DESIGN, MANUFACTURING AND CONTROL OF 3-DOF SPHERICAL PARALLEL MANIPULATOR

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
UNIVERSITY OF TURKISH AERONAUTICAL ASSOCIATION

BY

GÖRKEM TAŞKIRDI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER
IN
MECHANICAL AND AERONAUTICAL ENGINEERING

OCTOBER 2023

Approval of the thesis:

DESIGN, MANUFACTURING AND CONTROL OF 3-DOF SPHERICAL PARALLEL MANIPULATOR

submitted by **GÖRKEM TAŞKIRDI** in partial fulfillment of the requirements for the degree of **Master in Mechanical and Aeronautical Engineering, University of Turkish Aeronautical Association** by,

Assoc. Prof. Dr. Suat DENGIZ
Dean, Graduate School of **Natural and Applied Sciences** _____

Assoc. Prof. Dr. Hamit TEKIN
Head of the Department, **Mechanical Eng, UTAA** _____

Assist. Prof. Dr. Masoud LATIFINAVID
Supervisor, **Mechanical Eng, UTAA** _____

Examining Committee Members:

Assist. Prof. Dr. Masoud LATIFINAVID
Supervisor, **Mechanical Eng, UTAA** _____

Assoc. Prof. Dr. Hamit TEKİN
Mechanical Eng, UTAA _____

Assist. Dr. Ferit SAIT
Mechanical Eng., Çankaya Uni. _____

Date: 25.10.2023

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Görkem TASKIRDI

Signature:

ABSTRACT

DESIGN, MANUFACTURING AND CONTROL OF 3-DOF SPHERICAL PARALLEL MANIPULATOR

TASKIRDI, Görkem

MSc., Department of Mechanical and Aeronautical Engineering

Supervisor: Assist. Prof. Dr. Masoud LATIFINAVID

OCTOBER 2023, 113 pages

This study focuses on the design, manufacturing, and control of a spherical parallel manipulator with three degrees of freedom (3 DOF). Spherical parallel manipulators have a wide range of potential applications in various industrial and scientific fields. This research focuses on a design optimization technique for improving the efficiency of a prototype spherical manipulator, which includes kinematic calculations, modeling, and control. Critical performance parameters such as the manipulator's workspace, speed, and smooth rotation were optimized throughout the design process, taking its kinematic structure into account. To achieve this, several different mechanical design alternatives were evaluated. In the prototype manufacturing phase, the production and assembly of the designed manipulator parts were carefully carried out. Material selection and prototype manufacturing processes were optimized to increase the durability and precision of the manipulator. In the control phase, the software used in 3D printers, which is disposed to manage the precise movements of the designed manipulator, was adapted for use in the prototype manipulator. Additionally, a computer-based simulation of the prototype spherical manipulator was performed. As a result, this study has addressed the integration of the design, manufacturing, and control processes of 3 DOF spherical parallel manipulators, demonstrating the efficient usability of an optimized manipulator in

industrial applications. This work may provide significant contributions to future manipulator designs and the development of robotic systems.

Keywords: Spherical parallel manipulator, optimization design, kinematic verification, control, simulation.

ÖZ

3 SERBESTLİK DERECELİ KÜRESEL PARALEL MANİPÜLATÖR TASARIMI, İMALATI VE KONTROLÜ

TAŞKIRDI, Görkem

MSc., Makine ve Uçak Mühendisliği Bölümü

Tez Yöneticisi: Dr. Öğr. Üyesi Masoud LATIFINAVID

EKİM 2023, 113 Sayfa

Bu çalışma, 3 derecelik serbestlik derecesine (DOF) sahip bir küresel paralel manipülatörün tasarımı, imalatı ve kontrolü ile ilgilenmektedir. Küresel paralel manipülatörler, birçok endüstriyel ve bilimsel uygulama alanında geniş bir kullanım potansiyeline sahiptir. Bu çalışma, bir prototip küresel manipülatörün verimliliğini artırmak amacıyla bir tasarım optimizasyonu yaklaşımını ele almaktadır ve kinematik hesaplamaları, simülasyonu ve kontrolünün sağlanmasını içerir. Tasarım aşamasında, manipülatörün kinematik yapısı dikkate alınarak, hareket aralığı, hız ve rahat rotasyon sağlayabilmesi gibi önemli performans metrikleri optimize edilmeye çalışılmıştır. Bu amaçla, birçok farklı mekanik tasarım alternatifi değerlendirilmiştir. Prototip imalatı aşamasında, tasarlanan manipülatörün parçalarının üretimi ve montajı titizlikle gerçekleştirilmiştir. Yapılan malzeme seçimi ve prototip imalat süreçleri, manipülatörün dayanıklılığını ve hassasiyetini artırmak amacıyla optimize edilmiştir. Kontrol aşamasında, tasarlanan manipülatörün hassas hareketlerini yönetmek için 3D yazıcılarda kullanılan yazılım düzenlenip prototip manipülatörde kullanıldı. Ek olarak, prototip küresel manipülatörün bilgisayar ortamında simülasyonu gerçekleştirildi. Sonuç olarak, bu çalışma, 3 DOF küresel paralel manipülatörlerin tasarım, imalat ve kontrol süreçlerinin birleşimini ele almış ve optimize edilmiş bir manipülatörün endüstriyel uygulamalarda verimli bir şekilde

kullanılabilirliğini göstermiştir. Bu çalışma, gelecekteki manipülatör tasarımlarına ve robotik sistemlerin geliştirilmesine önemli katkılar sunabilir.

Anahtar Kelimeler: Küresel paralel manipülatör, optimizasyon tasarımı, kinematik doğrulama, kontrol, simülasyon

ACKNOWLEDGMENTS

I would like to express my sincere thanks to my supervisor Dr. Masoud LATIFINAVID for his technical and material support in this thesis, whose inspiration and divine guidance made it possible for me to reach this academic level.

A huge thanks to Florian STEINER and Wendelin STEINER at Steiner Engineering for all their support.

I would like to express my invaluable gratitude to my colleague Nazlıcan DONMEZ and my family members for their moral advice and financial support during my thesis.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

CHAPTER 1

INTRODUCTION

Spherical parallel manipulators (SPMs) provide three degrees of freedom which enables the capacity of an object to move with 3 degrees of freedom in space. They are robotic systems that have received significant attention in recent years. Parallel manipulators have widespread usage in applications such as flight and automobile simulators, industrial robots, and mechatronics systems. Spherical parallel manipulators comprise a pyramid-shaped top plate, a cylindrical base platform housing the motors, and three equally spaced interconnected legs. Each leg consists of two links and three rotating joints. These manipulators are capable of handling many complex tasks due to their high precision, wide range of motion, and rapid responsiveness. However, the design, kinematic calculations, and control of these manipulators require careful attention for successful implementation.

1.1 Spherical Parallel Manipulator Structural System

A Spherical Parallel Manipulator is a mechanical structural system with three degrees of rotational freedom in every direction and a spherical joint or gimbal mechanism at its base. It is usually made up of a stationary base, a movable platform, and a series of linked stiff arms or links with spherical joints at both ends. Because of its capacity to give fine orientation and positioning control in three-dimensional space, these parallel manipulators are frequently employed in robotics and automation applications. The structural structure of the spherical parallel manipulator is intended to assure stability, precision, and great load-bearing capacity, making it suited for jobs like 3D printing, machining, medical operations, and simulators. It is mentioned under 6 headings, these are;

- Base
- Linkages
- Actuators
- End-Effector
- Sensors
- Control System

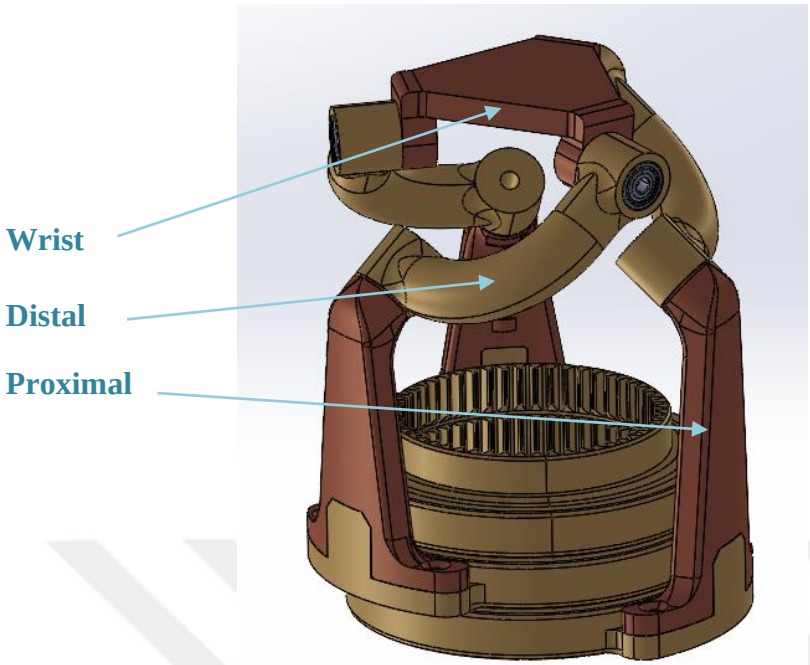


Figure 1. 1: Manipulator Links

This study focuses on the optimization of the design, kinematic calculations, and comparison with real-world data for a spherical manipulator with three degrees of freedom (DOF). Important performance features such as the range of motion, speed, and precision were attempted to be optimized, taking into consideration the kinematic structure of the manipulator, during the design phase, Various mechanical design alternatives were evaluated, resulting in an optimized design. Flexible couplings were used to ease the rotation of the motor shaft and bearings were employed to ensure there were no center irregularities. Gears were designed to increase the torque from the motors, while large bearings were designed to facilitate the rotation of the arms responsible for motion. The 3D model of the design was modeled in the SolidWorks and Catia V5 program. 3D printers were utilized in the production of prototype manipulator components.

The Marlin software, which is used in 3D printers, was adapted, and integrated according to the prototype's requirements to control this manipulator. Step, acceleration, and speed settings have been made according to gear and stepper motors. Sensors such as temperature etc. have been disabled. Besides, forward, and

inverse kinematics analysis were calculated, controlled, and simulated with the MATLAB program.

Finally, this study includes experimental tests aimed at evaluating the performance of the designed manipulator using the collected data. The primary objective is to validate the computer-based kinematic analyses with the improved prototype manipulator outputs. The results of this study may provide insights and significant contributions to the design and application of spherical manipulators, benefiting future projects and industrial applications.

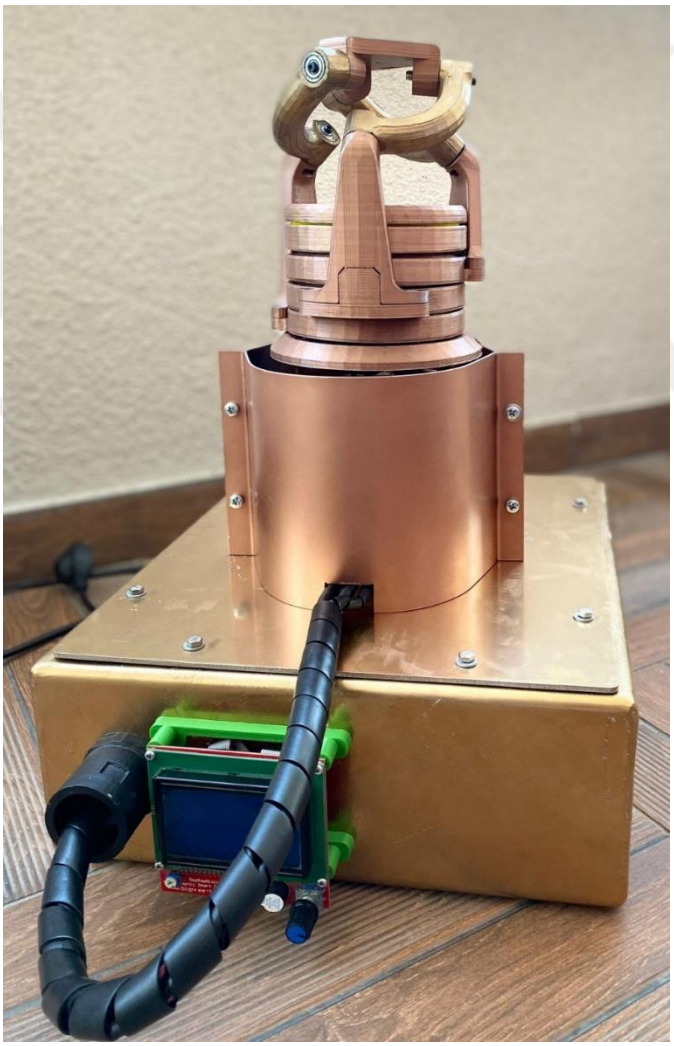


Figure 1. 2: Prototype Spherical Parallel Manipulator

1.2 Motivation

In the thesis we assume that,

- To make improvements on a conceptual design and to validate the computerized data and real-life outputs with kinematic analysis.
- The designed prototype spherical parallel manipulator to operate in the range of -30^{o}, $+30^{o}$.
- Prototype manipulator to carry 100 grams.
- Since stepper motors work more precisely than servo motors, it is easier to see the desired values more precisely.

Contribution of this study to the literature,

- Improved design addresses the shortcomings or problems of an existing concept. Creative new solutions have led to the development of more effective or more reliable manipulators.
- Kinematic calculations were used to understand the mobility of the design and how the mechanisms work. Accurate kinematic calculations helped predict the performance and behavior of the design.
- The computerized data was used to create virtual simulations of the design. This simulation showed how the design should theoretically work and enabled potential problems to be identified.
- Design improvements are experimentally tested in the real world. Real-world data was used to validate the performance and reliability of the design.
- This work can contribute to advances in related application areas (e.g., industrial machines, robotic systems, medical devices, transportation vehicles, etc.) by informing design improvements and kinematic analysis.

