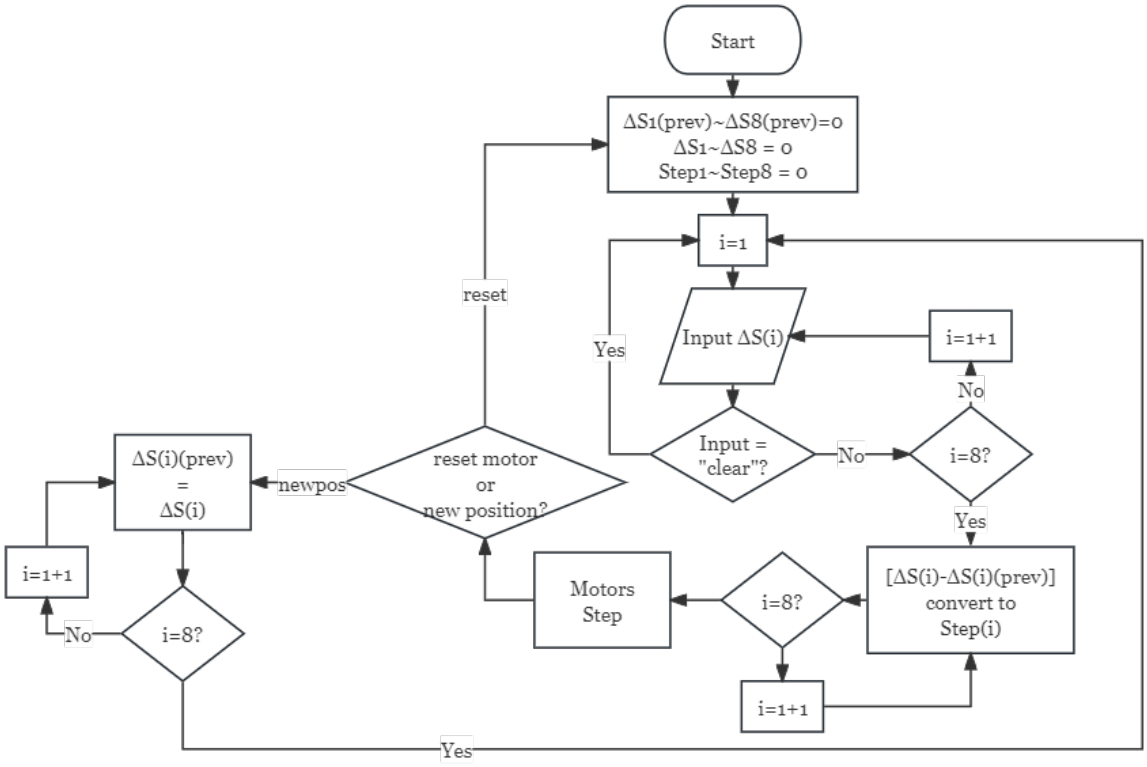


Figure 12: The flow chart of Arduino step motor control programme.

After the programme starts, the first thing is to declare the variables. $\Delta S_{i(prev)}$, ΔS_i and $Step_i$ is all set to 0 initially.

Then the programme asks the users to input 8 numbers representing the length change for 8 cables. The 8 values will be assigned to variable ΔS_1 ΔS_8 in order. During the input process, the user can input 'clear' instead of numbers to clear all the previous input and start over again from ΔS_1.

Once the input process is complete, the programme converts the ΔS_1 ΔS_8 to $Step_1$ $Step_8$ based on Equation 52, with the d set to 10 mm, N set to 2048 steps/rev. The motors will start to step immediately after the conversion is finished.

During the stepping process, the motors controlling the first and second unit (the first 4 motors) are set to have a stepping speed of 300 steps/second, and the motors controlling the third and fourth unit (the last 4 motors) are set to have $\frac{300}{17.5} \cdot (17.5 + 15) \approx 557$ steps/second. This is because the moving speed for the last two units should be $\frac{17.5+15}{17.5}$ times faster than the first two units so that the 4 units can finish moving approximately at the same time.

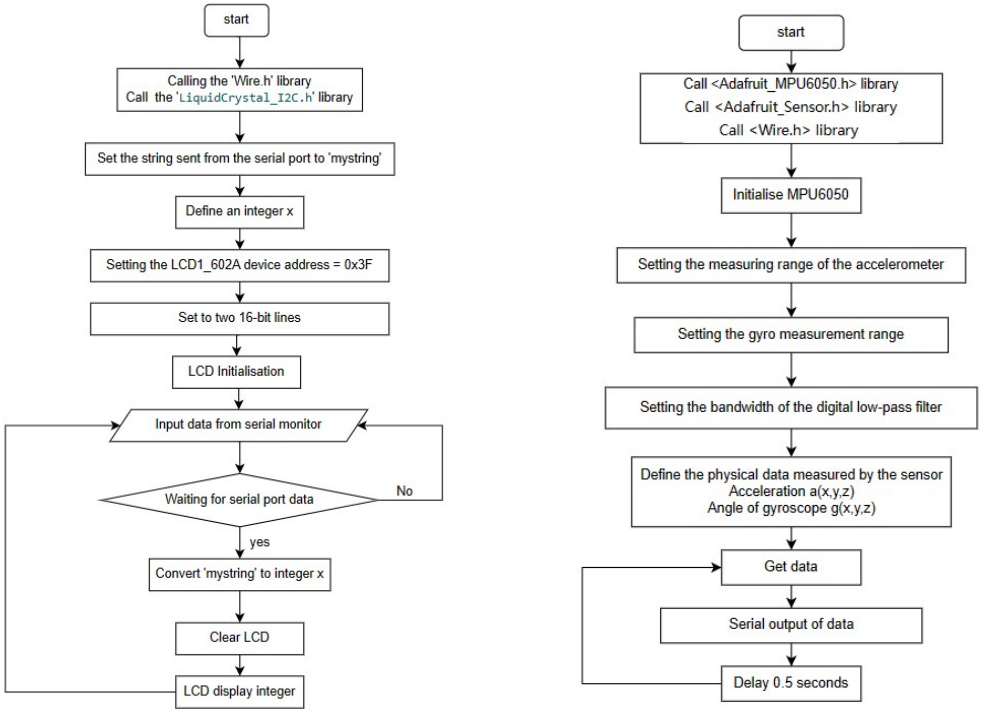
The programme shall wait until the motors finished stepping. Once the stepping

process is finished, the programme asks the user what to do next: reset the motors to initial position or move from the current position to the next position. If the user chooses to reset the motors, all the motors rotate backwards for same steps. However, If the user chooses to let the motors move to a new position, the values for current $\Delta S_1 \quad \Delta S_8$ will be stored to variable $\Delta S_{1(prev)} \quad \Delta S_{8(prev)}$. The programme then asks for a new set of $\Delta S_1 \quad \Delta S_8$, then calculate the new $Step_1 \quad Step_8$. Hence, the loop of the programme is closed.

3.5.2 Parameter Input by Serial Monitor

In the operating logic, the control parameters are inputted through the keyboard to control the motion of the multiple stepper motors. The basic function is to input the steps into the serial monitor through the keyboard of the host computer and the input string will be read by the microcontroller and converted into data, which are used as the number of steps of the stepper motor movement. For easy operation and display, the input parameters need to be displayed as text on the screen of the LCD IIC. LCD IIC will be in the form of a 2-line, 16-bit display to show more content.

Logic flowchart of the code to control the motor and LCD IIC by inputting parameters with the serial monitor of the host computer is shown in Figure 13a.



(a) The code logic of using serial to input command parameter

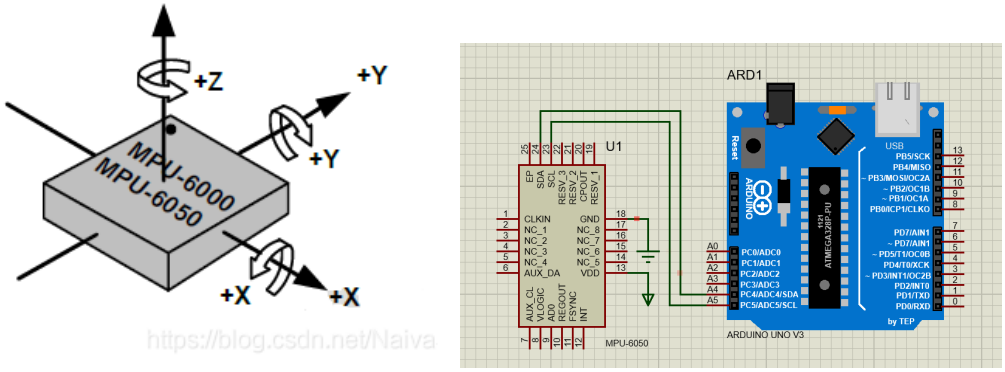
(b) The code logic of using MPU6050 to monitor gyro angles and accelerations

Figure 13: The code logic of input parameters and MPU6050.

3.5.3 Verification of MPU 6050

According to the project content, adding sensors to the end effector to measure its orientation angle and distance from the contact plane can assist the user to carry out more accurate operation. The function of detecting the orientation angle is realised by the MPU6050.

The Orientation angle (Euler angle) corresponding to the MPU6050 chip is shown in Figure 14a. Pitch is rotation around the x-axis. Yaw is rotation around the y-axis. Roll is rotation around the z-axis.



(a) The MPU6050 chip corresponding to the Euler angle

(b) The wiring diagram of MPU6050 and microcontroller

Figure 14: The MPU6050 chip with ancillary Arduino circuit.

The MPU6050 requires an external 5V power supply and GND. SCL is the Clock Pin of the IIC when the MPU6050 is used as a slave, and SDA is the Data Pin of the IIC when the MPU6050 is used as a slave. The wiring diagram is shown in Figure 14b.

The logic flowchart of the code that controls the MPU 6050 to display acceleration and direction angle is shown in Figure 13b. In this project, the acceleration parameters will not be acquired and displayed.

Since Proteus does not support dynamic simulation of the MPU6050, the verification of the MPU6050 has been carried out by physical parts and Arduino Mega while the project is in progress. The validation results have an acceptable accuracy and are able to display the corresponding orientation angle on the serial port and OLED screen.

3.5.4 Using HC-SR04 to measure distance

The HC-SR04 Ultrasonic Sensor uses sonar to determine the distance to an object. It has an ultrasonic transmitter and a receiver module which provides excellent non-contact range detection, high accuracy and stable readings. The measurement range is from 2 to 400 cm. The resolution is 0.3 cm and the measurement angle is within 30°. It operates in a process unaffected by sunlight or ferrous materials.

The sensor is triggered by sending a HIGH pulse of 10 microseconds. Previously, the speaker gives a short low-level pulse to ensure that a clean HIGH pulse is obtained.

The basic principle of distance measurement is Time-of-Flight (ToF). The formula is as

follows.

$$distance = (duration/2) \times v_{sound} \tag{53}$$
$$v_{sound} = 343\ m/s = 1/29.1\ cm/us$$

The wiring diagram and simulation results are shown in Figure 15. An SSD 1306 OLED screen is used to display the data in the simulation. The logic flowchart of the code that controls the HC-SR04 to measure distance and display acceleration by OLED screen is shown in Figure 16a.