CHAPTERs 2

LITERATURE REVIEW

2.1 Design

Soheil Sadeqi, Shaun P Bourgeois, Edward J Park, and Siamak Arzanpour [1] discuss the design and performance analysis of a 3-RRR spherical parallel manipulator for hip exoskeleton applications. The design process includes kinematic and dynamic analysis and optimization of manipulator geometry and actuator layout to improve its performance. The article also presented the results of experiments to evaluate the performance of the manipulator, including the working area, singularity avoidance, and dynamic stability. It is concluded that the 3-RRR spherical parallel manipulator is very suitable for hip exoskeleton applications due to its high dexterity, compact size and good dynamic performance.

The kinematic simulation and computer-aided design (CAD) of spherical parallel manipulators with prismatic or revolute actuators are the main topics of this study by Gosselin, C., Perreault, L. L., & Vaillancourt, C. (2002) [2]. Beginning with a study of the kinematic analysis of spherical parallel manipulators, it briefly discusses singularity locus expressions as well as direct and inverse kinematic issues. The study then goes into detail about figuring out where these manipulators might function. Notably, it offers a computer program specifically created for CAD needs that enables interactive examination of manipulators with all designs. This package simplifies tasks like graphic animation of trajectory provided by either the direct or inverse kinematic module, workspace representation, and singularity visualization.

The authors of this article, Liu, X., Jin, and Gao (2000) [3], investigated the use of assessment criteria, such as stiffness and the conditioning index, to direct the choice of link lengths in 3-DOF spherical parallel manipulators (3-DOF SPMs) and to evaluate their operational performance. The work creates atlases of the global

conditioning index and global stiffness index inside the solution space to optimize these link lengths. A comparison of the manipulators' dexterity and stiffness reveals a relationship between global or local conditioning and global or local stiffness indices. This technique has broad applications in the design of both serial and parallel robots, enhancing the understanding and optimization of their performance.

This paper by Bai, S. (2010) [4] focuses on the best design of spherical parallel manipulators (SPM) with a set workspace. It presents a numerical technique to figure out the ideal design parameters, considering link dimensions and architectural features, with the goal of maximizing dexterity. The solution procedure is made simpler by the method's transformation of the optimization issue into a nonlinear least squares problem using a well-constructed objective function. The paper also deals with the problem of design space by constructing a set of inequalities based on connection dimensions, which define the area for workable SPM designs. Using real-world examples, this optimization method may be applied to create SPMs that are as flexible as possible for a given workspace need.

It is mentioned by Andrea Bulgarelli, Giorgio Toscana, Ludovico Orlando Russo, Giuseppe Air Farulla Marco, Indaco Basilio, and Bona's [5] description of the design and development of an open-source, 3D printable, low-cost anthropomorphic robotic hand with parallel spherical jointed wrists that is intended to mimic sign languages. Utilizing ready-to-use parts and 3D printing technology, the robotic hand is intended to be inexpensive and simple to construct. The parallel spherical joint wrist offers excellent dexterity and flexibility in hand movements, making it suitable for mimicking the intricate hand motions used in sign language. The robotic hand's design, testing, and outcomes are discussed in length in the article, which also offers comprehensive building and operation guidelines. It is concluded that the proposed robotic hand is a promising solution for sign language production and other applications that require high dexterity and flexibility.

This paper was written by Essomba, T., Hsu, Y. C., Sandoval, J., Laribi, M. A., & Zeghloul, S. (2019) [6] presents the kinematic design of a robotic manipulator tailored for craniotomy procedures. The design process is driven by motion capture

experiments that measured the motion of a surgical drill during craniotomy on human cadavers. The paper discusses the experimental results, emphasizing the need for a remote center of motion (RCM) in this medical application. To address this requirement, the paper introduces a novel 3-RRR spherical parallel mechanism (SPM) featuring a reconfigurable base capable of reorienting the first revolute joint of the RRR legs. This innovation is achieved through the integration of three pantographic linkages that manipulate the SPM's base. Its effects on workspace and kinematic performance are taken into account when the kinematics of this novel mechanism are examined. This kinematic data is used as the basis for an optimization process, and the behavior of the improved mechanism is evaluated by analyzing the drill motion trajectories. The comparison highlights the considerable contribution of the reconfiguration variable to the mechanism's dexterity by contrasting the newly built mechanism with the standard SPM with a trihedral base.

In this study written by Dinh, T. V., & Kheylo, S. (2020) [7], a screw calculus-based theoretical framework for spherical mechanisms with a parallel structure is presented. The method tackles the identification of specific manipulator locations as well as direct and inverse velocity difficulties. The process entails figuring out which power screws, inside each kinematic chain, reciprocate with the unit vectors of the non-drive pair axes. After that, velocities of the spherical mechanism in a parallel construction are described by equations derived from these power screws, with solutions given for both direct and inverse velocity issues. The study shows that degeneration of kinematic screw systems in connecting chains is associated with the loss of one or more degrees of mobility, whereas loss of controllability arises from linear dependencies within the power screw system. The algorithms proposed here, relying on screw calculus, offer valuable tools for optimizing manipulator parameters and analyzing manipulator functionality.

In this study written by Vatsal, V., & Hoffman, G. (2021) [8] presents a design that achieves rotating motion for wrist prostheses without the use of motors, meeting the specifications of a conventional single 3-degree-of-freedom (3-DOF) spatial mechanism. Inspired by the Kresling origami idea, the design called the 3RRR

slanted arm allows for the necessary pronating and supinating motions that are sometimes absent from wrist prostheses based on parallel mechanisms. Furthermore, the circular movement capability of this design is in line with the requirements of a typical 3-DOF spatial mechanism. This work provides an initial analysis before building a physical prototype. It also forms the basis for a new prosthetic wrist for trans-radial amputees that can be activated utilizing motion capture data and surface electromyography (sEMG) control. Surface electromyography has already demonstrated success in enabling precise limb and joint control for prosthetic rehabilitation patients using the remaining muscles and nerves from post-amputation stumps.

This paper written by Chen, Y. J., Tung, W., Lee, W., Patel, B., Bučinskas, V., Greitāns, M., & Lin, P. T. (2023) [9] describes an automated platform stabilization system based on a three-RCC structure that can rotate on the Z-axis at different tilting degrees. With stabilization periods of 5 seconds at 30 degrees, 3.5 seconds at 20 degrees, and 3 seconds at 10 degrees, automated stabilization efficiency is established. The initial model had instability because of isolated gears, prompting changes in mechanism design, such as adding internal gears with bearings to improve system stability. To increase stability, torque was reduced and the number of arms supporting the platform was reduced. While real-time stabilization was discovered to be less efficient than kinematic analysis, which performed well with shorter duration, future research may investigate achieving accurate angle control and rotation over 360 degrees with servo motors, as well as improving responsiveness to sudden changes in tilting direction during stabilization. Furthermore, despite their increased space requirements, stepper motors may be used for more efficient and precise stabilization.

This research was written by Altuzarra, O., Tagliavini, L., Lei, Y., Petuya, V., & Ruiz-Erezuma, J. L. (2023) [10] compares the performance of a tripod-type parallel continuum manipulator, 3PFS, to that of its rigid equivalent, 3PRS. Reduced mobility in rigid parallel manipulators is described using equations that connect end effector pose parameters, whereas continuum architectures, characterized by flexible

linkages, do not readily allow for such algebraic expressions due to deformation under actuation and load. However, specific mechanical arrangements can limit deformation in certain directions, resulting in constraint-like effects. The study models the flexible 3PFS tripod continuum manipulator and reveals that, unlike the rigid case, the output-dependent parameters of the flexible mechanism exhibit some dependence on the end effector. This is due to the mechanical arrangement of flexible rods, which do not rigidly constrain spherical joints in specific vertical planes. This approach produces an analytical expression that connects output-independent parameters to parasitic movements, like constraint equations in the rigid 3PRS example, offering insight into the behavior of the 3PFS manipulator.

This study is written by Leal-Naranjo, J., Wang, M., Paredes-Rojas, J., & Rostro-González, H. (2019) [11] describes a three-degree-of-freedom spherical parallel manipulator that may be used as a prosthetic wrist. The investigation begins with the solution of the inverse position issue and the generation of a closed-form equation. A mobility study using the screw theory technique reveals that the suggested mechanism is capable of achieving spherical motion. Screw theory is used for velocity analysis, which results in an input-output velocity equation. A preliminary virtual design of the mechanism, including workspace analysis and analytical static assessment, is provided. The article finishes with dynamic simulations of three important wrist movements: pronation-supination, flexion-extension, and ulnar-radial deviation. The findings show that the suggested mechanism can achieve the whole range of motion necessary for daily activities, with actuation torque levels that are critical for future development.

2.2 Control and Kinematic Analysis

In their paper published in 2014, Aibek Niyetkaliyev and Almas Shintemirov [12] present an approach to obtain unique kinematic solutions of a spherical parallel manipulator. The approach relies on the use of algebraic equations to represent the kinematic constraints of the manipulator and uses these equations to solve for the manipulator's configuration. The article discusses the mathematical methods and algorithms used in this approach and demonstrates its effectiveness in obtaining

unique solutions for the kinematic configuration of the spherical parallel manipulator.

This paper is written by Bozorgi, E. R. J., Yahyapour, I., Karimi, A., Masouleh, M. T., & Yazdani, M. (2014) [13] delves into the design, dynamics, and control aspects of a two Degrees-of-Freedom (DOFs) spherical Parallel Mechanism (PM) developed at Laval University. The study begins with the determination of geometric and inertial parameters through CAD software to optimize weight and prevent interference between the two limb movements. The dynamic analysis employs a unique approach that dissects the mechanism into subsystems and combines kinematic analysis, Lagrangian, and Newtonian methodologies. The paper evaluates the approach's performance for the dynamics of the mechanism using various test trajectories for the end-effector, comparing the results to those obtained from dynamic analyzer software to validate the proposed algorithm. Additionally, a co-simulation between MATLAB and MD-Adams is conducted to implement control via the computed torque method, successfully guiding the end-effector along the desired trajectory. The paper demonstrates the effectiveness and correctness of the approach in designing, analyzing, and controlling this specific spherical Parallel Mechanism.

This paper written by Li, T., & Payandeh, S. (2002) [14] focuses on optimizing the workspace of a spherical parallel mechanism used in laparoscopic surgery, where the choice of this mechanism is driven by its unique characteristics. The study examines two specific designs: a haptic device intended for training purposes and a laparoscope holding mechanism. The latter design must meet additional constraints related to minimizing the space occupied above the patient during surgery. The primary objective is to address the design challenge and provide these mechanisms with the largest possible workspace. The paper not only presents the optimal design parameters but also outlines the design of a haptic interface and the laparoscope holding mechanism based on these parameters. To achieve this, the paper employs a Genetic Algorithm (GA) approach, which proves effective in selecting the optimal

design parameters to maximize the spherical parallel mechanism's workspace for laparoscopic surgery applications.

In a 2005 paper by N. S. TLALE and P. ZHANG [15], a method was found for teaching the design of parallel manipulators and controllers using MATLAB, Simulink, SimMechanics, and CAD. The method involves using these software tools to teach students the principles of parallel manipulator design, kinematics, dynamics and control. The article describes the use of MATLAB and Simulink for modeling and simulation, SimMechanics for dynamic analysis, and CAD for geometric modeling. The article also presents the results obtained by applying this teaching method in a university course, showing that students can gain a solid understanding of parallel manipulator design and control using these software tools.

In this article written by Shaoping Bai, Michael R. Hansen, and Torben O. Andersen [16], they present a method for modeling a special class of global parallel manipulators using Euler parameters. The method involves the use of Euler parameters to represent the direction of the end effector of the manipulator, allowing for a more compact and efficient mathematical representation of the manipulator's kinematics. The article presents an in-depth analysis of the kinematic and dynamic properties of this class of manipulators and shows that the use of Euler parameters leads to a significant simplification of the mathematical equations involved in the analysis. The article concludes that the proposed method provides a useful tool for modeling and analysis of this class of global parallel manipulators.

The paper by Houssem Saafi, Med Amine Laribi, and Said Zeghloul [17] presents a study of the advanced kinematic model resolution of a custom spherical parallel manipulator and compares it to real-time validation. The study focuses on the resolution of the forward kinematic model of the manipulator used to determine the position and orientation of the end effector relative to the base. The article compares different methods to solve the advanced kinematic model and evaluates their performance through simulations and experiments. Real-time verification of the kinematic model is also performed using data from motion capture systems. The study found that the proposed method to solve the forward kinematics model of the

manipulator yielded accurate results that were in good agreement with the real-time validation. The article concludes that this method is a reliable and efficient solution for the kinematic resolution of the custom spherical parallel manipulator.

This paper is written by Chaker, A., Laribi, M. A., Zeghloul, S., & Romdhane, L. (2011) [18] focuses on the design and analysis of a spherical parallel mechanism (SPM) intended for use as a haptic device in a medical application, specifically for minimally invasive surgery involving anastomosis techniques. The study begins by determining the necessary task space for performing an anastomosis procedure, utilizing a motion capture system and the expertise of an experienced surgeon. Experimental data from the surgeon's actions are used to define the required workspace. Subsequently, an SPM is designed with a workspace that encompasses the desired range. An optimization process is carried out to identify the optimal SPM configuration that closely matches the desired workspace, with Genetic Algorithms employed to solve this problem. The paper also explores the impact of the tool's rotational range on the structural parameters and workspace. Finally, through simulations, the paper demonstrates that the obtained design parameters successfully achieve the required workspace for the anastomosis technique in medical applications.

This paper is written by Kizir, S., & Bingül, Z. (2012) [19], sliding mode and PID controllers were used to operate a high-precision 6-degree-of-freedom Stewart platform. These controllers were built within a Dspace DS1103 real-time controller in the Simulink environment. These controllers' whole development cycle from subsystems to the main model—was discussed in detail. The experimental outcomes demonstrated outstanding performance, with both controllers demonstrating minimal steady-state error in position control and attaining exact target placement with errors below 0.5 m utilizing step inputs. Even though PID had faster responses because of larger gains, the sliding mode controller had better overshoot control than PID. Both controllers responded similarly in tracking trials without external loads, but when nonlinear external pressures were applied to the moving platform, the sliding mode controller beat the PID controller, demonstrating its robustness in

dealing with difficult circumstances. This study provides insights into the relative merits of PID and sliding mode controllers for high-precision motion control applications and serves as a notable example of real-time controller creation utilizing Matlab/Simulink and the Dspace DS1103 platform.

This paper is written by Vidaković, J., Lazarević, M. P., Kvrgić, V., Dančuo, Z., & Ferenc, G. (2014) [20] emphasizes how crucial it is to create effective robot kinematics algorithms, especially when it comes to serial manipulators. Even though kinematic modeling is typically done in Cartesian space, the shortcomings of usual orientation representations, such as Euler angles and rotational matrices, have prompted the search for a more reliable, succinct, singularity-free, and computationally efficient method to describe rotational information. The approach suggested here uses unit quaternions and results in kinematic modeling in dual quaternion space. In order to solve kinematic problems, the paper offers a thorough overview of the spatial descriptions and transformation methods that can be used in these spaces. A particular emphasis is placed on the various mathematical formalisms used to represent rigid body attitudes, such as rotation matrices, Euler angles, axis-angle representation, and unit quaternions, while also highlighting how these formalisms are interconnected. Additionally, it emphasizes the benefits of using quaternion-based kinematic modeling and proposes a unique direct kinematics technique in the dual quaternion space. This approach is used to the human centrifuge, a 3-DOF robot manipulator, to show how it works.

This work written by Tursynbek, I., & Shintemirov, A. (2020) [21], presents a new approach to modeling and simulating parallel manipulators in the CoppeliaSim robot simulator, including an example of a Specific Parallel Manipulator (SPM) with coaxial input shafts. The paper covers the manipulator's kinematics and offers a thorough breakdown of the CoppeliaSim modeling procedure, with an emphasis on using the CoppeliaSim-MATLAB API interface to operate the model from MATLAB. The process is demonstrated by means of simulation scenarios, which are confirmed by means of experimental testing on a real manipulator prototype. The outcomes show how accurate the suggested approach for simulating parallel

manipulators is, and they also imply that it may be used to describe and simulate other closed kinematic robot systems. Furthermore, it is demonstrated that the CoppeliaSim model's MATLAB-based control technique is a useful tool for motion analysis and collision detection in parallel manipulators, with exciting opportunities for design optimization and a range of applications.

This work is written by Güzin, D., & Gezgin, E. (2022) [22] proposes an improved conceptual design of a brain surgical manipulator, which builds on the same author's previous work of a 2-degree-of-freedom spherical parallel manipulator. The system becomes flexible for a variety of brain surgical techniques, including craniotomy, neuroendoscopy, and deep brain stimulation, by adding various modules to the manipulator platform. The work focuses on the scenario of opening a bone flap on a human skull and presents trajectory planning for this surgical procedure to demonstrate its capabilities. In the MATLAB SimMechanics environment, a virtual model of the manipulator is generated to assess the dynamic behavior and actuator torque needs under a particular trajectory. Analytical dynamic analysis using the Lagrange technique confirms the dependability of the virtual model, with a minimal divergence of 0.29% and 0.22% in needed torque values compared to the virtual and analytical models, respectively. The virtual model and dynamic analytic technique are confirmed further by hardware verification on a real prototype of the spherical manipulator, as well as an assessment of trajectory tracking capability, proving the system's potential for use in brain surgery.

This study written by Zarkandi, S. (2020) [23] describes a unique 3-DOF 3-PRR Spatial Parallel Manipulator (SPM) with symmetrical construction and a star-shaped foundation. The manipulator has actuated intermediate revolute joints with motors positioned on the base, similar to a traditional parallel manipulator. The paper describes the manipulator's construction in detail, performs mobility analysis using screw theory, and solves both inverse and forward position kinematic issues in closed form. The inverse issue yields up to eight real solutions, establishing eight working modes, whereas the forward kinematics are given by an eight-degree univariate polynomial. Numerical techniques are used to calculate the manipulator's orientation

workspace, and singularity and isotropy evaluations are done using Jacobian matrices, showing two forward kinematic singularities at the central point of the star-shaped base. A kinematic conditioning index (KCI) is used to analyze the manipulator's kinematic dexterity, and strategies for identifying distinct working and assembly modes during motion are described.

This study written by Enferadi, J., & Jafari, K. (2020) [24] proposes a systematic technique for doing closed-form dynamic analysis on the 3(RSS)-S parallel manipulator, a unique completely spherical robot with actual co-axial actuated shafts. This robot is designed to rotate completely around a vertical axis, making it suitable for applications such as celestial orientation and rehabilitation. The study describes the construction of the robot and performs inverse position, velocity, and acceleration evaluations. A systematic methodology for obtaining the dynamic equations of motion is provided using Kane's method, proving that the inverse dynamics of the manipulator can be reduced to solving a set of three linear equations with three unknowns. Furthermore, the research proposes a computational technique for solving the inverse dynamics and gives simulations for several moving platform trajectories, providing insights into the robot's dynamic behavior.

CHAPTER 3

MECHANICAL DESIGN

This prototype spherical manipulator was inspired by the spherical manipulator design known as the Orbitra [25]. The performance, energy efficiency, and usefulness of spherical manipulators could be improved through mechanical design optimization.

Geometric Tolerance Enhancement: The geometry of the spherical parallel manipulator's links and axes was optimized. This aided in making the manipulator smaller, lighter, and faster. The design was revamped in comparison to step motors, and internal structures were modified accordingly. While all dimensions of the components were increased, new bearing holes were introduced, and the components were redeveloped based on geometric tolerance.

Material Selection: Choosing the right materials for the manipulator's links and other components is essential for considerations of weight, strength, and durability. Therefore, lightweight, and high-strength materials were preferred, and parts printed from a 3D printer were favored. PLA was selected as the material of choice **Figure 1.2**.

Optimization of Range of Motion: The manipulator's range of motion was carefully considered during the design phase. Providing the necessary range of motion for desired applications while preventing unnecessary excessive movement contributed to energy savings. First, the motors and flexible couplings are connected to each other. Then the motors with couplings are connected to the gear via the shaft, thus, the torque and energy efficiency of the manipulator are increased. Additionally, large bearings were designed to facilitate the rotation of the motion-inducing arms.

Link Sizing: The dimensions and proportions of each link were carefully determined during the design process, enhancing balance and rotational performance. Part dimensions have been doubled compared to the norms and extra tolerance has been given to some parts, ensuring smooth rotation of the manipulator.

Load Distribution and Balancing: A structural design was created to support the main skeleton, reducing the load on motors or arms. Thus, the manipulator's load-carrying capacity was optimized, and balanced load distribution was achieved. Furthermore, both arms and the main skeleton featured gears with bearings, facilitating load distribution.

The manipulator was designed using computer-based programs, specifically Catia V5 [26] and Solidworks. [27]

Figure 3. 1: Spherical Parallel Manipulator

3.1 Used Materials on Design

The prototype spherical manipulator assembly image and bill of materials are as shown below **Figure 3.2**.

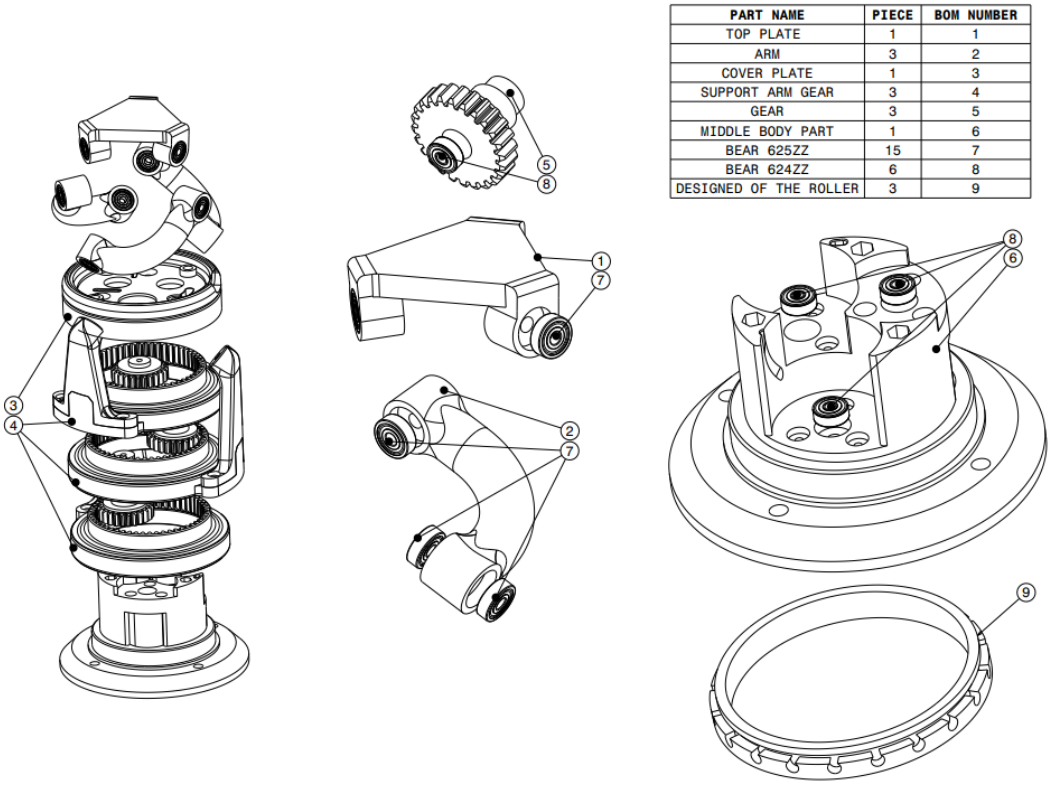


Figure 3. 2: Technical drawings of bill of material

The prototype spherical manipulator assembly of the main parts**.** They include the base, supported arms, middle body part and bearings shown below **Figure 3.3**.

Figure 3. 3: Status before basic assembly

3.1.1 Top Plate

The top plate of the manipulator is shown with 625zz bearings. By placing X, Y, and Z coordinates on the plate, the manipulator's movements can be observed clearly. The upper-side design is triangular shape. A cylindrical design was made to which bearings can be connected at the ends of the concentric triangular table. The reason for using bearings is to ensure smooth rotation during angular movement **Figure 3.4**.

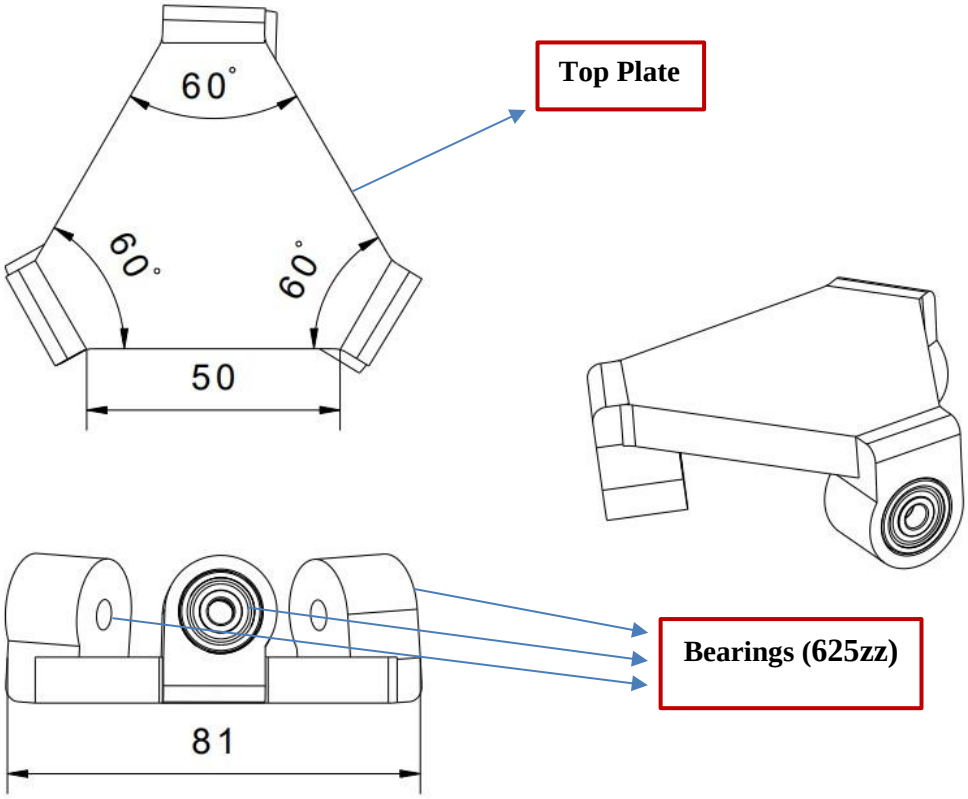


Figure 3. 4: Top plate with bearing 625zz

3.1.2 Arms

A total of 4 bearings (625zz) were used in one arm, at the back and front. In this project, 3 arms were used, and they were designed to be 90 degrees. It can easily perform cylindrical angular movements. Three arms will be connected to the top plate to support them. M3×M5×36 shafts, M3×40 bolts, flakes, and fiber nuts are used with the bearings as shown in **Figure 3.5**.