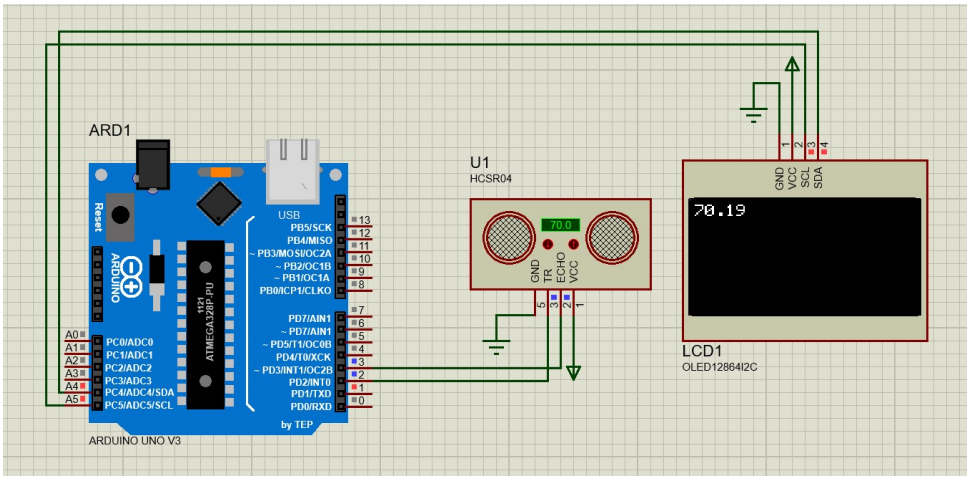
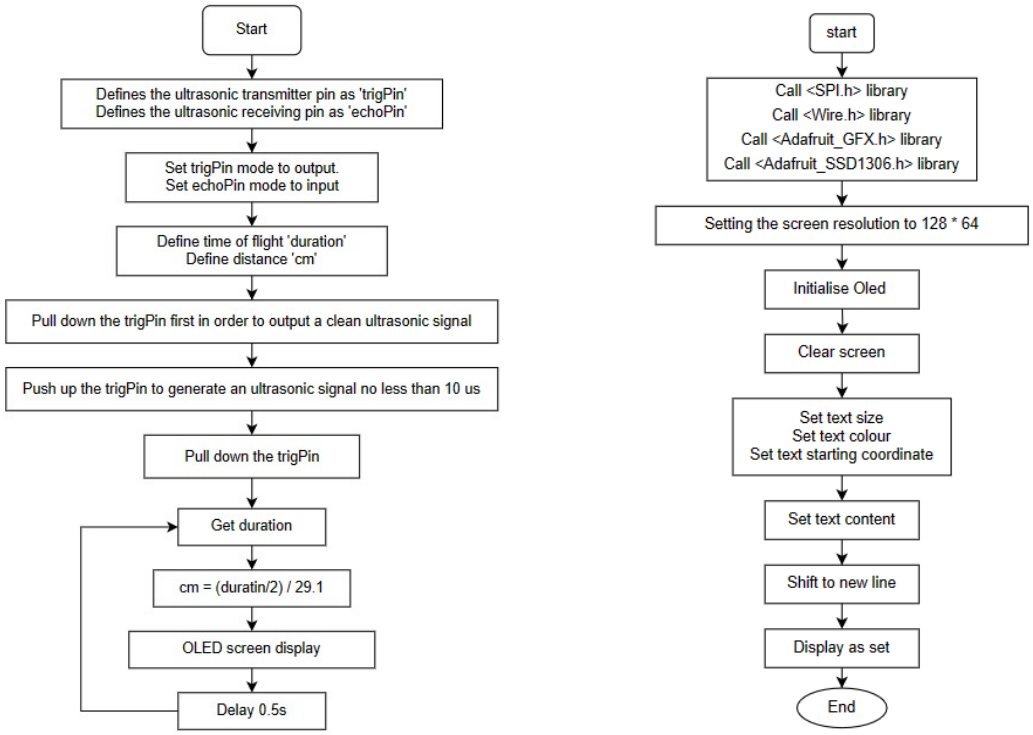


Figure 15: The wiring diagram and simulation of HC-SR04.



(a) The code logic of using HC-SR04 to measure distance

(b) The code logic of using SSD 1306 OLED to display data

Figure 16: The code logic of HC-SR04 and SSD 1306 OLED.

3.5.5 SSD 1306 OLED

Using SSD 1306 OLED requires calling two libraries, <Adafruit_SSD1306.h> and <Adafruit_GFX.h>. <Adafruit_SSD1306.h> is a dedicated display library for SSD1306 OLED screens. <Adafruit_GFX.h> library is the common parent graphics library for LCD and OLED screens.

The resolution of SSD1306 is 128x64 pixel dot array, SCK is the clock pin, SDA is the data pin, and external power supply and GND are required. Figure 16b is the logic flowchart of the code to control the OLED to display the text content.

4 Result and Discussion

4.1 Strength Analysis

4.1.1 The Result of Strength Analysis

The strength analysis conducted using ANSYS has yielded the following results.

- Displacement

  The maximum displacement observed in the manipulator was 117.44 mm, located at the end-effector, which is shown in Figure 17.

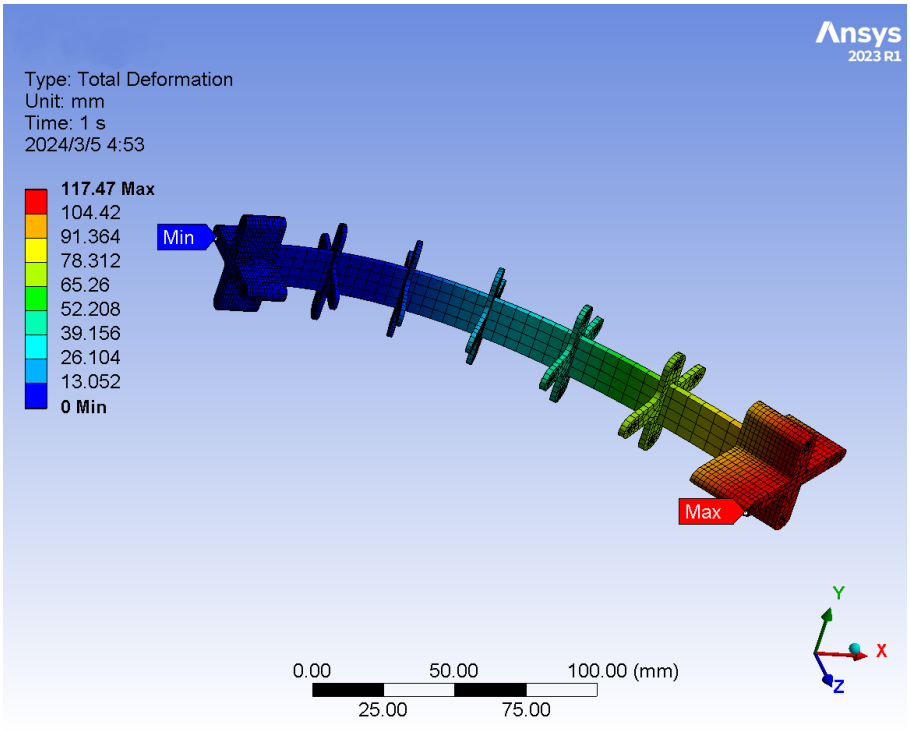


Figure 17: The displacement analysis of manipulator.

- Stress

  As shown in Figure 18, the maximum stress was found to be 476.63 MPa, occurring at the base of elastic sheet where the manipulator experiences the most significant load.

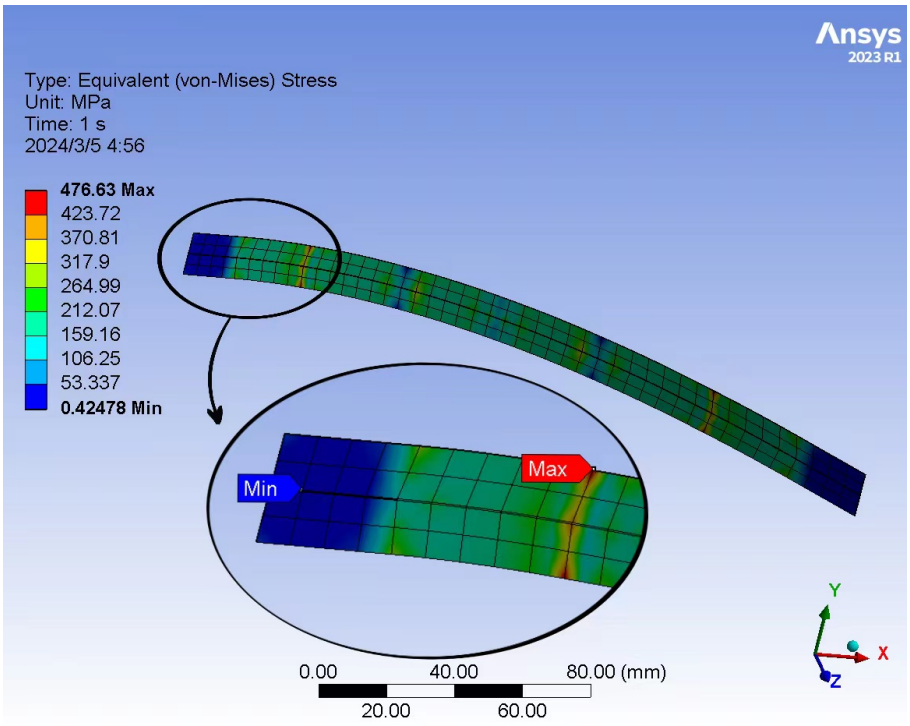


Figure 18: The stress analysis of manipulator.

The results indicate that Maximum stress is concentrated on the elastic sheet of the manipulator near the connector, which suggests that this is the most likely location for failure under the current design and loading conditions. The maximum von Mises stress did not exceed the typical yield strength of 65Mn alloy teel, indicating that the arm would return to its original shape after the removal of the load and there is no potential risk of plastic deformation or failure. This enhances the arm's reliability and extends its operational lifespan.

Given that the material remains within its elastic limit, the design could potentially be optimized further to reduce weight or increase load capacity. Future work may explore these possibilities while maintaining the safety factors within the design criteria.

4.1.2 Limitations and Solutions

- Simplification of the Model

  The initial analysis simplified the complex structure of the manipulator by focusing on a single module, potentially overlooking critical interactions between various modules that contribute to the overall strength and performance of the system. This simplification, while beneficial for computational efficiency, may not

accurately reflect the integrated behavior of the manipulator in real-world scenarios where interactions between modules can significantly influence outcomes.
Solution: Future analyses could incorporate the interactions between all modules to provide a more comprehensive understanding of the manipulator's behavior under various conditions. Such an approach will ensure a more accurate representation of the manipulator's behavior under various operational conditions, leading to more reliable performance predictions and informed design decisions.

- Failure Point Concentration

  Stress concentration at the base of the elastic sheet indicates a potential failure point but does not account for fatigue or long-term wear. This oversight could underestimate the long-term durability of the manipulator, potentially leading to unexpected maintenance issues or failure.
  Solution: To address this gap, incorporating a detailed fatigue analysis into future work is crucial. This analysis should evaluate the impact of repeated stress cycles on the manipulator's components over extended periods. By understanding these effects, design modifications can be implemented to enhance the durability and operational lifespan of the manipulator. The modifications could include optimizing material selection, redesigning stress-prone areas to reduce concentration points, and introducing preventive maintenance schedules based on predicted failure.