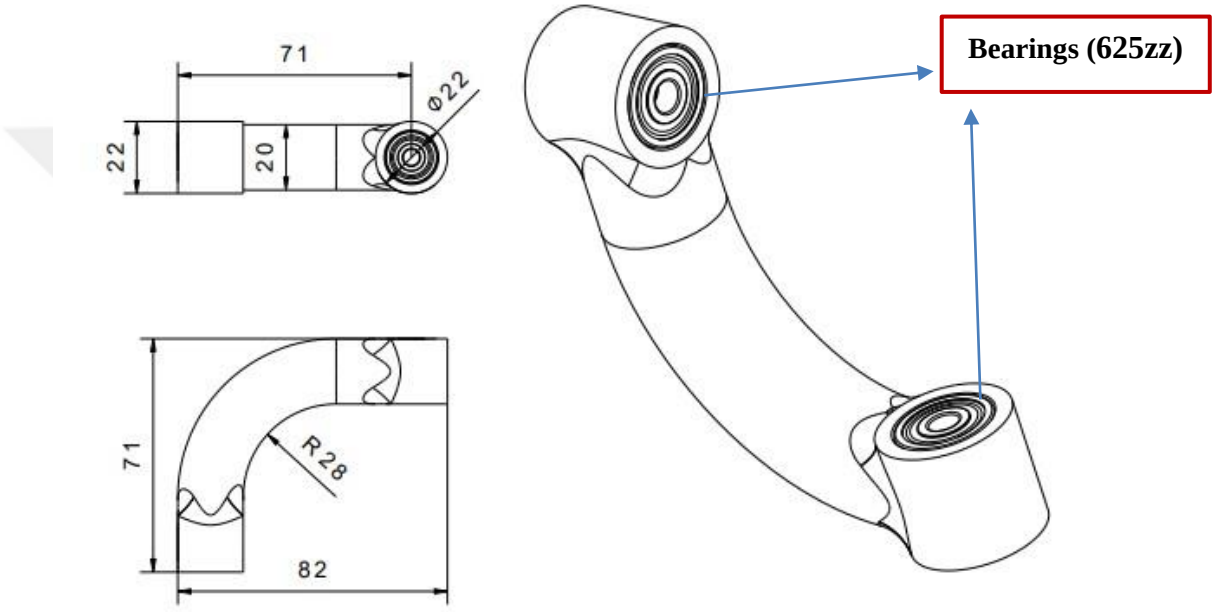


Figure 3. 5: Arm with bearing 624zz & 625zz

3.1.3 Support Arms

As mentioned earlier, the support arms are connected by the arms. M3×M5×24 shaft, M3×30 bolt, flake and fiber nut are used in the bearing. The upper part is designed with an angle of 130 degrees, this is to provide an easy movement in arm rotations. In addition, countersunk bolts are used in holes with a center spacing of 60 mm compared to other places. The reason for this is to hide the bolts in the support arm and ensure that they do not hit another section during rotation **Figure 3.6**.

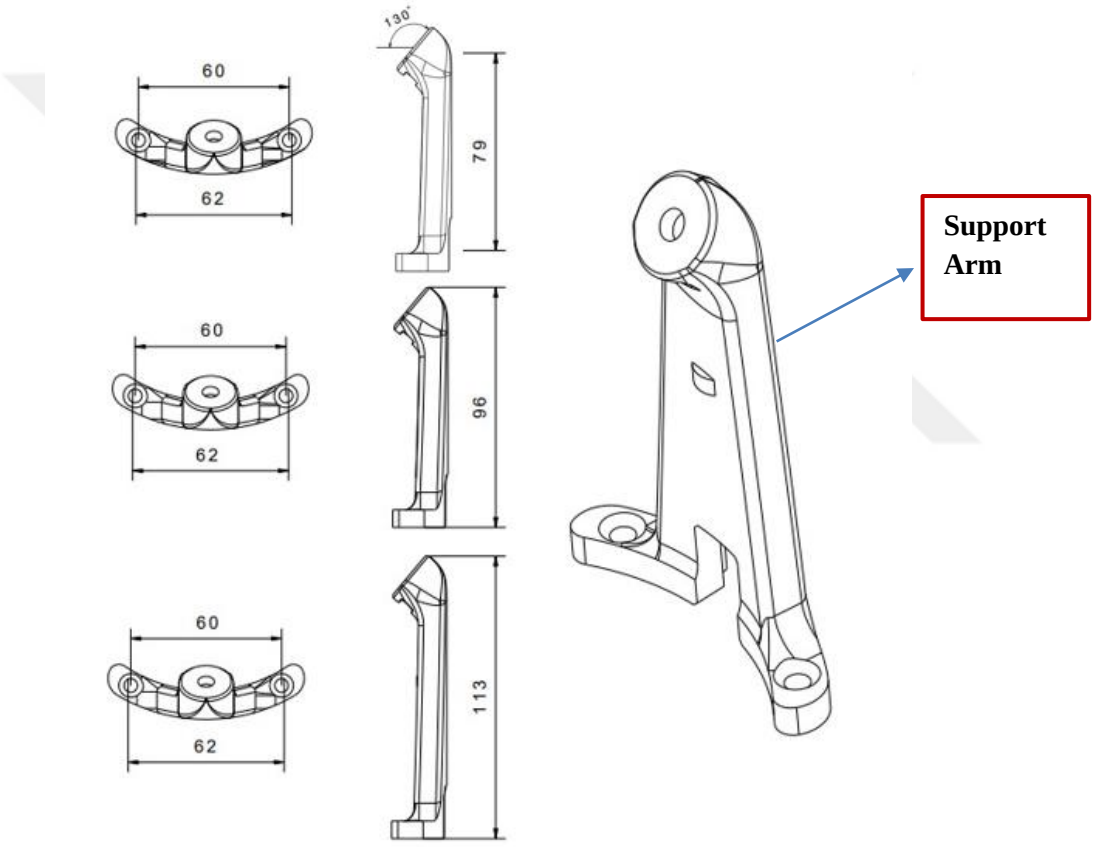


Figure 3. 6: Support arms and measurements

3.1.4 Top cover

To cover the top of the manipulator, a design was designed. It is secured with M4 bolts and placed over the internal gear shown in **Figure 3.7**.

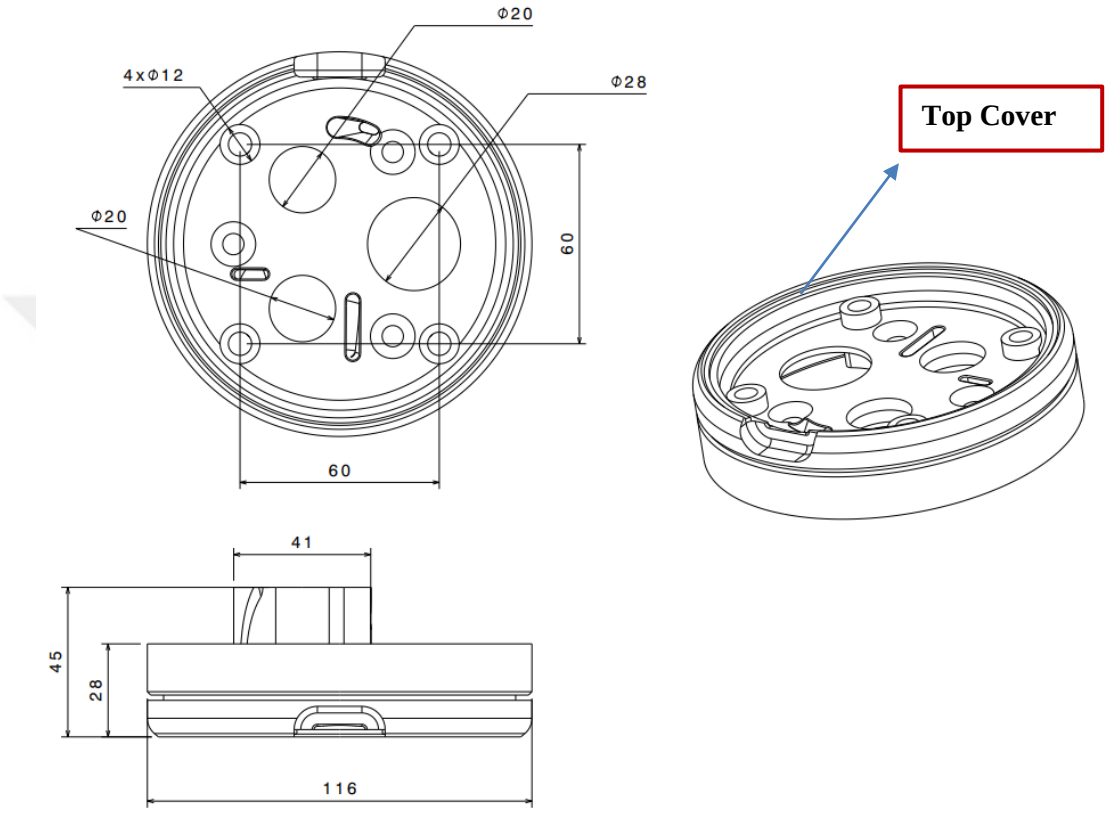


Figure 3. 7: Top Cover

3.1.5 Designed Bearings

The designed bearing has been utilized to reduce friction between two surfaces and facilitate smoother rotation or sliding motion. This has helped in minimizing wear and tear, thus extending the manipulator's lifespan, reducing energy losses, and enabling more efficient operation. Additionally, it is predicted that it will increase weight-bearing and load-carrying capacity. The designed bearing aims to reduce vibration and noise. After the part was produced, the balls used in bearings were placed into the designed bearing.

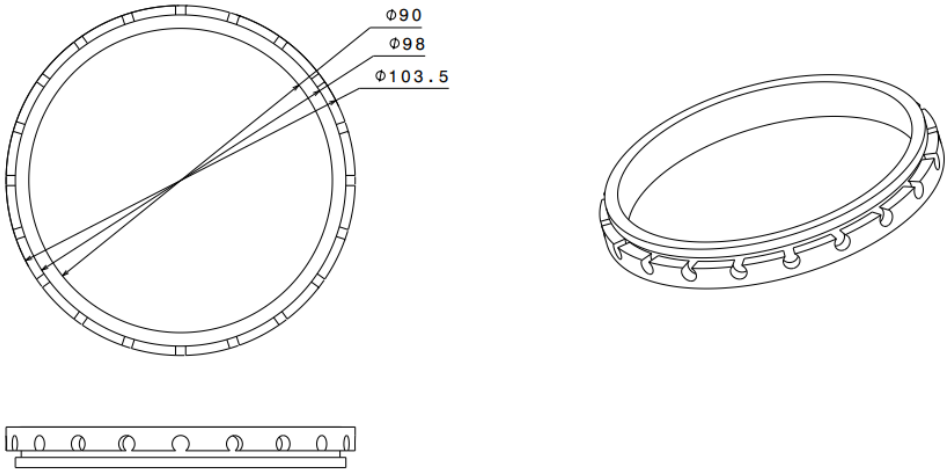


Figure 3. 8: Technical drawing of designed bearing

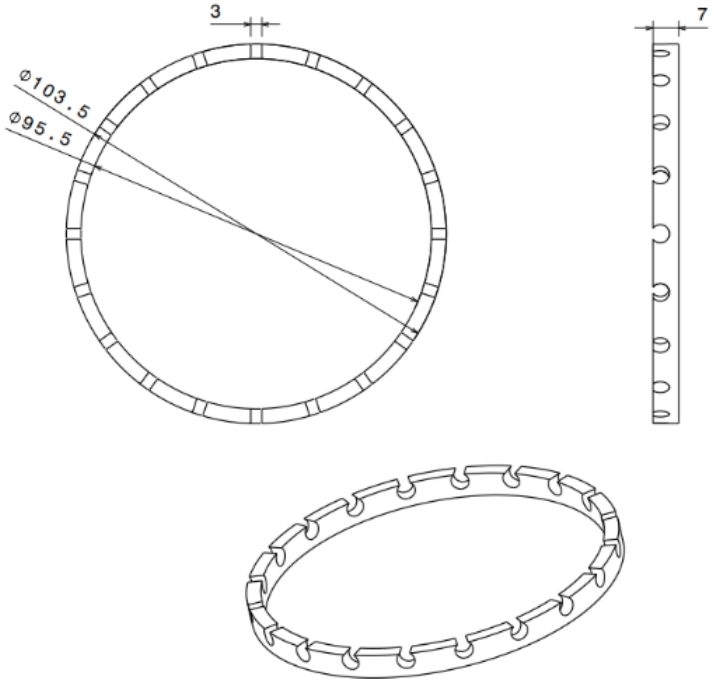


Figure 3. 9: Detailed technical drawing of designed bearing

Figure 3. 10: Designed bearing

A design was made to reduce the manipulator's upper and lower clearances. Balls were assembled on both sides of the bearing designed for this purpose, thereby enhancing the stability of its operation. Although it wasn't used in the thesis, it will be in other research projects in the future.

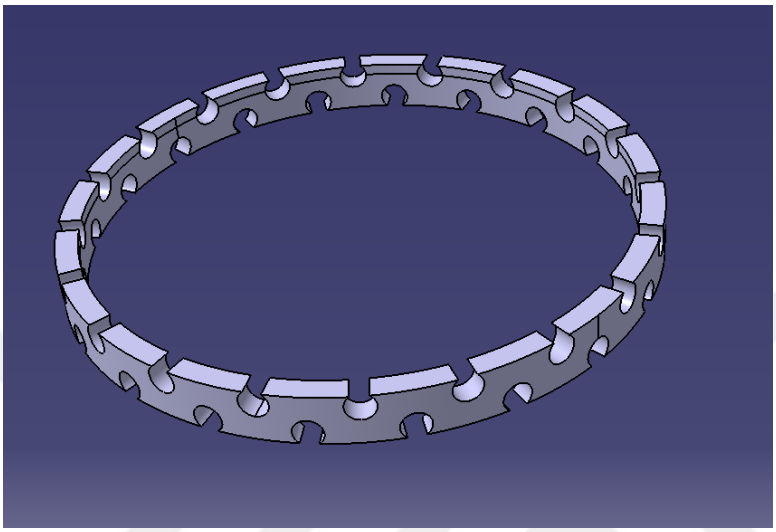


Figure 3. 11: Designed bearing for future work

Figure 3. 12: Designed bearing for future work

3.1.6 Support arm gear

This gear design is intended to make angular rotation smooth. The support arm gear is assembled on the support arm. In this project, three support arm gears were used shown **Figure 3.13**.