- Impact of cables

  The analysis did not consider the impact of cable tension on the manipulator's strength. Cable tension can significantly affect the manipulator's mechanical behavior. Neglecting this factor may lead to an incomplete understanding of the manipulator's operational capabilities and potential failure modes.
  Solution: Future analyses should integrate the effects of cable tension to achieve a more accurate assessment of the manipulator's performance. This could involve modifying the existing model to include boundary conditions that accurately represent the forces exerted by cables under various load scenarios. Implementing

dynamic simulations that account for variations in tension forces during different stages of operation will further enhance the fidelity of the model. These improvements will enable a more nuanced understanding of the manipulator's capabilities, assisting in the design of a more robust and reliable system capable of withstanding the complexities of real-world applications.

4.2 Error Analysis of Cross-Shaped Sheets

At the early stage of the project, given that the number of cross-shape sheets in each module may have an influence on the maximum bending angle $\boldsymbol{\alpha}_{max}$, which is an essential parameter of those which affect the range of working space of the manipulator, hence the number of sheets should be determined by deriving the functions of both absolute error and relative error [3] against the number of sheets. The simulation results are shown in Figure 19.

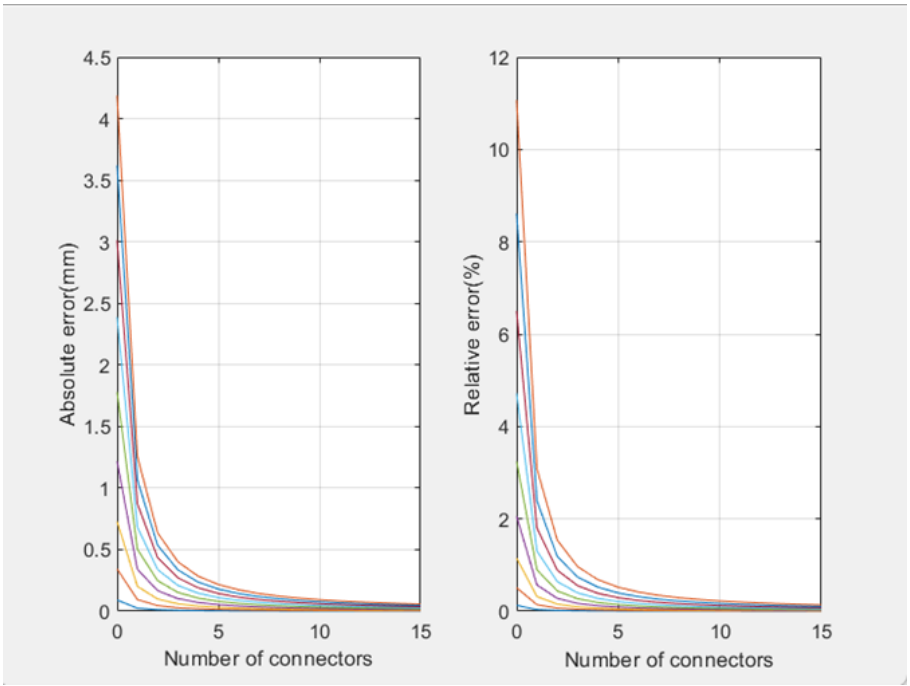


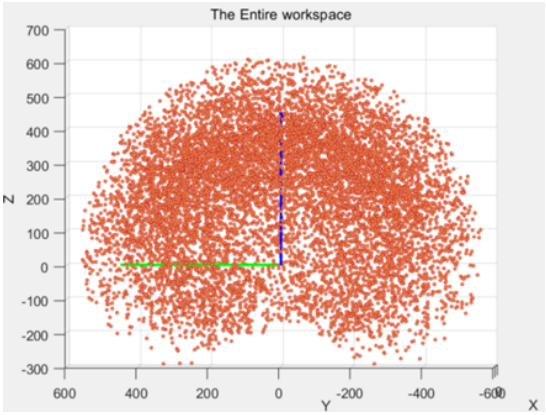
Figure 19: The simulation results of errors against the number of connecting sheets.

As illustrated in Figure 19, the changing trends of both absolute and relative errors are almost identical since the relative error is proportional to the absolute error, and the different lines with different colors represent the data with different bending angles whose range is from 10 degrees to 90 degrees in both directions (positive or negative). It

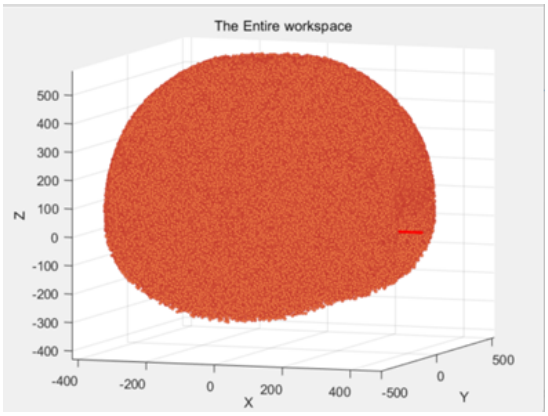
is obvious that both errors decrease with the increase in the number of connecting sheets, hence theoretically, the number of sheets should be as large as possible to make both errors could be negligible. However, the number of sheets also has an influence on the size of simulation space, more the number of sheets less the size of simulation space given that if the length of manipulator is much longer, the space near to the basement of manipulator could not be reached due to the characteristics of the material. Hence, the number of sheets could be assumed to be 10 rather than 15, even though both the errors are smaller when the number of sheets is 15.

4.3 Manipulator Workspace Analysis

The workspace simulation of the manipulator has been illustrated since it is important to determine the variables of joints based on the required workspace. The segmented workspace is affected by several aspects, which include the simulation index and the range within the three-dimensional coordinates. The effective workspace of the manipulator is a cubic whose dimension is 300x300x300 (mm). The specific workspace has a fixed range on the x and y axes from -150 to 150 mm. Therefore, it is only necessary to determine the range of the workspace along the z-axis. The parameter H, representing the height of the basement of specific workspace, is utilized to determine the range of the workspace along the z-axis. The impacts of different indices and parameter H will be discussed separately, aiming to segment the appropriate cubic workspace of the manipulator.



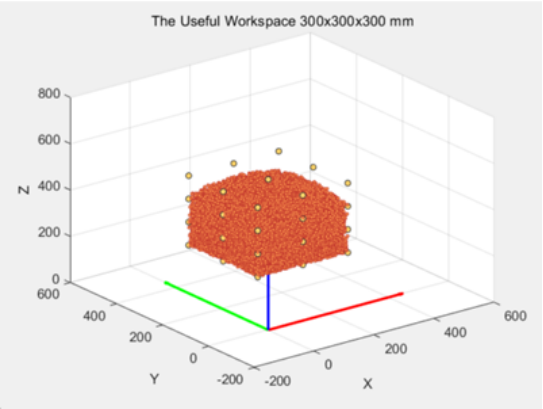
(a) The entire workspace where simulation index=10000



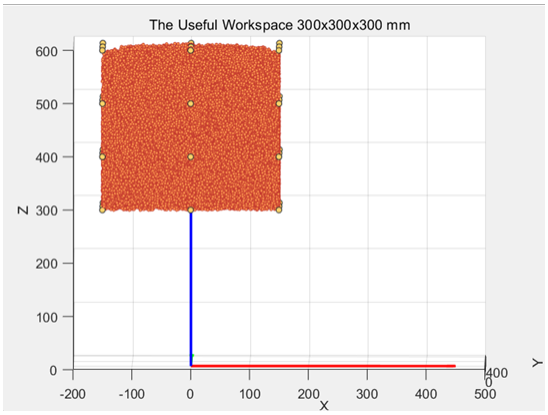
(b) The entire workspace where simulation index=1000000

Figure 20: The entire workspace with different simulation indices.

As shown in Figure 20a, the working space of manipulator forms a spherical shell-shaped point cloud consisting of 10000 floating points since the simulation index is 10000, which is insufficient to fully occupied the theoretical workspace. To further generalize the theoretical workspace, it is necessary to increase the simulation index. The appropriate workspace should be divided from the workspace whose index is 1000000.



(a) The cubic workspace of manipulator with index = 1000000 H=350-650 mm



(b) index = 1000000 H=300-600 mm

Figure 21: The cubic workspace with different segmentation.

Comparing the simulation results illustrated in Figure 21, it can be observed that the upper area of the segmented cubic workspace whose H=350mm failed to be occupied. However, the segmented cubic workspace whose H=300mm is almost fully occupied. To determine threshold of parameter H, a series of yellow detection points are utilized to test whether specific locations in the workspace can be reached. A programme is utilized to calculate the distance between the detection points and the simulated point cloud and identify the minimum distances. The simulation results whose index=1000000 and H=300mm are shown in Figure B.6 in Appendix B. The top view of the cubic workspace in Figure 22 associates labels with detection points, facilitating subsequent explanations.

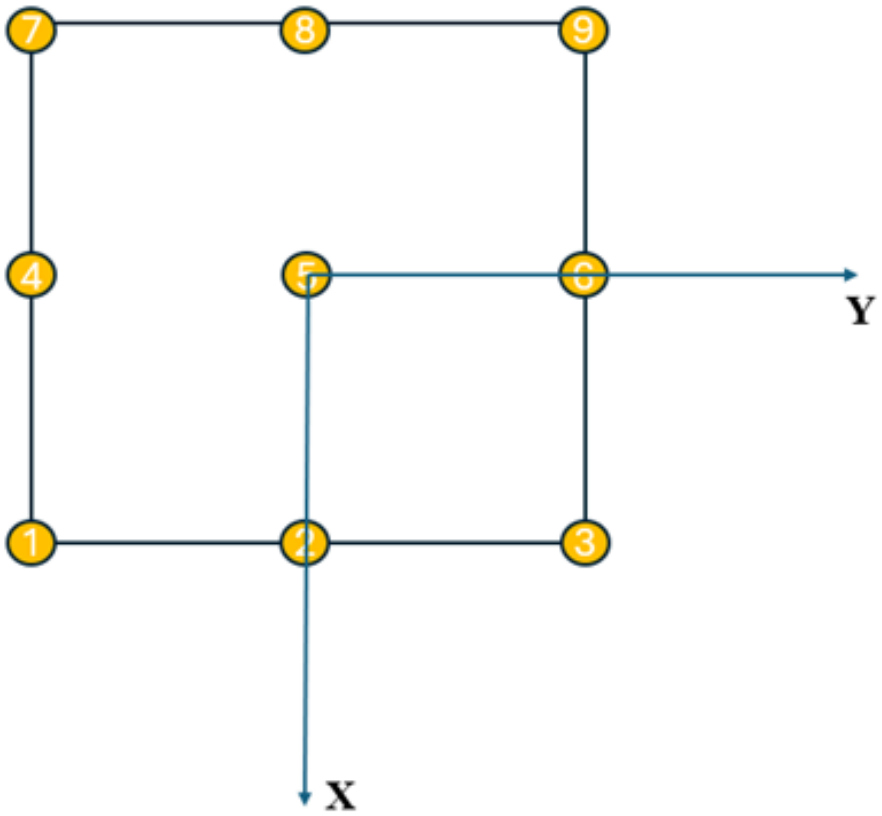


Figure 22: The top view of the workspace and the labels of detection points.

While H=300 mm, The heights of the first, second, third, and fourth layers are 300 mm, 400 mm, 500 mm, and 600 mm, respectively. As the second and third layers are situated in the middle, theoretically all detection points on these two layers are reachable. Therefore, the mean and variance of distances to these points can be calculated. Combined with a normal distribution whose $\mu = 0$, a threshold is set to be 99.7% of the data are within 3 standard deviations of the mean can be derived to determine whether detection points on the first and fourth layers are reachable. The threshold τ can be calculated in Equation 54.

$$\begin{aligned} \bar{d}_{2,3} &= \sum_{i=2}^{3}\sum_{j=1}^{9} d_{ij,min} \\ \sigma_{2,3} &= \sqrt{\frac{1}{n} \times \sum_{i=1}^{n}(d_i - \bar{d})^2} \\ \tau &= \bar{d}_{2,3} + 3 \times \sigma_{2,3} \end{aligned} \tag{54}$$

The segmentation of the effective workspace for other values of H, specifically H=240, 260, and 280 mm, can also be performed using the same approach. The numerical values corresponding to different H parameters are illustrated in Figures B.3, B.4, B.5 in Appendix B. The results and reachability for different H are presented in Table 3.

Table 3: The Reachability of Detection Points with Different H.

H (mm)	**Layer**	$\bar{d}_{2,3}$ **(mm)**	$\sigma_{2,3}$ **(mm)**	**Threshold** τ **(mm)**	**Reachability**
300	1	5.79	2.50	13.29	9
	4	5.79	2.50	13.29	3
280	1	4.87	2.02	10.94	9
	4	4.87	2.02	10.94	4
260	1	5.37	2.01	11.41	7
	4	5.37	2.01	11.41	7
240	1	5.35	2.48	12.78	8
	4	5.35	2.48	12.78	9

As shown in Table 3, setting H to 240 mm achieves better reachability, which is more reasonable compared to other scenarios. Additionally, to obtain a more accurate threshold, multiple tests or experimenting with a greater number of simulation indices can be undertaken to reduce random and systematic errors.

4.4 Inverse Kinematics

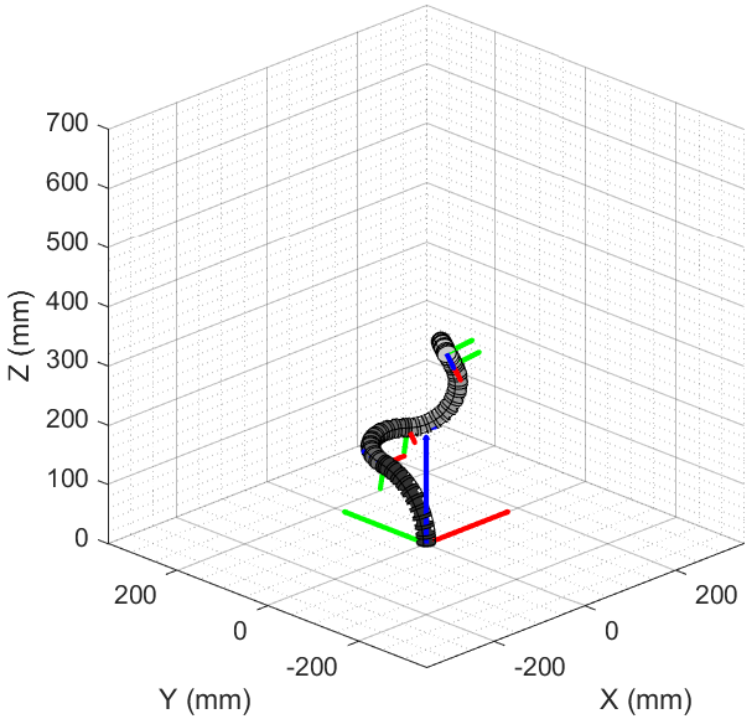
This section will demonstrate the posture solution of single modules and trajectory replication results obtained through the IK algorithm. An in-depth analysis of accuracy and computational speed will be conducted, along with a discussion of its advantages and limitations. Using angles as input for the programme testing was preferred due to the complexity involved in specifying 12 indices of $\mathbf{O}target$ and $\mathbf{P}target$.

4.4.1 Posture Solution of Single Module Bending

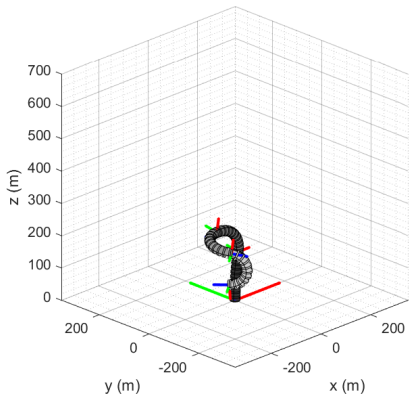
Firstly, the bending of individual modules was tested as the target for the IK solution. The IK algorithm was tested for different angles of bending for Modules 1, 2, 3, and 4,

and the specific results are presented in Table A.2 in Appendix A. The solutions of the Modules 3 and 4 were highly satisfactory, with Module 3 even converging to an error of 0.02 mm in 6 epochs. However, the results of the Modules 1 and 2 were unsatisfactory, which were caused by inappropriate initialization. According to the flowchart in Figure 10, the IK algorithm requires an initialization to start iterations. Take the example of the target with $\boldsymbol{\alpha} = [0,\ 90,\ 0,\ 0]^\circ$, after initializing with the initial posture in Figure 8 and applying the IK algorithm for iterations, the first iteration yielded a posture with $\boldsymbol{\theta} = [-0.0,\ 104.04,\ -0.0,\ -22.0]^\circ$. Moreover, since each iteration starts from Module 4, the error in Module 4 cannot be eliminated. To mitigate this effect, selecting a more suitable posture for initialization is crucial. If the posture used for initialization with $\boldsymbol{\alpha} = [0,\ 100,\ 0,\ 0]^\circ$, the solution of the IK algorithm would change to $\boldsymbol{\theta} = [-0.0,\ 81.77,\ -0.0,\ 9.3]^\circ$. Nevertheless, due to errors generated by the iteration sequence, complete elimination remains challenging, and efforts were make to minimize the error to the greatest extent possible.

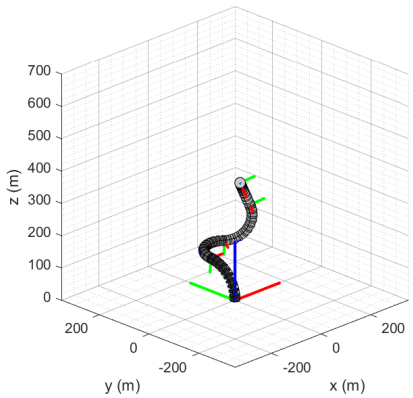
Afterwards, the complex bending angles of modules was utilized as the target to investigate the influences of initialization. The target with angles $\boldsymbol{\alpha} = [80,\ 120,\ -120,\ 90]^\circ$ was selected because its solution is complex, and it lies within the workspace of manipulator. The posture of the manipulator is shown in Figure 23a. This scenario is likely to occur in practical applications and necessitates resolution through relevant methods.



(a) target: $\boldsymbol{\alpha} = [80,\ 120,\ -120,\ 90]^\circ$



(b) initialization: $\boldsymbol{\alpha} = [0, 0, 0, 0]^\circ$;
$\boldsymbol{\theta} = [-8.88,\ -89.99,\ -126.81,\ -169.73]^\circ$



(c) initialization: $\boldsymbol{\alpha} = [85, 125, -115, 55]^\circ$;
$\boldsymbol{\theta} = [84.98,\ 122.93,\ -114.9,\ 63.01]^\circ$

Figure 23: The inverse kinematics algorithm with different initialization.

With the initialization using the initial posture, the solution was $\boldsymbol{\theta} = [-8.88, -89.99, -126.81, -169.73]^\circ$, which is shown in Figure 23b. However, the solution of IK algorithm was $\boldsymbol{\theta} = [84.98,\ 122.93,\ -114.9,\ 63.01]^\circ$, which is significantly close to the target while the angles for initialization were $[85,\ 125,\ -115,\ 55]^\circ$. The solution of the IK algorithm with more appropriate initialization is shown in Figure 23c. This signifies that if the manipulator is presently in a posture similar to the target configuration, it can accurately determine the corresponding bending angles $\boldsymbol{\theta}$ through

the IK algorithm. This proved to be highly beneficial for trajectory replication.
Ultimately, The efficient computational capability of algorithm is one of its strengths. The algorithm only took 1.905 seconds to complete 10,000 epochs updating. In practical applications, it requires only 200 epochs to determine convergence. In comparison to traditional methods like inverse Jacobian [61], which involve matrix transformations and derivatives, this approach provides potential solutions in a short time, addressing the issue of singular points. However, this method has its limitations. In scenarios with multiple segments, the algorithm may introduce significant errors due to the iteration order, and these errors were difficult to be eliminated. The only viable solution is to enhance the algorithm's performance through suitable initialization methods.

4.4.2 Trajectory Replication

The preceding discussion has highlighted genetic method, which is utilizing the current posture for initialization in trajectory replication. In this phase, a comparison of different initializations aimed to highlight the benefits of incorporating genetic method in trajectory replication and planning for the IK algorithm. Subsequently, the genetic method and IK algorithm are employed to replicate two types of trajectories: arc segments and closed paths. This section will analyze the genetic method and these two trajectories, elucidating the strengths and weaknesses of inverse kinematics algorithm.
The blue trajectory in Figure 24 is the target trajectory, which bending from $\boldsymbol{\alpha} = [0,\ 0,\ 0,\ 0]$ to $\boldsymbol{\alpha} = [20,\ 20,\ 20,\ 20]$ equally in 20 steps. The red and purple trajectories are the replications using consistent initialization method and the genetic initialization method, respectively. The trajectory replication using the genetic method was comparatively better aligned with the target. The corrective effectiveness of the genetic algorithm was validated for other targets. The data of trajectories in Figure 24 is listed in Table A.3.

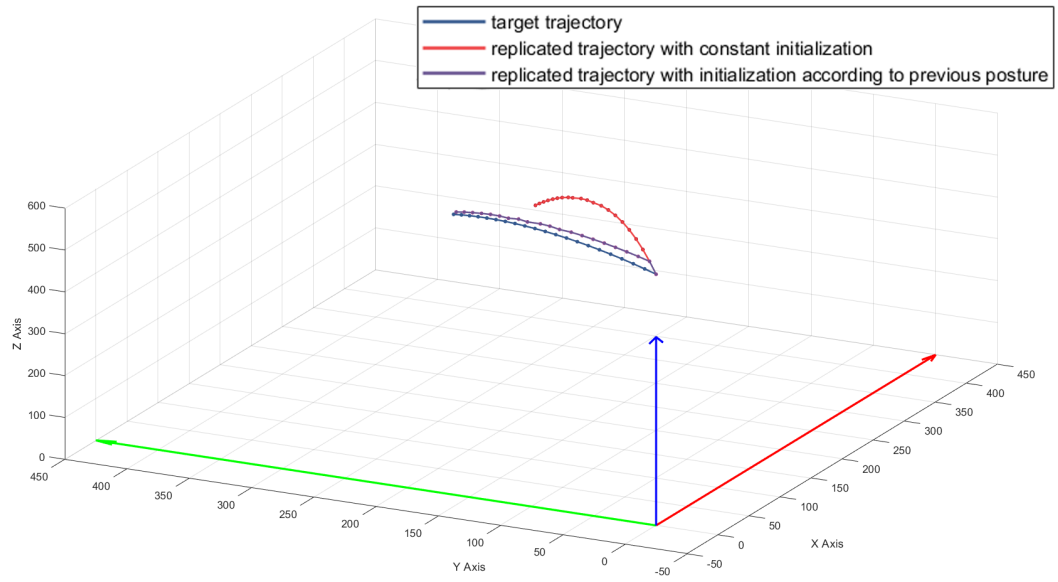


Figure 24: The trajectory replications with and without genetic method.