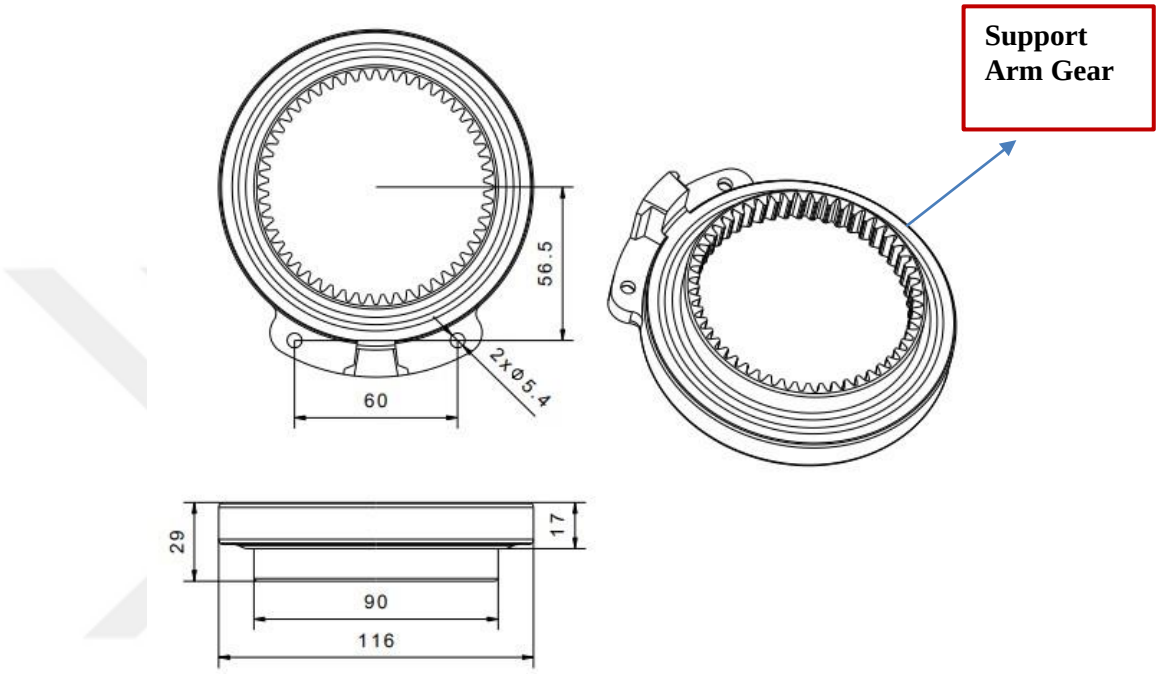


Figure 3. 13: Support arm Gear

3.1.7 Middle body part

The design is the main body where all the parts are assembled. The gears are connected to the support arm gears and rotated angularly. The gears are directly connected to the motors with the shaft and move with the motors. Bearings 624zz are used to achieve better rotation on the underside of the gears. Supported arm gears shown in **Figure 3.13** , and assembled body with motors shown in **Figure 3.14,**

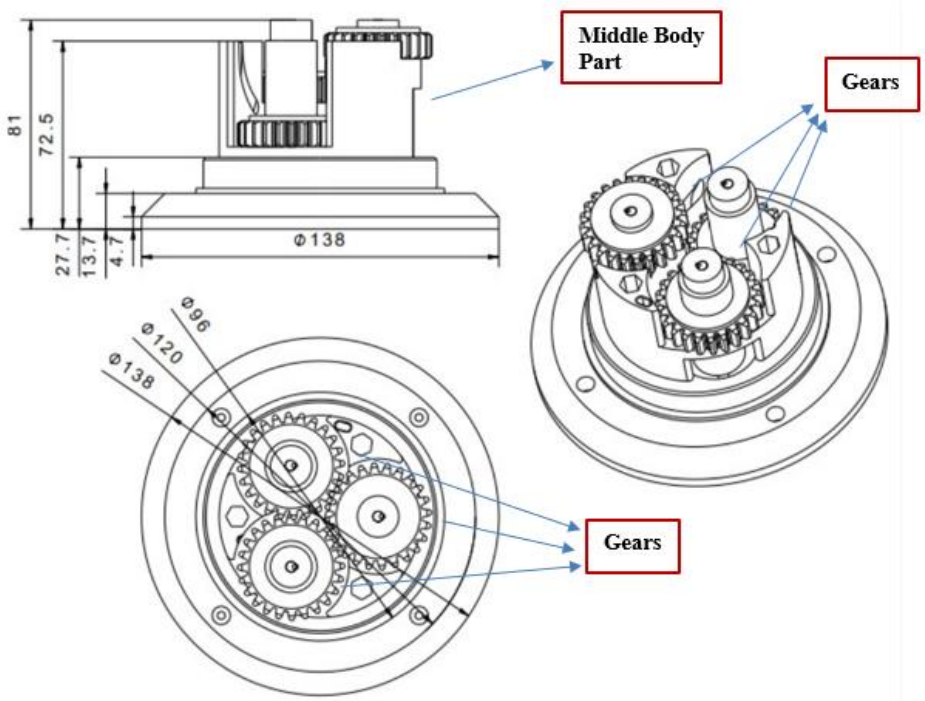


Figure 3. 14: Middle body part and gears

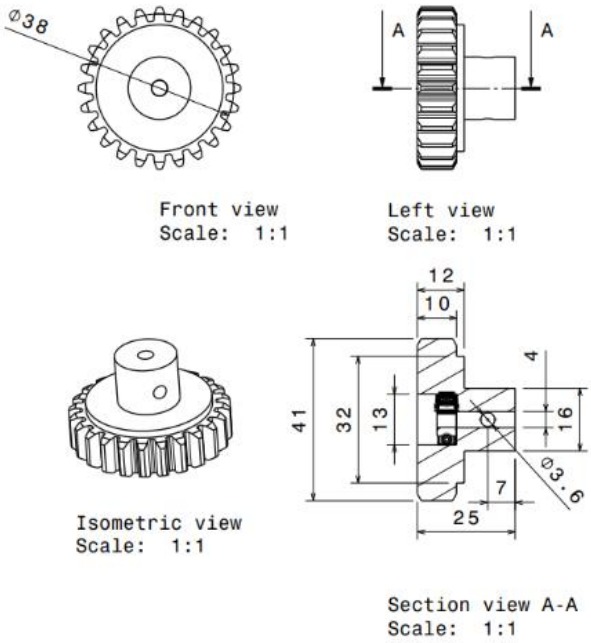


Figure 3. 15: Gear

3.1.8 Base and motor holder

In this design, the base and motor holders are designed separately the assembled state. The base supports the entire system, and the middle body part is positioned on top of the base. Motor cables are not visible from outside the system. There are three openings on the base, which serve as areas for the cables to exit and connect to the power supply **Figure 3.16**.

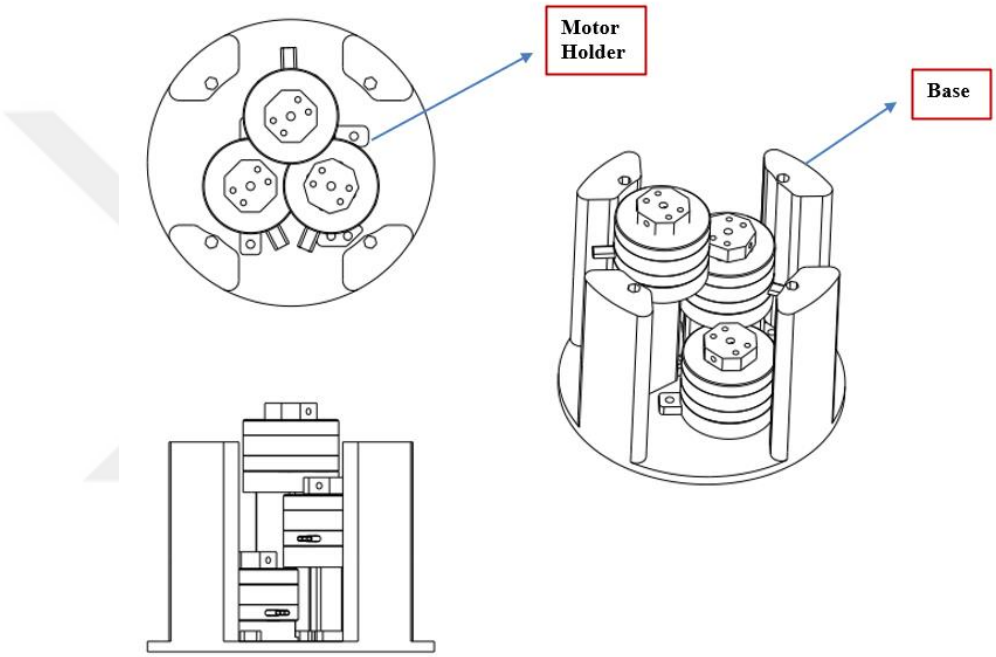


Figure 3. 16: Base and motor holder

3.1.9 Designed Manipulator Box

The manipulator's electrical box is designed to keep electrical connections safe, organized and protected. The box ensures safe and orderly operation of electrical connections. The Box is designed to avoid any problems, and includes an LCD screen, control card and power supply.

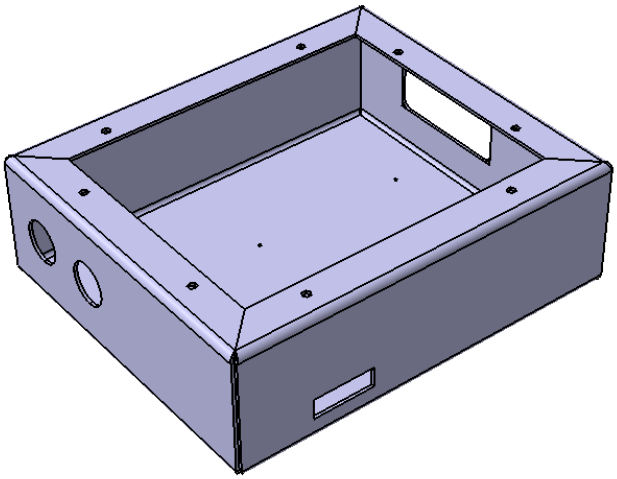


Figure 3. 17: Sheet metal electrical box

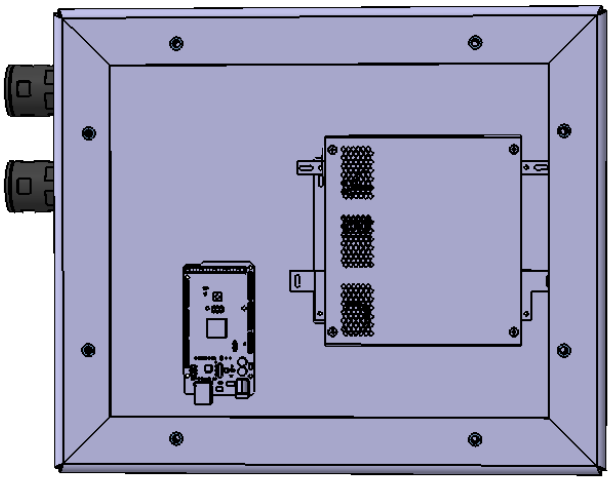


Figure 3. 18: Completed electrical box

Note: The reason is the length measurements of the parts in the images are not given is that they are made in 3D because they are processed parts. For this reason, all dimensions are not shown in the figures.

3.2 Production of Materials

Prototype Production: During the design phase of mechanical systems, it is critical to see how the design will operate in practice. Using 3D printers allows designers rapid prototyping to convert their concepts into tangible products. This assists in identifying design problems and improvement possibilities at an early stage.

Custom Part Manufacturing: It is common for mechanical systems to require specialized parts and components. These parts can be produced quickly and affordably by using 3D printers.

Manufacturing Complex Geometries: For generating complex or geometrically complex objects, 3D printers are more effective than conventional manufacturing techniques. Particularly for mechanical systems of the next generation, this provides design freedom.

Low Production Costs: Using conventional techniques to produce small quantities or manufacture on demand might be expensive. For low-volume production, 3D printers are cheaper.

Lightweight Complex Structures: Mechanical systems can require lightweight, complex, or optimized structures. 3D printers allow materials to be built up in layers, making it simpler to build lightweight and detailed designs.

Tolerance and Accurate Fit: Because 3D printers can make components with perfect tolerances more accurate mechanical systems may be created.

Quick Manufacturing: Compared to conventional machining techniques, 3D printers provide a faster production process, allowing ideas to be realized quickly.

Design Innovation: By enabling designers to realize new concepts quickly and imaginatively, 3D printers promote innovation in the design of mechanical systems.

As a result, 3D printers are an important tool used in various stages of mechanical systems, such as design, prototype production and special parts production. They make design processes more efficient and effective. For the reasons mentioned

above, a 3D printer was found to be suitable and was preferred when producing the prototype spherical parallel manipulator in this thesis.

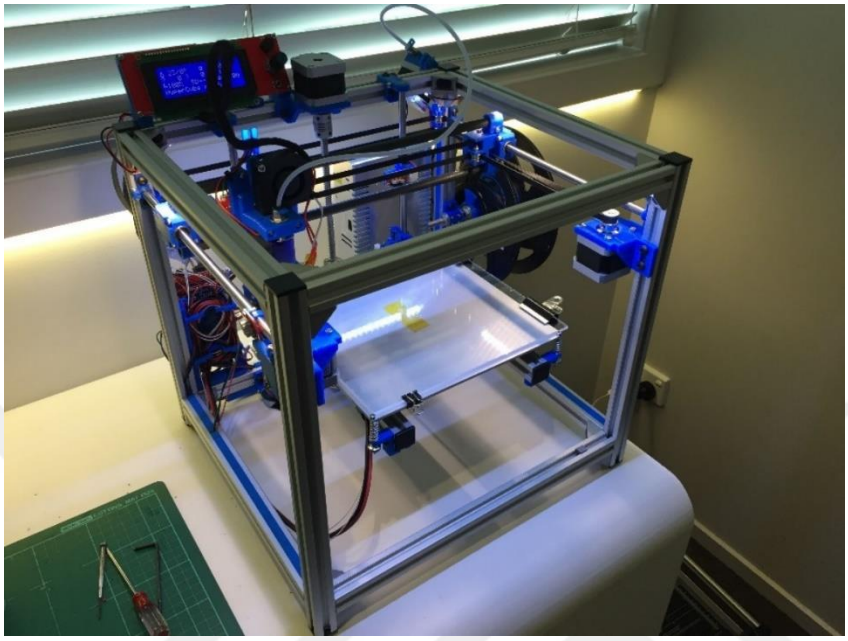


Figure 3. 19: Hybercube 3D printer

3.3 Errors in the Prototype due to Using 3D Printer

In this study, 3D printer errors were observed and the reasons for this error and deviation are as shown in **Figure 3.20** and **Figure 3.21**.