The replicated trajectories of both target trajectories closely align, but there exists a certain degree of error. This discrepancy was attributed to the inclusion of the error threshold ϵ set at 0.02 in the programme, allowing for computational efficiency while maintaining accuracy. However, when dealing with significantly larger angular deviations, instances may arise where the replicated trajectories failed to precisely match the target trajectories. This situation was overlooked as it fell beyond the defined workspace. The average errors about the trajectory replication were calculated in Table 4.

Table 4: The Average Errors of the replicated trajectories.

Replicated Trajectory	$e_{average}$ **(mm)**	$e_{average}$ **(mm)**
total	3.576157	6.415941
in workspace	1.032255	2.642474

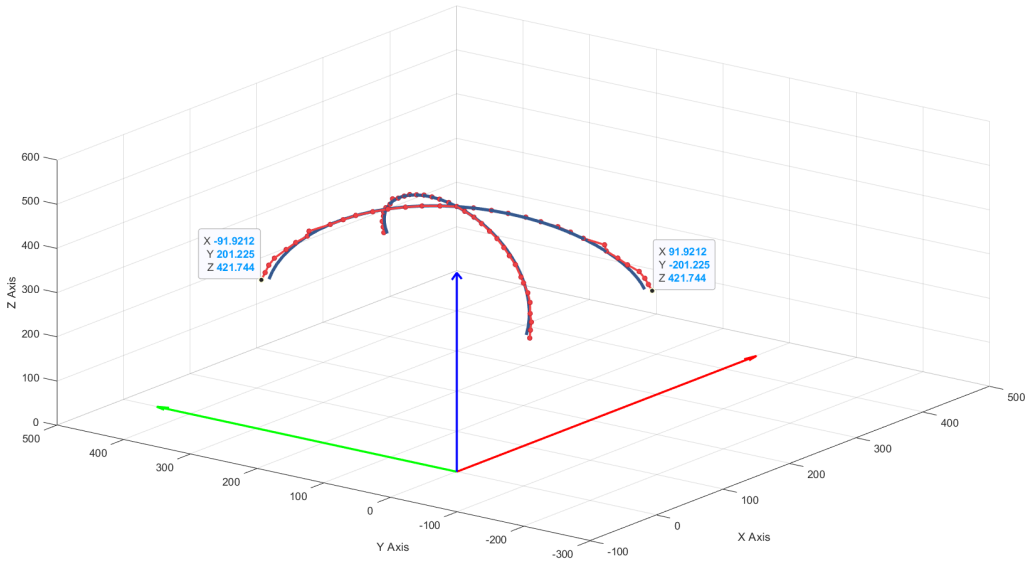


Figure 25: The cross-shaped trajectory and its replication.

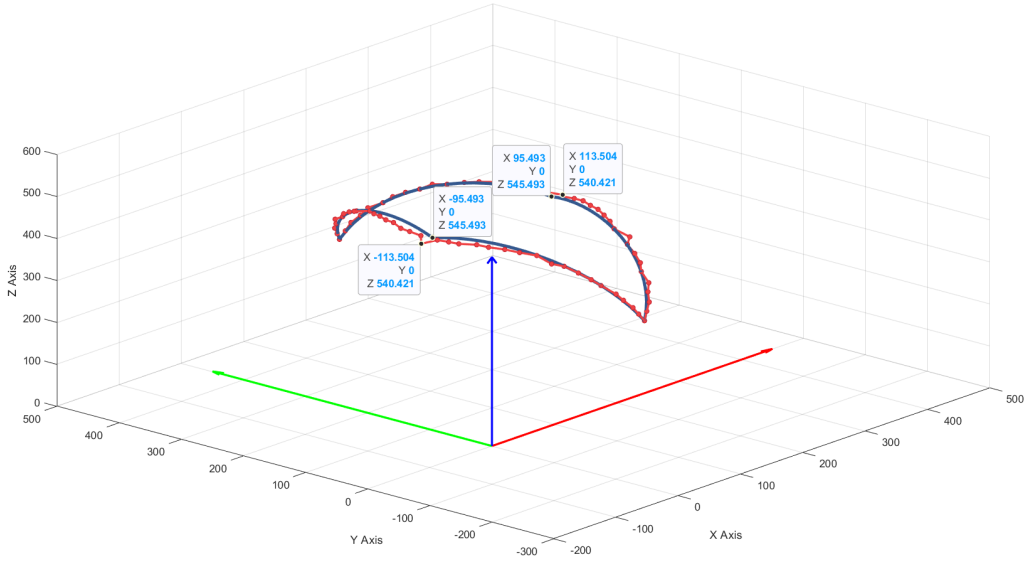


Figure 26: The closed trajectory and its replication.

4.4.3 Limitation and Improvement

However, it is challenging to calculate certain postures with IK algorithm. A method has been proposed involving continuous posture updating through a factor f_u. Nevertheless, this approach lacks systematic validation and is currently feasible only in specific circumstances. Therefore, it is mentioned here solely as a potential solution.

In practical applications, the initialization can be started with a relatively similar posture. In this context, the target where $\boldsymbol{\theta_{target}}$ = [45, 45, -45, -45] ° and the initialization with $\boldsymbol{\alpha_0}$ = [20, 20, -20, -20] ° were utilized to exemplify the corresponding calculations. The ***target*** is the position and orientation of the manipulator end effector.

The current posture with $\boldsymbol{\alpha_0}$ = [20, 20, -20, -20] ° was used as an initialization for calculations through inverse kinematics algorithm, resulting in the solution $\boldsymbol{\theta_1}$. The difference between $\boldsymbol{\alpha_0}$ and $\boldsymbol{\theta_1}$ was calculated by $f_u = 0.5$ to acquire $\boldsymbol{\alpha_1}$, which is shown in Equation 55.

$$\begin{aligned}\boldsymbol{\theta_1} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_0}) = [42.38, 73.25, -69.59, -62.85]^\circ \\ \boldsymbol{\alpha_1} &= \boldsymbol{\alpha_0} + f_u \cdot (\boldsymbol{\theta_1} - \boldsymbol{\alpha_0}) = [31.19, 46.63, -34.80, -41.43]^\circ \end{aligned} \tag{55}$$

The $\boldsymbol{\alpha_i}$ can be continuously updated according to the method, which the results yield in Equations 56,57, 58, 59, and 60.

$$\begin{aligned}\boldsymbol{\theta_2} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_1}) = [56.95, 34.48, -56.40, -38.96]^\circ \\ \boldsymbol{\alpha_2} &= \boldsymbol{\alpha_1} + f_u \cdot (\boldsymbol{\theta_2} - \boldsymbol{\alpha_1}) = [44.07, 40.56, -45.60, -40.20]^\circ \end{aligned} \tag{56}$$

$$\begin{aligned}\boldsymbol{\theta_3} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_2}) = [43.64, 51.49, -45.97, -48.84]^\circ \\ \boldsymbol{\alpha_3} &= \boldsymbol{\alpha_2} + f_u \cdot (\boldsymbol{\theta_3} - \boldsymbol{\alpha_2}) = [43.56, 46.03, -45.79, -44.52]^\circ \end{aligned} \tag{57}$$

$$\begin{aligned}\boldsymbol{\theta_4} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_3}) = [46.22, 42.71, -45.9, -43.88]^\circ \\ \boldsymbol{\alpha_4} &= \boldsymbol{\alpha_3} + f_u \cdot (\boldsymbol{\theta_4} - \boldsymbol{\alpha_3}) = [44.89, 44.37, -45.85, -44.20]^\circ \end{aligned} \tag{58}$$

$$\begin{aligned}\boldsymbol{\theta_5} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_4}) = [44.87, 45.23, -45.03, -45.30]^\circ \\ \boldsymbol{\alpha_5} &= \boldsymbol{\alpha_4} + f_u \cdot (\boldsymbol{\theta_5} - \boldsymbol{\alpha_4}) = [44.88, 44.80, -45.85, -44.20]^\circ \end{aligned} \tag{59}$$

$$\begin{aligned}\boldsymbol{\theta_6} &= \mathbf{FABRIKc}(\boldsymbol{target}, \boldsymbol{\alpha_5}) = [45.08, 44.91, -45.09, -44.96]^\circ \\ \boldsymbol{\alpha_6} &= \boldsymbol{\alpha_5} + f_u \cdot (\boldsymbol{\theta_6} - \boldsymbol{\alpha_5}) = [44.94, 44.86, -45.47, -44.58]^\circ \end{aligned} \tag{60}$$

In addition, another approach involves manually adjusting the manipulator's posture through observation, which is relatively straightforward but lacks precision. It is important to note that due to orientation constraints, the sum of angles for modules 1 and 3 and modules 2 and 4 should remain constant during the adjustment process.

4.5 Electronic Control

4.5.1 Actuation Control

Initially, the testing of the results was supposed to be carried out through Proteus simulation. However, it was discovered that during the simulation process in Proteus, the motor components did not function normally, with frequent occurrences of stepping errors. Meanwhile, the simulation could not be completed in real-time due to the heavy load on the computer CPU. After consideration, the team members purchased a complete set of real components for testing. In the real circuit, the pin that each motor occupies are listed in Table 5 to reduce the complexity of wiring.

Table 5: The Pin Assignment of Stepper Motors.

Stepper Motor	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Pin	22	30	38	46	23	31	39	47
Pin	24	32	40	48	25	33	41	49
Pin	26	34	42	50	27	35	43	51
Pin	28	36	44	52	29	37	45	53

After connecting the Arduino to the stepper motors as depicted in Figure 27, the programme is ready for execution.

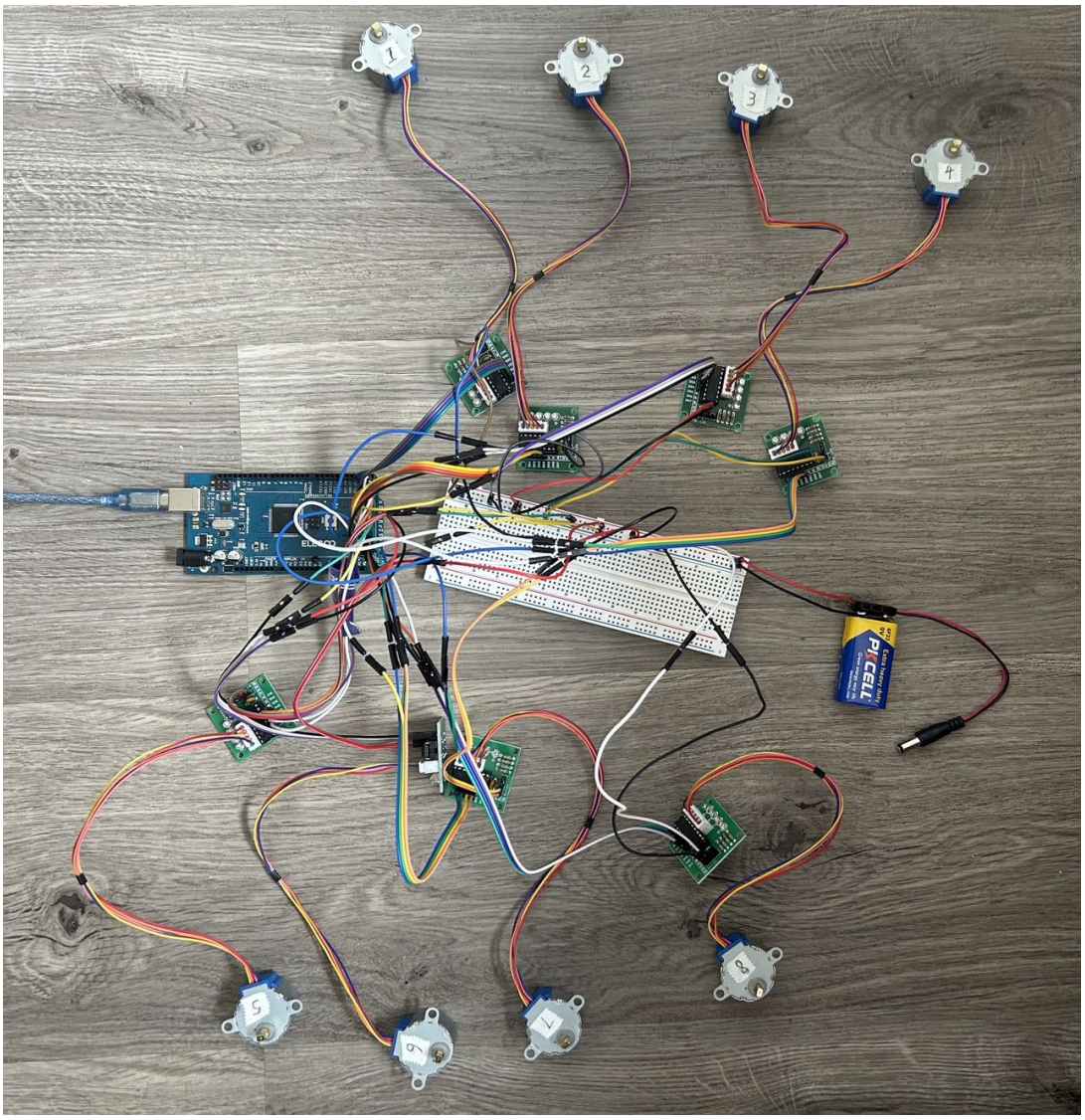


Figure 27: The circuit connection of Arduino and stepper motors.

Users are required to sequentially input eight values of $\Delta S_1 \sim \Delta S_8$ to initiate the corresponding operation of the stepper motor. If the input is less than eight, the control system will not be executed.

According to Figure B.2 in Appendix B, the change volume of cables are input into the Arduino program for execution, with the input data series being $\boldsymbol{\Delta S}$ = [0, 0, -8, 8, -15, 15, 65, -65] (mm). The value of ΔS_i is always in pairs, e.g, two cables controlling the same unit should always have the opposite ΔS_i value, this is decided by the working principle of the manipulator. After conversion, the corresponding $Step_1 \sim Step_8$ is [0, 0, -521, 521, -978, 978, 4237, -4237] (steps), where the "-" sign represents counterclockwise steps. Afterwards, the stepping can be finished in certain time. The function *Serial.print(motor.currentPosition())* can be utilized to check the steps of motors.

According to Figure B.1a in Appendix B, result of first testing is demonstrated, with all the motors are working appropriately. In this programme, all floating-point precision is taken to the 10^{-4} millimeter while the result printed on the monitor is 10^{-2} millimeter, which means that the accuracy of the result is acceptable.

Then, reset the motors, the motors went back to initial condition, which is $Step_1 \sim Step_8 = 0$. Repeat the motor stepping process, but this time, the motors are starting from a previous location from $\boldsymbol{\Delta S_{now}}$ = [0, 0, -8, 8, -15, 15, 65, -65] (mm) to $\boldsymbol{\Delta S_{next}}$ = [33, -33, 16, -16, 0, 0, 10, -10] (mm). The difference $\boldsymbol{\Delta S_{diff}}$ = [33, -33, 24, -24, 15, -15, -55, 55] (mm), while the steps for motors are [2151, -2151, 1565, -1565, 978, -978, -3585, 3585] (steps). The result can be verified by checking how many revolutions each motor rotated in Figure B.1b in Appendix B. The results are aligned with the calculation. Hence, the actuation control system runs correctly. The Arduino programme is updated in GitHub repository in Arduino Simulation.

Overall, the method of using Arduino to control the stepper motor and drive the manipulator is a feasible solution. Although the initial idea of using simulation software failed to yield satisfying results, the testing outcomes with real components solved this problem.

Additionally, from the perspective of test results, it can be observed that the precision of the stepper motors are very high, which is a great advantage for an open-loop (no feedback) system as it minimizes errors to the greatest extent. In the 28YBJ-48 motor parameter settings, full stepping configuration implies that the motor is divided into 2048 steps per revolution. For a rotor driving a cable with a circumference of 31.416 mm, this means a resolution of 0.01534 mm per step, which is enough to provide the accuracy required.

However, at the same time, there are some limitations to consider in this aspect. The most apparent limitation is that when cables wrap around the rotor, it effectively increases the diameter of the rotor, leading to some errors. For a cable with a total length of 68 mm, it can wrap around the rotor twice. Assuming the cable has a thickness of 1 mm, this would introduce a maximum error of approximately $31.416/(31.416 + 2) = 0.05$. Although in practice, it is unlikely that two layers of cable would perfectly stack on top of each other, this should be considered, because it's a factor that will cause errors. To solve this, an equation relating the length change of

cables and the change in diameters should be deduced and applied in the program to minimize the potential error, but due to limited time, this part was not finished.

Furthermore, in this programme, only a rough setting for the step speed of the motors has been set. However, for achieving synchronization in starting/ending movement of each unit of the manipulator, more precise mathematical derivations for the speed of motors are required. In the following development process, it would be helpful to associate the speed of stepper motors with the distance to the target position, thereby dynamically setting a speed value for each motor to accomplish this synchronization.

Ultimately, the input parameters of the control system are $\boldsymbol{\Delta S}$, which consist of eight indices. It is inconvenient for users to input a series of indices. The Arduino programme can be optimized to take the angles $\boldsymbol{\theta}$ as input parameters, which only require the users to input four parameters. The Python version programme is uploaded in the GitHub and ready to be converted into Arduino programme.

4.5.2 End Effector Information Acquisition and Display Yuehan

5 Conclusion

This is the Conclusion of the final report.

References

[1] Sarthak Misra, K. T. Ramesh, and Allison M. Okamura. Modeling of tool-tissue interactions for computer-based surgical simulation: A literature review. *Presence*, 17(5):463–491, 2008.

[2] Jessica Burgner-Kahrs, D. Caleb Rucker, and Howie Choset. Continuum robots for medical applications: A survey. *IEEE Transactions on Robotics*, 31(6):1261–1280, 2015.

[3] Pan Zhou, Jiantao Yao, Shuai Zhang, Chunjie Wei, Hongyu Zhang, and Shupeng Qi. A bioinspired fishbone continuum robot with rigid-flexible-soft coupling structure. *Bioinspiration & Biomimetics*, 17(6):066012, 2022.

[4] Jonathan M Sackier, Chuck Wooters, Lisa Jacobs, Amy Halverson, Darrin Uecker, and Yulun Wang. Voice activation of a surgical robotic assistant. *The American Journal of Surgery*, 174(4):406–409, 1997.

[5] Satyam Kalan, Sanket Chauhan, Rafael F Coelho, Marcelo A Orvieto, Ignacio R Camacho, Kenneth J Palmer, and Vipul R Patel. History of robotic surgery. *Journal of Robotic Surgery*, 4:141–147, 2010.

[6] TOMMASO FALCONE, JEFFREY GOLDBERG, Antonio Garcia-Ruiz, HAROUT MARGOSSIAN, and LAUREL STEVENS. Full robotic assistance for laparoscopic tubal anastomosis: a case report. *Journal of Laparoendoscopic & Advanced Surgical Techniques*, 9(1):107–113, 1999.

[7] Hermann Reichenspurner, Ralph J Damiano, Michael Mack, Dieter H Boehm, Helmut Gulbins, Christian Detter, Bruno Meiser, Reinhard Ellgass, and Bruno Reichart. Use of the voice-controlled and computer-assisted surgical system zeus

for endoscopic coronary artery bypass grafting. *The Journal of thoracic and cardiovascular surgery*, 118(1):11–16, 1999.

[8] Bertrand Guillonneau, Olivier Cappèle, Juan Bosco Martinez, Stèphane Navarra, and Guy Vallancien. Robotic assisted, laparoscopic pelvic lymph node dissection in humans. *The journal of urology*, 165(4):1078–1081, 2001.

[9] F. Pugin, P. Bucher, and P. Morel. History of robotic surgery : From aesop® and zeus® to da vinci®. *Journal of Visceral Surgery*, 148(5, Supplement):e3–e8, 2011. Robotic surgery.

[10] Chih Yu An, Jia Hao Syu, Ching Shiow Tseng, Chih-Ju Chang, et al. An ultrasound imaging-guided robotic hifu ablation experimental system and accuracy evaluations. *Applied Bionics and Biomechanics*, 2017, 2017.

[11] Sunita Chauhan, Hagey Amir, Guang Chen, Axel Hacker, Maurice Stephan Michel, and Kai Uwe Koehrmann. Intra-operative feedback and dynamic compensation for image-guided robotic focal ultrasound surgery. *Computer Aided Surgery*, 13(6):353–368, 2008.

[12] Sunita Chauhan. A mechatronic system for non invasive treatment of the breast cancers. In *Mechatronics and Machine Vision: John Billingsly (Eds)*. Research Studies Press Ltd, 2002.

[13] Sunita Chauhan, Ming Yeong Teo, and Wendy Teo. Robotic system for ablation of deep-seated skull base cancers—a feasibility study. In *Proceedings of the 34th International MATADOR Conference: Formerly The International Machine Tool Design and Conferences*, pages 21–27. Springer, 2004.

[14] Mohamed K Almekkaway, Islam A Shehata, and Emad S Ebbini. Anatomical-based model for simulation of hifu-induced lesions in atherosclerotic plaques. *International Journal of Hyperthermia*, 31(4):433–442, 2015.

[15] Laure-Anaïs Chanel, Florent Nageotte, Jonathan Vappou, Jianwen Luo, Loïc Cuvillon, and Michel de Mathelin. Robotized high intensity focused ultrasound (hifu) system for treatment of mobile organs using motion tracking by ultrasound imaging: An in vitro study. In *2015 37th Annual International Conference of*

the IEEE Engineering in Medicine and Biology Society (EMBC), pages 2571–2575. IEEE, 2015.

[16] Andrea Cafarelli, Marco Mura, Alessandro Diodato, Andrea Schiappacasse, Matteo Santoro, Gastone Ciuti, and Arianna Menciassi. A computer-assisted robotic platform for focused ultrasound surgery: Assessment of high intensity focused ultrasound delivery. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 1311–1314. IEEE, 2015.

[17] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, 2020.

[18] Michael E Moran. Evolution of robotic arms. *Journal of robotic surgery*, 1(2):103–111, 2007.

[19] Mark E Rosheim. *Robot evolution: the development of anthrobotics*. John Wiley & Sons, 1994.

[20] Bhaskar Dasgupta and TS1739334 Mruthyunjaya. The stewart platform manipulator: a review. *Mechanism and machine theory*, 35(1):15–40, 2000.

[21] Ray L Page. Brief history of flight simulation. *SimTecT 2000 proceedings*, pages 11–17, 2000.