Figure 3. 20: 3D part with errors

Figure 3. 21: 3D printed part with errors

- Using filaments of different brands caused clogging in the nozzle, which affected print quality. Since the melting temperature of each brand of filament varied, difficulties occurred at the time of printing, which affected the print quality.
- The 3D printer used has 3D parts that have already been printed and there is a %1 margin of error for new products at the time of printing.
- The 3D printer lacks a cooling system; if the hot filament is not cooled down immediately, deformations can occur in the printed part. This can cause problems, especially for objects with complex geometry.

Figure 3. 22: Deviation

It is seen that the cube, which is expected to come out 20, comes out 20.15, so the margin of error is 0.75. In total, 20 parts were produced from the 3D printer and each part has a margin of error of 0.75 as shown in **Figure 3.22**. The produced prototype spherical parallel manipulator may have a 15% margin of error in total. Due to reasons arising from the 3D printer, errors were observed in the printed parts, which creates an error margin of 1^{o}or 2^{o} between the angles entered in MATLAB and the angles measured in real. For this prototype manipulator, ∓ 1 or ∓ 2 degrees of deviation are acceptable according to 3D printer's margin of error.

CHAPTER 4

SPM MANIPULATOR KINEMATICS

The kinematics of a Spherical Parallel Manipulator (SPM) involves the study of how the manipulator's end-effector or platform moves in space based on the motions of its various components, such as the links and joints. SPMs are typically characterized by their ability to provide three degrees of freedom (3-DOF) in spatial movement.

The joint angles for the 3-DOF kinematic calculations. Each step comprises three joint angles, θ_i, φ_i and ψ_i, with its roll, pitch, and yaw axis, where "i" stands for the leg's index. Although step motors cannot be seen here, they impact the θ_i. The working range of the designed spherical parallel manipulator is between $\pm 30^o$ degrees. The angles are explained in detail in [28] seen in **Figure 4.1**.

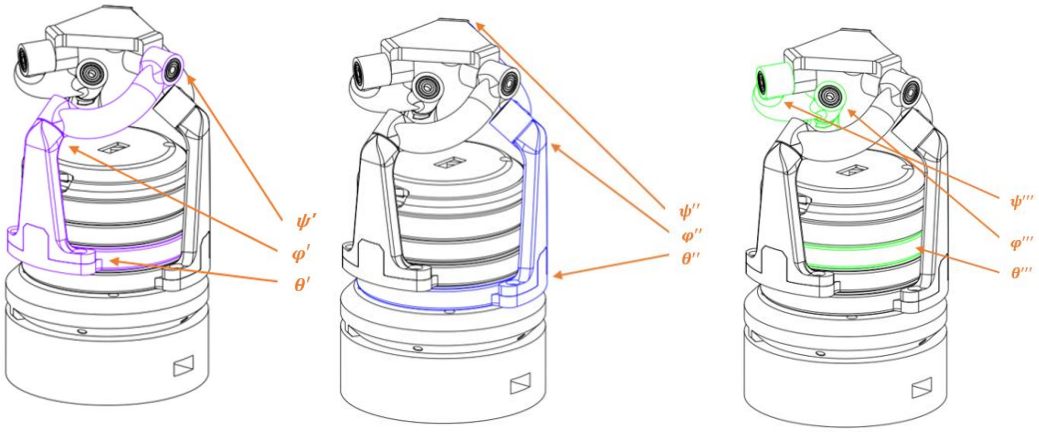


Figure 4. 1: Roll, pitch and yaw angles

4.1 Denavit Hartenberg Parameters

Denavit-Hartenberg parameters (DH parameters) are a widely used method for modeling the kinematics of robotic manipulators, including spherical parallel manipulators (SPMs). However, the application of DH parameters to SPMs is different from traditional serial manipulators due to the structure of SPMs. In the DH parameter framework, each joint is represented by a set of four parameters:

(a): Length of the common normal X_i is symbolized by the letter "a," which should not be confused with the Greek letter " α " This is the radius around the Z_i if a revolute joint is used.

(d): Z_i offset to the common normal X_i and X_{i+1}. For spherical parallel manipulator this may correspond to the distance from one spherical joint to another.

(θ): Angle from X_i to new X_{i+1}, with respect to Z_i. In an SPM, the joint angles can correspond to the angles of the spherical joints.

(α): The angle from the Z_i to the new Z_{i+1} around the common normal X_i. This parameter may not be applicable if spherical parallel manipulator spherical joints provide multi-axis rotation.

With the given information, the Denavit-Hartenberg table is shown for transition from Base to Top Plate **Table 4.1.**

Table 4.1: Denavit Hartenberg Table

Indication	Proximal	Distal	Wrist
$\alpha_{DH,j-1}$	**0**	α_P	α_D
$a_{DH,j-1}$	**0**	**0**	**0**
$d_{DH,j-1}$	**0**	**0**	**0**
$\theta_{DH,j-1}$	θ_i	φ_i	ψ_i

Where $\alpha_P = \frac{\pi}{3}$ and $\alpha_D = -\frac{\pi}{2}$ are constants [28]. The initial position of the joints, angles shown below θ_i, φ_i and ψ_i:

- $\theta_{i,0} = 0°$
- $\varphi_{i,0} = \frac{\pi}{2} = 90°$
- $\psi_{i,0} = -\frac{\pi}{6} = -30°$

4.2 Reference Frame and Initial Position

The reference frame is positioned at the midpoint of the base section so that the $\hat{X}$-component points in the direction of the main chain, the $\hat{Z}$-component points upwards toward the articular joint axis, and the $\hat{Y}$-component is selected so that the $\hat{X}, \hat{Z}$ and $\hat{Y}$ form a frame in accordance with the right hand rule [28] **Figure 4.2.**

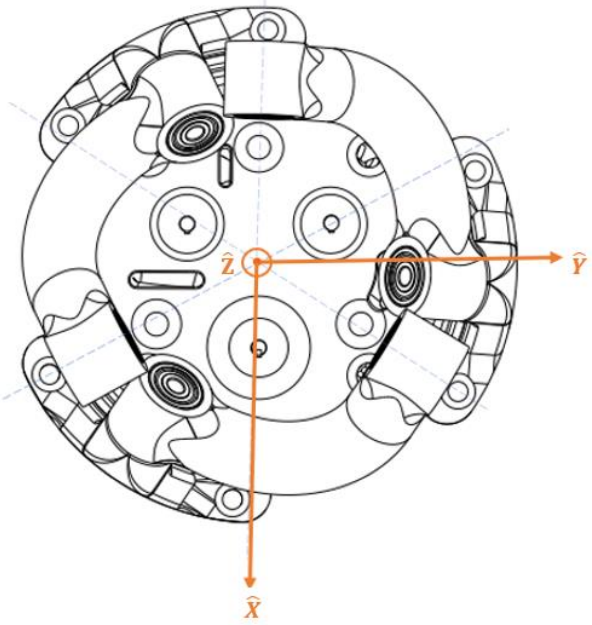


Figure 4. 2: Reference frame

Each leg needs to be at $\frac{2\pi}{3}$ =120° angle when the initial position is matched with the Reference frame orientation. [28] see **Figure 4.3**.

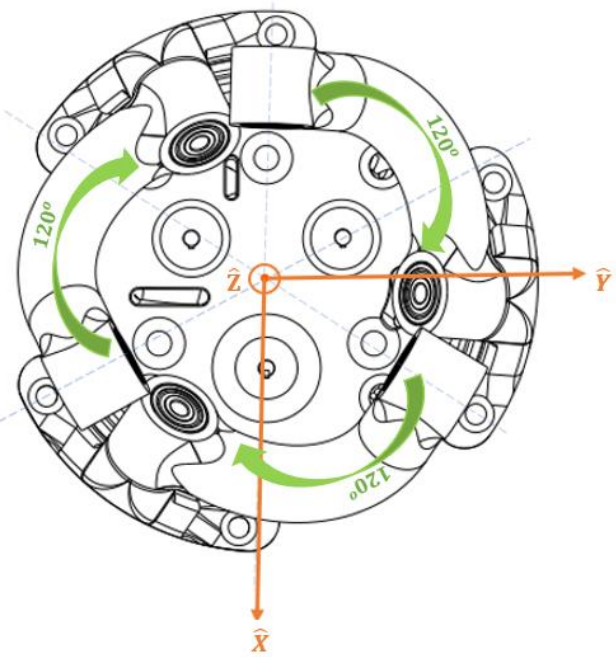


Figure 4. 3: Initial position

4.3 Forward Kinematic Analysis

Forward kinematics takes three angles as input and calculates a homogeneous matrix [29] based on these angles. Forward mapping [28] calculations based on the reference frame are shown below and depicted graphically shown in **Figure 4.4**. This orientation matrices are follows:

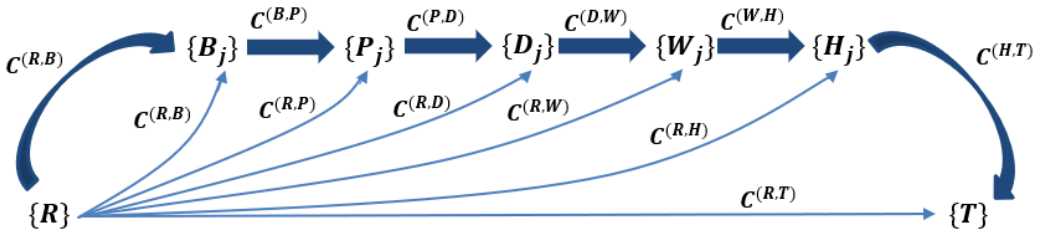


Figure 4. 4: Forward kinematics map

Transformation of Base frame to Reference frame:

$$C^{(B,R)} \tag{1}$$

Transformation of Proximal frame to Reference frame:

$$C^{(R,P)} = C^{(R,B)}C^{(B,P)} \tag{2}$$

Transformation of Distal frame to Reference frame:

$$C^{(R,D)} = C^{(R,B)}C^{(B,P)}C^{(P,D)} \tag{3}$$

Wrist to Reference frame:

$$C^{(R,W)} = C^{(R,B)}C^{(B,P)}C^{(P,D)}C^{(D,W)} \tag{4}$$

Hexagon to Reference frame:

$$C^{(R,H)} = C^{(R,B)}C^{(B,P)}C^{(P,D)}C^{(D,W)}C^{(D,H)} \tag{5}$$

Tool to Reference frame:

$$C^{(R,T)} = C^{(R,B)}C^{(B,P)}C^{(P,D)}C^{(D,W)}C^{(D,H)}C^{(H,T)} \tag{6}$$

4.4 Inverse Kinematic Analysis

Inverse kinematics can be used to predict the robot's motion to arrive at a specific place. A robotic arm used in a production line, for instance, requires accurate motion from the starting point to the goal. If the inverse kinematics orientation is known

from the Forward kinematics equations, then the equation steps can be solved with Euler angles in inverse kinematics. The calculation results of three angles of homogeneous matrix and position with respect to Inverse kinematics and the angles that entered forward kinematics as input should be matched [28].

Calculation of θ_j angle according to quaternion calculations [30] is as follows;

$$sin(a_P)\begin{bmatrix}2e_0e_3\cos(\theta_i) - 2e_1e_2\cos(2\eta_j + \theta_i) + ({e_3}^2 - {e_0}^2)\sin(\theta_i) \\ +({e_1}^2 - {e_2}^2)\sin(2\eta_i + \theta_i)\end{bmatrix} \tag{7}$$

$$+2\cos(a_P)\left[(e_0e_1 + e_2e_3)\cos(\eta_i) + (e_0e_2 - e_1e_3)\sin(\eta_i)\right] = 0$$

$$\sin(\theta_j) = \frac{2u_j}{1+{u_j}^2} \tag{8}$$

$$\cos(\theta_j) = \frac{1-{u_j}^2}{1+{u_j}^2} \tag{9}$$

$$u_j = \tan(\frac{\theta_j}{2}) \tag{10}$$

$$A_j{u_j}^2 + B_ju_j + C_j = 0 \tag{11}$$

$$A_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{12}$$

$$B_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{13}$$

$$C_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{14}$$

$$u_{j,1/2} = \frac{-B_j \pm \sqrt{{B_j}^2 - 4A_jC_j}}{2A_j} \tag{15}$$

Since αD is a mechanical constant, its value is αD=-90° due to the construction and the Denavit-Hartenberg notation. Consequently, the statement can be simplified even further by multiplying it by the term $(1 + u^2)$ [28] and setting $\cos(\alpha D) =$

$\cos(-90°) = 0$. The quadratic equation can be expressed in terms of u to identify the variables A, B, and C in the equation $Au^2 + Bu + C = 0$.