[22] Roger Bostelman, James Albus, Nicholas Dagalakis, Adam Jacoff, John Gross, et al. Applications of the nist robocrane. In *Proceedings of the 5th International Symposium on Robotics and Manufacturing*, volume 5, page 1, 1994.

[23] Anastasia Puzatova, Pshtiwan Shakor, Vittoria Laghi, and Maria Dmitrieva. Large-scale 3d printing for construction application by means of robotic arm and gantry 3d printer: A review. *Buildings*, 12(11):2023, 2022.

[24] Byung-In Kim, Sunderesh S Heragu, Robert J Graves, and Art St Onge. Clustering-based order-picking sequence algorithm for an automated warehouse. *International Journal of Production Research*, 41(15):3445–3460, 2003.

[25] Ishak WI Wan, WH Kit, and MA Awal. Design and development of eggplant harvester for gantry system. *Pertanika Journal of Science & Technology*, 18(2), 2010.

[26] Eberhard Abele, Matthias Weigold, and Stefan Rothenbücher. Modeling and identification of an industrial robot for machining applications. *CIRP annals*, 56(1):387–390, 2007.

[27] Shin-ichi Matsuoka, Kazunori Shimizu, Nobuyuki Yamazaki, and Yoshinari Oki. High-speed end milling of an articulated robot and its characteristics. *Journal of materials processing technology*, 95(1-3):83–89, 1999.

[28] Hoai Nam Huynh, Hamed Assadi, Edouard Rivière-Lorphèvre, Olivier Verlinden, and Keivan Ahmadi. Modelling the dynamics of industrial robots for milling operations. *Robotics and Computer-Integrated Manufacturing*, 61:101852, 2020.

[29] See Han Tay, Wai Heng Choong, and Hou Pin Yoong. A review of scara robot control system. In *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, pages 1–6, 2022.

[30] Rishabh Chaturvedi, Anas Islam, and Kamal Sharma. Anticipated investigation of a cylindrical robot arm by means of compound materials. *Eur J Mol Clin Med*, 7(4):736–745, 2020.

[31] Foster Collins and Mark Yim. Design of a spherical robot arm with the spiral zipper prismatic joint. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 2137–2143. IEEE, 2016.

[32] Varnita Verma, A Gupta, MK Gupta, and P Chauhan. Performance estimation of computed torque control for surgical robot application. *Journal of Mechanical Engineering and Sciences*, 14(3):7017–7028, 2020.

[33] Erhan Akdoğan and Mehmet Arif Adli. The design and control of a therapeutic exercise robot for lower limb rehabilitation: Physiotherabot. *Mechatronics*, 21(3):509–522, 2011.

[34] Lu Zongxing, Li Wanxin, and Zhang Liping. Research development of soft manipulator: A review. *Advances in Mechanical Engineering*, 12(8):1687814020950094, 2020.

[35] Deepak Trivedi, Christopher D Rahn, William M Kier, and Ian D Walker. Soft robotics: Biological inspiration, state of the art, and future research. *Applied bionics and biomechanics*, 5(3):99–117, 2008.

[36] Cecilia Laschi, Barbara Mazzolai, Virgilio Mattoli, Matteo Cianchetti, and Paolo Dario. Design of a biomimetic robotic octopus arm. *Bioinspiration & biomimetics*, 4(1):015006, 2009.

[37] Michael W Hannan and Ian D Walker. Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots. *Journal of robotic systems*, 20(2):45–63, 2003.

[38] Cecilia Laschi, Matteo Cianchetti, Barbara Mazzolai, Laura Margheri, Maurizio Follador, and Paolo Dario. Soft robot arm inspired by the octopus. *Advanced robotics*, 26(7):709–727, 2012.

[39] Fionnuala Connolly, Conor J Walsh, and Katia Bertoldi. Automatic design of fiber-reinforced soft actuators for trajectory matching. *Proceedings of the National Academy of Sciences*, 114(1):51–56, 2017.

[40] Shujun Wei, Tianyu Wang, and Guoving Gu. Design of a soft pneumatic robotic gripper based on fiber-reinforced actuator. *Journal of Mechanical Engineering*, 53(13):29–38, 2017.

[41] Bobak Mosadegh, Panagiotis Polygerinos, Christoph Keplinger, Sophia Wennstedt, Robert F Shepherd, Unmukt Gupta, Jongmin Shim, Katia Bertoldi, Conor J Walsh, and George M Whitesides. Pneumatic networks for soft robotics that actuate rapidly. *Advanced functional materials*, 24(15):2163–2170, 2014.

[42] Ramses V Martinez, Jamie L Branch, Carina R Fish, Lihua Jin, Robert F Shepherd, Rui Nunes, Zhigang Suo, and George McClelland Whitesides. Robotic tentacles with three-dimensional mobility based on flexible elastomers. *Advanced materials*, 2013.

[43] Run Wang, Nan Jiang, Jian Su, Qu Yin, Yue Zhang, Zhongsheng Liu, Haibao Lin, Francisco A Moura, Ningyi Yuan, Siegmar Roth, et al. A bi-sheath fiber sensor for giant tensile and torsional displacements. *Advanced Functional Materials*, 27(35):1702134, 2017.

[44] Eric Brown, Nicholas Rodenberg, John Amend, Annan Mozeika, Erik Steltz, Mitchell R Zakin, Hod Lipson, and Heinrich M Jaeger. Universal robotic gripper based on the jamming of granular material. *Proceedings of the National Academy of Sciences*, 107(44):18809–18814, 2010.

[45] Caleb Christianson, Nathaniel N Goldberg, Dimitri D Deheyn, Shengqiang Cai, and Michael T Tolley. Translucent soft robots driven by frameless fluid electrode dielectric elastomer actuators. *Science Robotics*, 3(17):eaat1893, 2018.

[46] QM Zhang, Hengfeng Li, Martin Poh, Feng Xia, Z-Y Cheng, Haisheng Xu, and Cheng Huang. An all-organic composite actuator material with a high dielectric constant. *Nature*, 419(6904):284–287, 2002.

[47] Bertrand Tondu, Serge Ippolito, Jérémie Guiochet, and Alain Daidie. A seven-degrees-of-freedom robot-arm driven by pneumatic artificial muscles for humanoid robots. *The International Journal of Robotics Research*, 24(4):257–274, 2005.

[48] David Jakes, Zongyuan Ge, and Liao Wu. Model-less active compliance for continuum robots using recurrent neural networks. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2167–2173. IEEE, 2019.

[49] Ernar Amanov, Thien-Dang Nguyen, and Jessica Burgner-Kahrs. Tendon-driven continuum robots with extensible sections—a model-based evaluation of path-following motions. *The International Journal of Robotics Research*, 40(1):7–23, 2021.

[50] Ammar Amouri, Abdelhakim Cherfia, Ayman Belkhiri, and Halim Merabti. Bio-inspired a novel dual-cross-module sections cable-driven continuum robot: design, kinematics modeling and workspace analysis. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 45(5):265, 2023.

[51] Tomer Anor, Joseph R. Madsen, and Pierre Dupont. Algorithms for design of continuum robots using the concentric tubes approach: A neurosurgical example.

In *2011 IEEE International Conference on Robotics and Automation*, pages 667–673, 2011.

[52] Jorge G Cham, Sean A Bailey, Jonathan E Clark, Robert J Full, and Mark R Cutkosky. Fast and robust: Hexapedal robots via shape deposition manufacturing. *The International Journal of Robotics Research*, 21(10-11):869–882, 2002.

[53] Andrew D Marchese, Robert K Katzschmann, and Daniela Rus. A recipe for soft fluidic elastomer robots. *Soft robotics*, 2(1):7–25, 2015.

[54] Haili Li, Jiantao Yao, Pan Zhou, Xinbo Chen, Yundou Xu, and Yongsheng Zhao. High-force soft pneumatic actuators based on novel casting method for robotic applications. *Sensors and Actuators A: Physical*, 306:111957, 2020.

[55] Ali Zolfagharian, Abbas Z Kouzani, Sui Yang Khoo, Amir Ali Amiri Moghadam, Ian Gibson, and Akif Kaynak. Evolution of 3d printed soft actuators. *Sensors and Actuators A: Physical*, 250:258–272, 2016.

[56] Cosima Du Pasquier, Tian Chen, Skylar Tibbits, and Kristina Shea. Design and computational modeling of a 3d printed pneumatic toolkit for soft robotics. *Soft robotics*, 6(5):657–663, 2019.

[57] Ammar Amouri, Abdelhakim Cherfia, Ayman Belkhiri, and Halim Merabti. Bio-inspired a novel dual-cross-module sections cable-driven continuum robot: design, kinematics modeling and workspace analysis. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, 45(5):265, 2023.

[58] Andreas Aristidou and Joan Lasenby. Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models*, 73(5):243–260, 2011.

[59] Weihao Zhang, Zhixiong Yang, Tianlai Dong, and Kai Xu. Fabrikc: an efficient iterative inverse kinematics solver for continuum robots. In *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 346–352, 2018.

[60] Arduino comparison guide - sparkfun learn. (Accessed on 03 March 2024).

[61] Isuru S Godage, Emanuele Guglielmino, David T Branson, Gustavo A Medrano-Cerda, and Darwin G Caldwell. Novel modal approach for kinematics of multisection continuum arms. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1093–1098. IEEE, 2011.

Appendix A

The Appendix of Tables

Table A.1: The results of FABRIKc algorithm with target angle $\alpha = [0\ 0\ 90\ 0]$.

Epoch	θ_1 (°)	θ_2 (°)	θ_3 (°)	θ_4 (°)	**Error (mm)**
1	6.71	-0.0	83.29	-0.0	33.77506
2	6.77	-0.0	83.23	-0.0	34.07841
3	6.50	-0.0	83.50	-0.0	32.69662
4	6.37	-0.0	83.63	-0.0	32.07427
5	6.32	-0.0	83.68	-0.0	31.79337
6	6.29	-0.0	83.71	-0.0	31.66917
7	5.94	-0.0	84.06	-0.0	29.88369
8	5.74	-0.0	84.26	-0.0	28.91057
9	5.66	-0.0	84.34	-0.0	28.4723
10	5.62	-0.0	84.38	-0.0	28.2791
11	5.6	-0.0	84.4	-0.0	28.19218
12	5.25	-0.0	84.75	-0.0	26.43303
13	5.1	-0.0	84.9	-0.0	25.64898
14	5.03	-0.0	84.97	-0.0	25.29275
15	5.0	-0.0	85.0	-0.0	25.13415
16	4.98	-0.0	85.02	-0.0	25.06414
17	4.63	-0.0	85.37	-0.0	23.31238
18	4.48	-0.0	85.52	-0.0	22.52919
19	4.41	-0.0	85.59	-0.0	22.17736

Continued on next page

Table A.1 – continued from previous page

Epoch	θ_1 (°)	θ_2 (°)	θ_3 (°)	θ_4 (°)	Error (mm)
20	4.38	-0.0	85.62	-0.0	22.01754
21	4.36	-0.0	85.64	-0.0	21.94852
22	4.01	-0.0	85.99	-0.0	20.19744
23	3.84	-0.0	86.16	-0.0	19.29379
24	3.76	-0.0	86.24	-0.0	18.8911
25	3.72	-0.0	86.28	-0.0	18.71167
26	3.7	-0.0	86.3	-0.0	18.63148
27	3.7	-0.0	86.3	-0.0	18.59747
28	3.35	-0.0	86.65	-0.0	16.86972
29	3.2	-0.0	86.8	-0.0	16.09738
30	3.13	-0.0	86.87	-0.0	15.7551
31	3.1	-0.0	86.9	-0.0	15.59885
32	3.09	-0.0	86.91	-0.0	15.53487
33	3.08	-0.0	86.92	-0.0	15.50114
34	3.08	-0.0	86.92	-0.0	15.48938
35	2.74	-0.0	87.26	-0.0	13.77029
36	2.59	-0.0	87.41	-0.0	13.00123
37	2.52	-0.0	87.48	-0.0	12.65954
38	2.47	-0.0	87.53	-0.0	12.43367
39	2.45	-0.0	87.55	-0.0	12.33092
40	2.44	-0.0	87.56	-0.0	12.28876
41	2.44	-0.0	87.56	-0.0	12.26754
42	2.44	-0.0	87.56	-0.0	12.25842
43	2.1	-0.0	87.9	-0.0	10.55236
44	1.95	-0.0	88.05	-0.0	9.79285
45	1.88	-0.0	88.12	-0.0	9.45857
46	1.85	-0.0	88.15	-0.0	9.3098
47	1.84	-0.0	88.16	-0.0	9.24442
48	1.83	-0.0	88.17	-0.0	9.21438
49	1.83	-0.0	88.17	-0.0	9.19955
50	1.83	-0.0	88.17	-0.0	9.19355

Continued on next page

Table A.1 – continued from previous page

Epoch	θ_1 (°)	θ_2 (°)	θ_3 (°)	θ_4 (°)	**Error (mm)**
51	1.49	-0.0	88.51	-0.0	7.48619
52	1.34	-0.0	88.66	-0.0	6.73267
53	1.27	-0.0	88.73	-0.0	6.39321
54	1.24	-0.0	88.76	-0.0	6.24293
55	1.23	-0.0	88.77	-0.0	6.17847
56	1.22	-0.0	88.78	-0.0	6.14882
57	1.22	-0.0	88.78	-0.0	6.13682
58	1.22	-0.0	88.78	-0.0	6.1309
59	1.22	-0.0	88.78	-0.0	6.12797
60	1.22	-0.0	88.78	-0.0	6.12521
61	0.88	-0.0	89.12	-0.0	4.42272
62	0.73	-0.0	89.27	-0.0	3.66632
63	0.66	-0.0	89.34	-0.0	3.31015
64	0.63	-0.0	89.37	-0.0	3.1559
65	0.61	-0.0	89.39	-0.0	3.08583
66	0.61	-0.0	89.39	-0.0	3.05386
67	0.61	-0.0	89.39	-0.0	3.03937
68	0.6	-0.0	89.4	-0.0	3.03352
69	0.6	-0.0	89.4	-0.0	3.03063
70	0.6	-0.0	89.4	-0.0	3.0305
71	0.6	-0.0	89.4	-0.0	3.03044
72	0.6	-0.0	89.4	-0.0	3.03041
73	0.6	-0.0	89.4	-0.0	3.0304
74	0.27	-0.0	89.73	-0.0	1.33635
75	0.12	-0.0	89.88	-0.0	0.59092
76	0.05	-0.0	89.95	-0.0	0.25938
77	0.02	-0.0	89.98	-0.0	0.11515
78	0.01	-0.0	89.99	-0.0	0.05169
79	0.0	-0.0	90.0	-0.0	0.02296

Table A.2: The Singular Posture Solution by FABRIKc start with initial posture.

α_1	α_2	α_3	α_4	**Converge Epoch**	θ_1	θ_2	θ_3	θ_4	**Error**
0	0	0	90	None	-0.0	3.34	-0.0	89.09	18.6954
0	0	0	60	None	-0.0	1.61	-0.0	59.58	9.5398
0	0	0	30	None	-0.0	0.49	-0.0	29.67	2.77844
0	90	0	0	79	0.0	-0.0	90.0	-0.0	0.02296
0	60	0	0	36	0.01	-0.0	59.99	-0.0	0.03408
0	30	0	0	6	0.01	-0.0	29.99	-0.0	0.02928
0	0	90	0	None	-0.0	89.84	-0.0	0.28	0.66891
0	0	60	0	None	-0.0	76.97	-0.0	-22.73	79.01594
0	0	30	0	None	-0.0	52.28	-0.0	-22.87	114.34898
90	0	0	0	None	106.26	-0.0	-16.26	-0.0	80.98985
60	0	0	0	None	82.04	-0.0	-22.04	-0.0	111.99102
30	0	0	0	None	52.72	-0.0	-22.72	-0.0	117.37369

Table A.3: The Singular Posture Solution by FABRIKc start with initial posture.

Step	**Target**			**Replication**			**Replication (Genetic)**		
0	0.00	600.00	0.00	0.00	600.00	0.00	0.00	0.00	600.00
1	7.85	13.09	599.74	26.09	18.29	598.85	26.09	18.29	598.85
2	15.70	26.15	598.96	51.83	36.48	595.42	33.92	31.32	597.63
3	23.53	39.18	597.67	76.79	54.33	589.87	41.60	44.30	595.92
4	31.34	52.15	595.86	100.65	71.73	582.38	49.32	57.20	593.68
5	39.12	65.04	593.55	123.25	88.55	573.19	56.99	70.09	590.94
6	46.87	77.83	590.73	144.35	104.70	562.58	64.50	82.78	587.73
7	54.57	90.50	587.41	163.86	120.13	550.84	72.01	95.39	584.01
8	62.22	103.03	583.59	181.74	134.70	538.26	79.41	107.80	579.82
9	69.82	115.41	579.28	196.54	148.46	525.92	85.14	120.05	575.50
10	77.35	127.61	574.49	212.76	161.50	511.45	94.01	132.20	570.00
11	84.82	139.63	569.23	226.25	173.16	497.69	100.42	143.05	564.81
12	92.20	151.43	563.51	235.36	184.43	486.09	105.37	155.90	559.14

Continued on next page

Table A.3 – continued from previous page

Step	Target			Replication			Replication (Genetic)		
13	99.50	163.00	557.33	248.94	194.70	470.21	114.85	167.77	551.86
14	106.72	174.34	550.72	257.19	203.79	458.15	118.44	177.76	546.33
15	113.84	185.41	543.67	266.79	212.39	444.19	127.41	189.24	538.21
16	120.85	196.21	536.20	275.39	220.38	430.47	135.17	200.44	529.96
17	127.76	206.71	528.33	282.41	227.69	417.87	141.56	210.90	521.89
18	134.55	216.92	520.07	288.70	234.38	405.67	147.84	221.06	513.45
19	141.22	226.81	511.43	294.33	240.59	393.81	154.01	230.89	504.66
20	147.77	236.36	502.43	299.41	246.28	382.29	159.37	240.06	495.95

Appendix B

The Code Display

```
Movement Complete!
The steps motor 1 took is: 0, corresponding to a cable length change of: 0.00mm
The steps motor 2 took is: 0, corresponding to a cable length change of: 0.00mm
The steps motor 3 took is: -521, corresponding to a cable length change of: -8.00mm
The steps motor 4 took is: 521, corresponding to a cable length change of: 8.00mm
The steps motor 5 took is: -978, corresponding to a cable length change of:-15.00mm
The steps motor 6 took is: 978, corresponding to a cable length change of:15.00mm
The steps motor 7 took is: 4237, corresponding to a cable length change of:64.99mm
The steps motor 8 took is: -4237, corresponding to a cable length change of:-64.99mm
```

(a) the result of first stepping test

```
Movement Complete!
The steps motor 1 took is: 2151, corresponding to a cable length change of: 33.00mm
The steps motor 2 took is: -2151, corresponding to a cable length change of: -33.00
The steps motor 3 took is: 1565, corresponding to a cable length change of: 24.00mm
The steps motor 4 took is: -1565, corresponding to a cable length change of: -24.00mm
The steps motor 5 took is: 978, corresponding to a cable length change of:15.00mm
The steps motor 6 took is: -978, corresponding to a cable length change of:-15.00mm
The steps motor 7 took is: -3585, corresponding to a cable length change of:-54.99mm
The steps motor 8 took is: 3585, corresponding to a cable length change of:54.99mm
```

(b) the result of second stepping test

Figure B.1: The Display of Arduino Control System.

```
Welcome to Arduino control panel!
Please input the 8 angles in order
type 'clear' to re-input

---------------------------------------

Now, please input the 1st angle:

delta s1 is:0.00
Now, please input the 2nd delta s:

delta s2 is:0.00
Now, please input the 3rd delta s:

delta s3 is:-8.00
Now, please input the 4th delta s:

delta s4 is:8.00
Now, please input the 5th delta s:

delta s5 is:-15.00
Now, please input the 6th delta s:

delta s6 is:15.00
Now, please input the 7th delta:

delta s7 is:65.00
Now, please input the 8th delta s:

delta s8 is:-65.00
Input complete!
```

Figure B.2: The input parameters for first stepping test.

```
Distance to the nearest point:
Height:240 mm
   10.1916    5.9839    9.6846    8.9962   25.3592    5.7040    7.1835    6.2378    4.1283

Height:340 mm
    5.9085    3.2570    5.4114    4.1478    5.1970    2.1388    5.5338    4.1535    6.4524

Height:440 mm
    5.8770    2.5728   11.3366    4.0684    3.2286    2.8352    8.7876    5.1880   10.1459

Height:540 mm
    7.8421    8.8544   10.5260    8.6281    5.5724    7.2780    7.5941    6.2517   11.6773
```

Figure B.3: The detection result while index=1000000 and H=240.

```
Iteration: 1000000
Distance to the nearest point:
Height:260 mm
    7.5602    6.4772    7.0509    2.4059   19.3812    4.0904   11.5647    7.4664    6.8634

Height:360 mm
    4.0825    5.6066    5.5248    1.9642    3.8820    3.0379    7.8672    3.1782    7.8525

Height:460 mm
    7.2979    5.3467    2.2831    7.4993    3.6660    6.2619    7.4820    5.4839    8.3480

Height:560 mm
    8.1052   12.6313    6.9750    9.3641    3.6025    5.9004   17.1690    4.9512    7.4249
```

Figure B.4: The detection result while index=1000000 and H=260.

```
Distance to the nearest point:
Height:280 mm
    6.5362    4.0102    7.9380    2.6343   10.0980    4.1573    6.8283    3.2028    4.8241

Height:380 mm
    6.0912    1.5101    4.3438    4.4031    3.4151    2.6523    5.5229    2.3801    3.9658

Height:480 mm
    8.7091    5.0989    3.5441    8.3705    3.1214    8.0341    4.9664    4.7573    6.8580

Height:580 mm
   12.0959    7.0461   13.6910    7.1938    7.6483   11.4657   18.2672    8.7651   11.4104
```

Figure B.5: The detection result while index=1000000 and H=280.

```
Height:300 mm
    7.0696    3.3457    5.8420    5.1691    6.9859    9.6064   12.0161    3.1330    8.0543

Height:400 mm
    3.2065    5.6137    6.2471    1.9011    3.2564    6.4037    7.3970    5.9186    7.0688

Height:500 mm
   10.3989    3.3835    9.2886    7.4610    3.5080    5.3177    9.7733    6.1874    1.8141

Height:600 mm
   33.8095   13.3727   30.6118    6.0586    9.5076    8.8049   30.4824   16.8214   32.5093
```

Figure B.6: The detection result while index=1000000 and H=300.