The parameters A, B and C from the quadratic equation $Au^2 + Bu + C = 0$ can be determined:

Parameter A,

$$2(-e_0e_3\sin[\alpha P] + e_1e_2\cos[2\eta_i]\sin[\alpha P] + \cos[\alpha P]\left((e_0e_1 + e_2e_3)\cos[\eta_i] + (e_0e_2 - e_1e_3)\sin[\eta_i]\right) - \frac{1}{2}e_1^2\sin[\alpha P]\sin[2\eta_i] + \frac{1}{2}e_2^2\sin[\alpha P]\sin[2\eta_i]) \quad (16)$$

Parameter B,

$$2(-e_0^2\sin[\alpha P] + e_3^2\sin[\alpha P] + e_1^2\cos[2\eta_i]\sin[\alpha P] - e_2^2\cos[2\eta_i]\sin[\alpha P] + 2e_1e_2\sin[\alpha P]\sin[2\eta_i]) \quad (17)$$

Parameter C,

$$2(e_0e_3\sin[\alpha P] - e_1e_2\cos[2\eta_i]\sin[\alpha P] + \cos[\alpha P]\left((e_0e_1 + e_2e_3)\cos[\eta_i] + (e_0e_2 - e_1e_3)\sin[\eta_i]\right) + \frac{1}{2}e_1^2\sin[\alpha P]\sin[2\eta_i] - \frac{1}{2}e_2^2\sin[\alpha P]\sin[2\eta_i]) \quad (18)$$

Both answers to θ_j can be computed using the expression (15).

$$\theta_{j,1} = Atan2\left(\frac{1-{u_{j,1}}^2}{1+{u_{j,1}}^2}, \frac{2u_{j,1}}{1+{u_{j,1}}^2}\right) \quad (19)$$

$$\theta_{j,2} = Atan2\left(\frac{1-{u_{j,2}}^2}{1+{u_{j,2}}^2}, \frac{2u_{j,2}}{1+{u_{j,2}}^2}\right) \quad (20)$$

Calculation of φ_j angle according to quaternion calculations [30] is as follows;

$$\sin\left(\varphi_j\right) = \frac{2t_j}{1+{t_j}^2} \quad (21)$$

$$\cos\left(\varphi_j\right) = \frac{1-{t_j}^2}{1+{t_j}^2} \quad (22)$$

$$t_j = \tan(\frac{\varphi_j}{2}) \quad (23)$$

$$D_j t_j{}^2 + E_j t_j + F_j = 0 \tag{24}$$

$$D_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{25}$$

$$E_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{26}$$

$$F_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \tag{27}$$

The mechanical constant Cos[D]=Cos [-90°]=0 is then equal to the dot product of the normal vector and $Y\left(C^{(R,D)}\right)$ [28]. It is clear that the angle I is expressed in terms of Sin [φ_i] and Cos [φ_i], which are then replaced by the variable t using the Weierstrass substitution (appendix I). This substitution leads to a quadratic equation with coefficients D, E, and F = f (e_0, e_1, e_2, e_3,B,P, I) where $Dt^2 + Et + F = 0$ is the solution.

Parameter D,

$$-e_0^2 \cos[\theta_\text{i}] + e_3^2 \cos[\theta_\text{i}] + e_1^2 \cos[2\eta_\text{i} + \theta_\text{i}] - e_2^2 \cos[2\eta_\text{i} + \theta_\text{i}] - 2e_0 e_3 \sin[\theta_\text{i}] + 2e_1 e_2 \sin[2\eta_\text{i} + \theta_\text{i}] \tag{28}$$

Parameter E,

$$-2(e_0 e_2 - e_1 e_3)\cos[\alpha P - \eta_\text{i}] + 2(e_0 e_2 - e_1 e_3)\cos[\alpha P + \eta_\text{i}] + 2e_0 e_3 \cos[\alpha P - \theta_\text{i}] - 2e_1 e_2 \cos[\alpha P - 2\eta_\text{i} - \theta_\text{i}] + 2e_0 e_3 \cos[\alpha P + \theta_\text{i}] - 2e_1 e_2 \cos[\alpha P + 2\eta_\text{i} + \theta_\text{i}] - 2e_0 e_1 \sin[\alpha P - \eta_\text{i}] - 2e_2 e_3 \sin[\alpha P - \eta_\text{i}] - 2e_0 e_1 \sin[\alpha P + \eta_\text{i}] - 2e_2 e_3 \sin[\alpha P + \eta_\text{i}] + e_0^2 \sin[\alpha P - \theta_\text{i}] - e_3^2 \sin[\alpha P - \theta_\text{i}] - e_1^2 sin[\alpha P - 2\eta_\text{i} - \theta_\text{i}] + e_2^2 sin[\alpha P - 2\eta_\text{i} - \theta_\text{i}] - e_0^2 \sin[\alpha P + \theta_\text{i}] + e_3^2 \sin[\alpha P + \theta_\text{i}] + e_1^2 sin[\alpha P + 2\eta_\text{i} + \theta_\text{i}] - e_2^2 sin[\alpha P + 2\eta_\text{i} + \theta_\text{i}] \tag{29}$$

Parameter F,

$$e_0^2 \cos[\theta_\text{i}] - e_3^2 \cos[\theta_\text{i}] - e_1^2 \cos[2\eta_\text{i} + \theta_\text{i}] + e_2^2 \cos[2\eta_\text{i} + \theta_\text{i}] + 2e_0 e_3 \sin[\theta_\text{i}] - 2e_1 e_2 \sin[2\eta_\text{i} + \theta_\text{i}] \tag{30}$$

$$t_{j,1/2} = \frac{-E_j \pm \sqrt{E_j{}^2 - 4D_j F_j}}{2D_j} \tag{31}$$

Both answers to φ_j can be computed using the expression (31).

$$\varphi_{j,1} = Atan2\left(\frac{1-{t_{j,1}}^2}{1+{t_{j,1}}^2}, \frac{2t_{j,1}}{1+{t_{j,1}}^2}\right) \quad (32)$$

$$\varphi_{j,2} = Atan2\left(\frac{1-{t_{j,2}}^2}{1+{t_{j,2}}^2}, \frac{2t_{j,2}}{1+{t_{j,2}}^2}\right) \quad (33)$$

Calculation of ψ_j angle according to quaternion calculations [30] is as follows;

$$\sin\left(\psi_j\right) = \frac{2s_j}{1+{s_j}^2} \quad (34)$$

$$\cos\left(\psi_j\right) = \frac{1-{s_j}^2}{1+{s_j}^2} \quad (35)$$

$$s_j = \tan(\frac{\psi_j}{2}) \quad (36)$$

$$G_j {s_j}^2 + H_j s_j + I_j = 0 \quad (37)$$

$$G_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \quad (38)$$

$$H_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \quad (39)$$

$$I_j(e_0, e_1, e_2, e_3, \eta_i, \alpha_B, \alpha_P) \quad (40)$$

The dot product is equal to 0 because the vectors $C^{(R,W)}$ and $C^{(R_2,W)}$ [28] are perpendicular to each other. It is clear that the angle i is expressed in terms of Sin [ψ_i] and Cos[ψ_i], which are then replaced by the variable s using the Weierstrass substitution.

Parameter G,

$$\begin{aligned}&(e_0^2\cos[\alpha P] - e_1^2\cos[\alpha P] - e_2^2\cos[\alpha P] + e_3^2\cos[\alpha P] - 2(e_0e_1 + \\ &e_2e_3)\cos[\eta_i]\cos[\theta_i]\sin[\alpha P] - 2(-e_0e_1 + e_2e_3)\cos[\theta_i] + \\ &2e_0e_2\sin[\alpha P]\sin[\eta_i] - 2(-e_0e_1 + e_2e_3)\cos[\eta_i]\sin[\alpha P]\sin[\theta_i] + \\ &2e_0e_2\sin[\alpha P]\sin[\eta_i]\sin[\theta_i] + 2e_1e_3\sin[\alpha P]\sin[\eta_i]\sin[\theta_i])\end{aligned} \quad (41)$$

Parameter H,

$$-4(e_0e_1 - e_2e_3)\cos[\eta_i]\cos[\theta_i]\cos[\varphi_i] - 4(e_0e_1 + e_2e_3)\cos[\eta_i]\cos[\theta_i]\sin[\theta_i] + 4e_0e_1\cos[\varphi_i]\sin[\eta_i]\sin[\theta_i] - 4e_2e_3\cos[\varphi_i]\sin[\eta_i]\sin[\theta_i] - 2e_3^2\sin[\alpha P]\sin[\varphi_i] + 2e_1^2\sin[\alpha P]\sin[\varphi_i] + 2e_2^2\sin[\alpha P]\sin[\varphi_i] + 2e_3^2\sin[\alpha P]\sin[\varphi_i] + 4(e_0e_1 - e_2e_3)\cos[\alpha P]\cos[\theta_i]\sin[\eta_i]\sin[\varphi_i] + 4e_0e_2\cos[\alpha P]\sin[\eta_i]\sin[\theta_i]\sin[\varphi_i] + 4e_1e_3\cos[\alpha P]\sin[\eta_i]\sin[\theta_i]\sin[\varphi_i] - 4\cos[\alpha P]\cos[\eta_i]\left((e_0e_1 + e_2e_3)\cos[\theta_i] + (-e_0e_1 + e_2e_3)\sin[\varphi_i]\right) \quad (42)$$

Parameter I,

$$-e_0^2\cos[\alpha P] + e_1^2\cos[\alpha P] + e_2^2\cos[\alpha P] - e_3^2\cos[\alpha P] - 2(-e_0e_1 - e_2e_3)\cos[\eta_i]\cos[\theta_i]\sin[\alpha P] - 2(e_0e_1 - e_2e_3)\cos[\eta_i]\sin[\alpha P]\sin[\theta_i] - 2e_0e_2\sin[\alpha P]\sin[\eta_i]\sin[\theta_i] - 2e_1e_3\sin[\alpha P]\sin[\eta_i]\sin[\theta_i] \quad (43)$$

$$s_{j,1/2} = \frac{-H_j \pm \sqrt{{H_j}^2 - 4G_jI_j}}{2G_j} \quad (44)$$

Both answers to ψ_j can be computed using the expression (44).

$$\psi_{j,1} = Atan2\left(\frac{1-{s_{j,1}}^2}{1+{s_{j,1}}^2}, \frac{2s_{j,1}}{1+{s_{j,1}}^2}\right) \quad (45)$$

$$\psi_{j,2} = Atan2\left(\frac{1-{s_{j,2}}^2}{1+{s_{j,2}}^2}, \frac{2s_{j,2}}{1+{s_{j,2}}^2}\right) \quad (46)$$

According to [28] Weierstrass method was used to solve the angles. You can see the solutions below.

To determine $\boldsymbol{\theta_j}$;

$$\theta_j = Atan2(cos\theta_j, sin\theta_j) \quad (47)$$

Has two options according to

$$\theta_{j,1} = Atan2\left(\frac{1-{u_{j,1}}^2}{1+{u_{j,1}}^2}, \frac{2u_{j,1}}{1+{u_{j,1}}^2}\right) \quad (48)$$

$$\theta_{j,2} = Atan2\left(\frac{1-{u_{j,2}}^2}{1+{u_{j,2}}^2}, \frac{2u_{j,2}}{1+{u_{j,2}}^2}\right) \quad (49)$$

To determine $\boldsymbol{\varphi_j}$;

$$\varphi_j = Atan2(cos\varphi_j, sin\varphi_j) \quad (50)$$

Has two options according to

$$\varphi_{j,1} = Atan2\left(\frac{1-{t_{j,1}}^2}{1+{t_{j,1}}^2}, \frac{2t_{j,1}}{1+{t_{j,1}}^2}\right) \quad (51)$$

$$\varphi_{j,2} = Atan2\left(\frac{1-{t_{j,2}}^2}{1+{t_{j,2}}^2}, \frac{2t_{j,2}}{1+{t_{j,2}}^2}\right) \quad (52)$$

To determine $\boldsymbol{\psi_j}$;

$$\psi_j = Atan2(cos\psi_j, sin\psi_j) \quad (53)$$

Has two options according to

$$\psi_{j,1} = Atan2\left(\frac{1-{s_{j,1}}^2}{1+{s_{j,1}}^2}, \frac{2s_{j,1}}{1+{s_{j,1}}^2}\right) \quad (54)$$

$$\psi_{j,2} = Atan2\left(\frac{1-{s_{j,2}}^2}{1+{s_{j,2}}^2}, \frac{2s_{j,2}}{1+{s_{j,2}}^2}\right) \quad (55)$$

CHAPTER 5

CONTROL

MATLAB/Simscape serves as modeling software for mechanical systems and offers a unified simulation environment for creating both mechanical and controller models. SimScape also provides exporting a computer-aided design (CAD) to MATLAB, simplifying the automatic generation of SimScape models from SolidWorks or CAD assemblies. It allows the development of physical component models based on physical connections. SimScape greatly assists in the development of control systems.

In the thesis on the spherical parallel manipulator, attention is given to the simulation and modeling of the manipulator. In this thesis, SolidWorks, Catia v5, SimScape, Simulink, and MATLAB were utilized for the design and analysis of the spherical parallel manipulator. SimScape facilitated the rapid creation of physical system models within the MATLAB environment. Consequently, the choice to use this software is driven by the aim of achieving a one-to-one correspondence between the manipulator's simulation and real-life performance. Moreover, its compatibility with SolidWorks, a solid modeling program, streamlines the data transfer process.

Figure 5. 1: SPM simulation

To simulate a spherical parallel manipulator in MATLAB [31] shown in **Figure 5.1**, following steps need to be proceeded:

1. Creating a mathematical representation of the manipulator, which involves using equations, such as the forward and inverse kinematics equations, to specify the kinematics of the manipulator.
2. The Simulink toolbox, which enables to build models of systems and processes using blocks and diagrams, would be used to construct a simulation environment in MATLAB.
3. The simulation environment should be used to implement the mathematical model. In order to do this, the simulation environment made in Step 2 would need to include the equations and algorithms generated in Step 1.
4. The manipulator is simulated in Simulink and the data received as a result of the simulation can be examined.

5. Visualize and evaluate the results: MATLAB's visualization and analysis capabilities can be used to evaluate the simulations' outcomes. These options include graph plotting, statistical analysis, and animation creation.

5.1 Spherical Parallel Manipulator SimScape Model

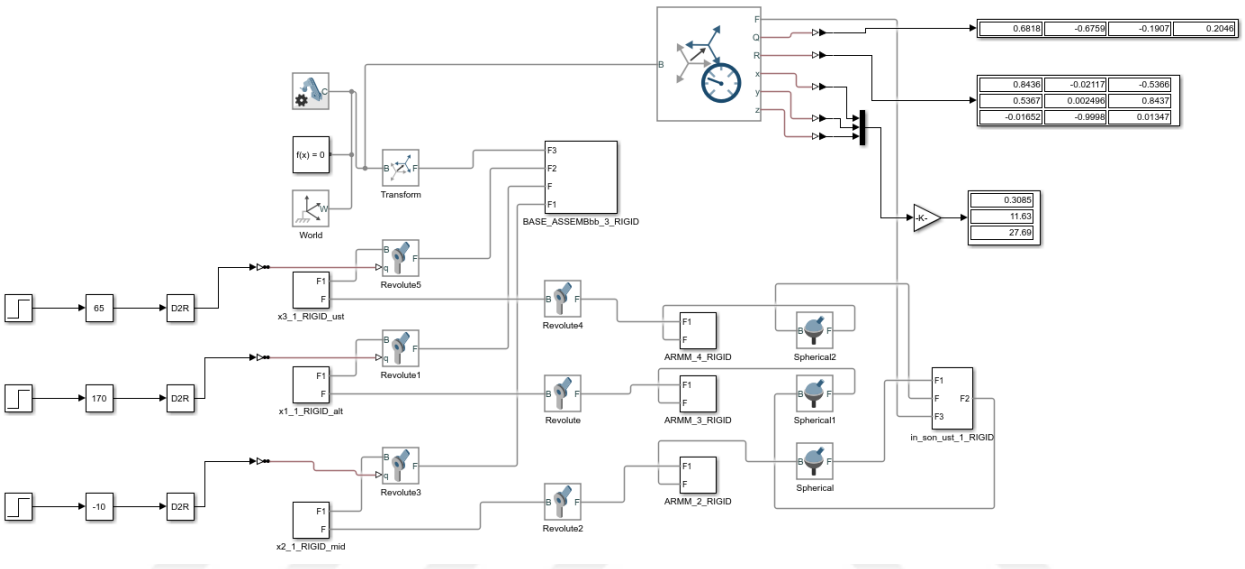


Figure 5. 2: SPM simulation blocks

The Spherical parallel manipulator model exported from Solidworks to Simscape. A detailed explanation of how to export the CAD model from Solidworks to SimScape is stated in **appendix F**. Simscape blocks shown in **Figure 5.2.**

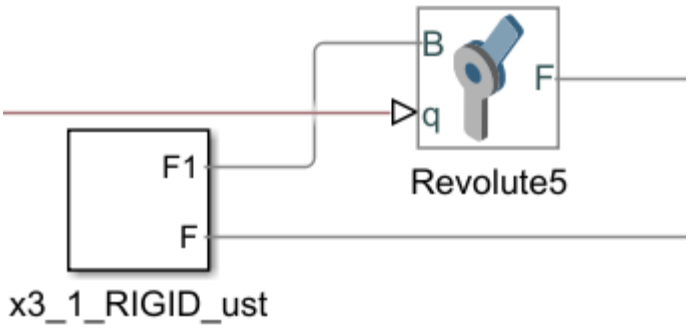


Figure 5. 3: Revolute joint

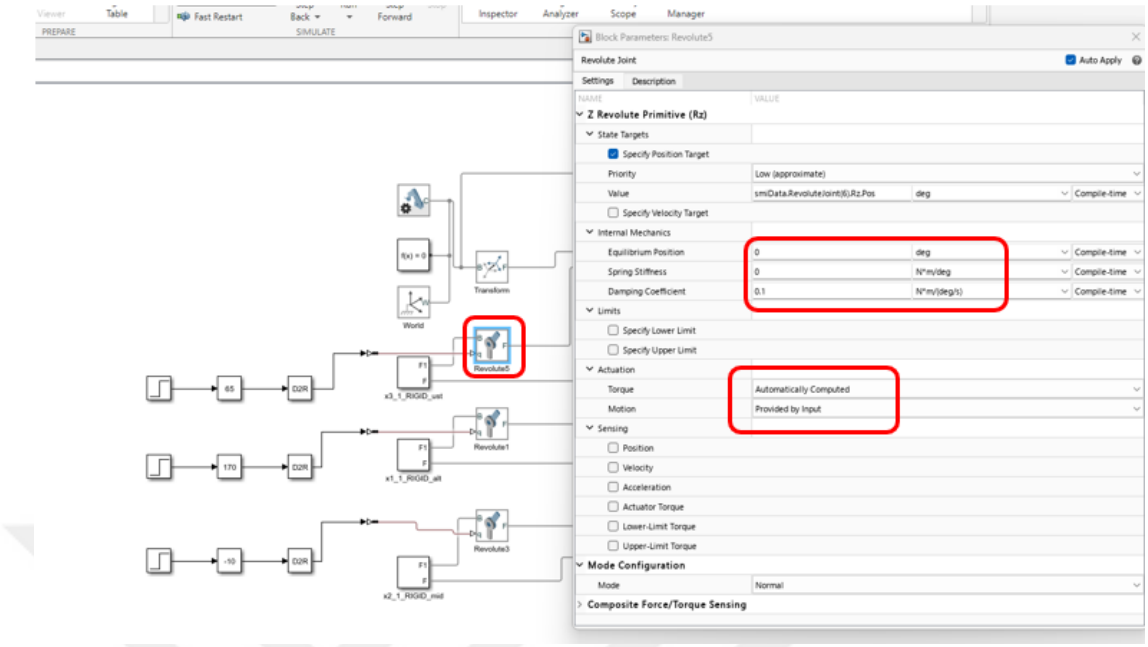


Figure 5. 4: Revolute joint settings

To set motion limits to restrict the range of motion, the Torque needs to be automatically computed and the motion section needs to be provided by input. This can be useful for modeling realistic mechanical systems with physical constraints.

When the CAD file was exported from SolidWorks to MATLAB, the damping coefficient was entered as 0.1 to prevent the simulation movement rotating differently from the desired rotation.

Equilibrium position refers to the position in a system that is in equilibrium or at rest. In this state, all forces and moments should be zero, so equilibrium position 0 is entered in the simulation. Since there is no spring stiffness value in the system, 0 is entered.

5.2 Reference Frames

In Simscape, the orientation and location of various components inside a physical model are specified using a reference frame, often known as a local reference frame or a local coordinate system. Reference frames are necessary to comprehend the

connections and motions between different parts of a mechanical or multi-domain system.

- Bodies, joints, and sensors of every Simscape model have their own special local coordinate system or reference frame. When creating this local coordinate system, the component's shape and orientation were taken into consideration.
- Simscape models usually have a global coordinate system that serves as a point of reference for all the model's components. Model components may be positioned and orientated consistently with the assistance of this global coordinate system.
- Joint Motion: It was discussed how the motion of a joint, such as a revolute joint or a prismatic joint, affects the relative transformation between connected components. Joint motion is important for accurate observation of the mechanical behavior of the system.

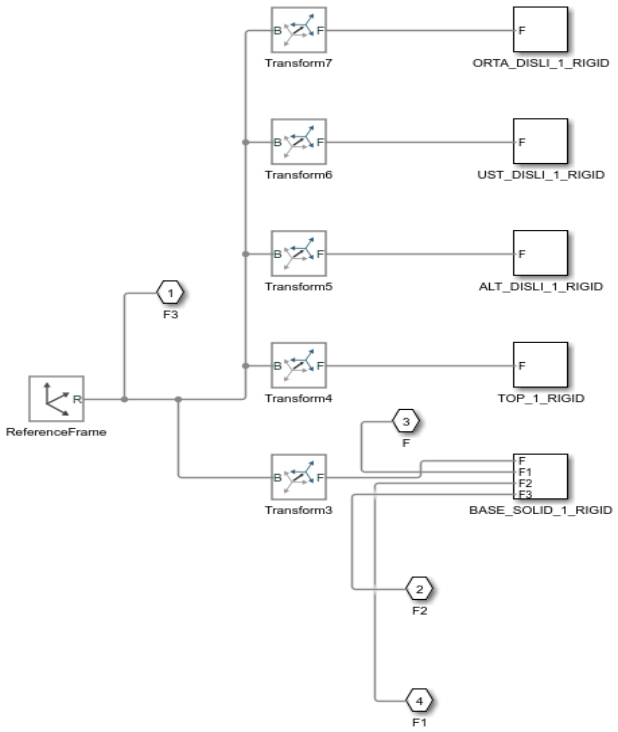


Figure 5. 5: Subsystem of middle body part and base

The subsystem of where the middle body part and the base are assembled is shown in **Figure 5.5**.

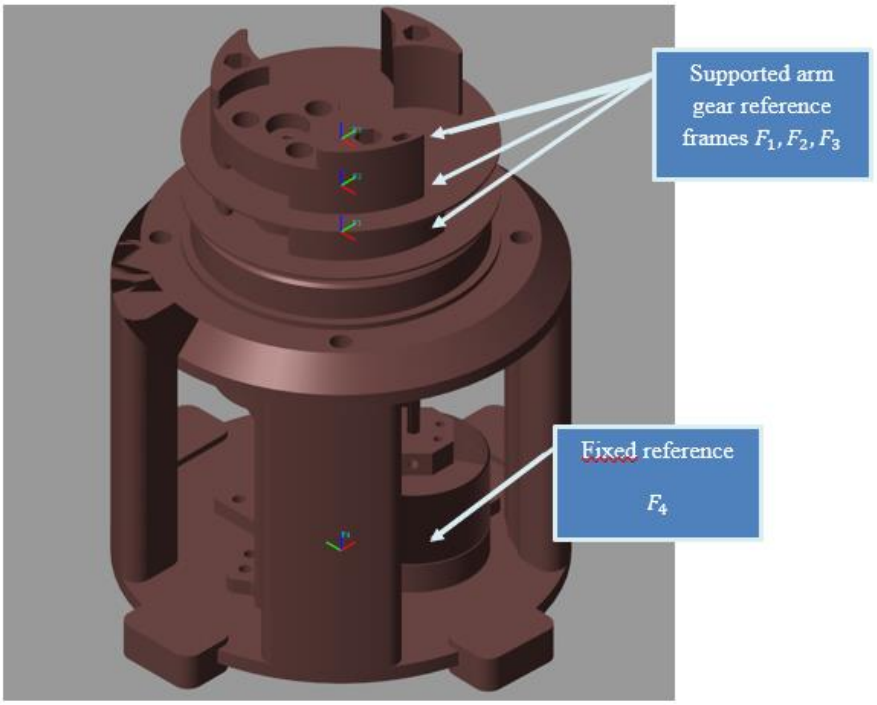


Figure 5. 6: References shown on simulation

Four reference frames are defined in the subsystem. The first one is the reference frame given F_4 to fix it the base on the ground. The other three F_1, F_2, F_3 are supported arm gear reference frames.

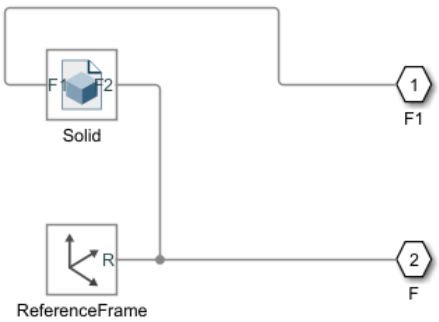


Figure 5. 7: Subsystem of supported arm gear

The subsystem of the supported arm gear part is shown in **Figure 5.7**.

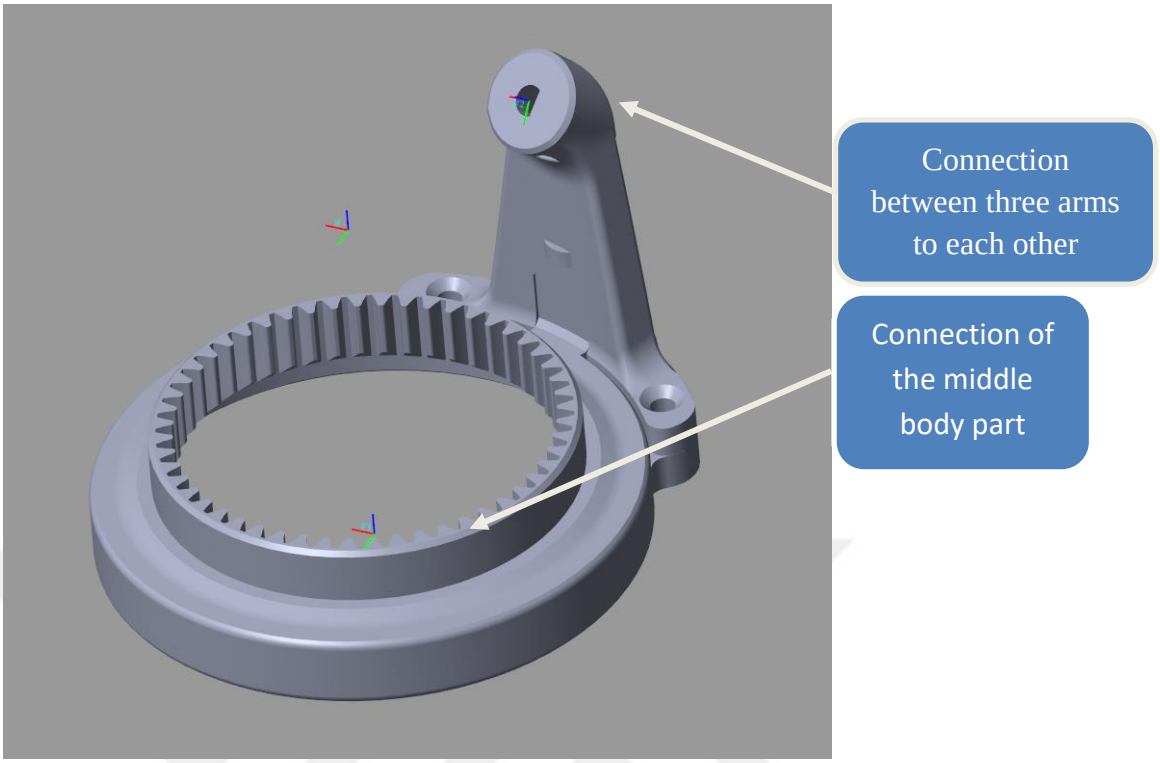


Figure 5. 8: Reference frame connections of supported arm gear

Each supported arm gear F_1 is connected to the reference frames in the base. Also, in supported arm gear F_2 connected on the arms.

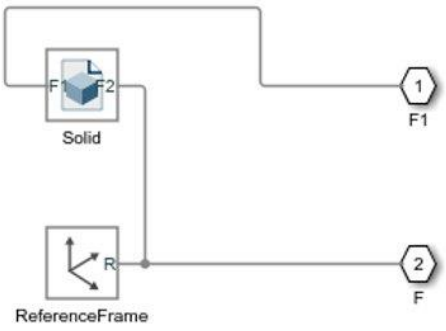


Figure 5. 9: Subsystem of the arm

The subsystem of the arm part is shown in **Figure 5.9**.

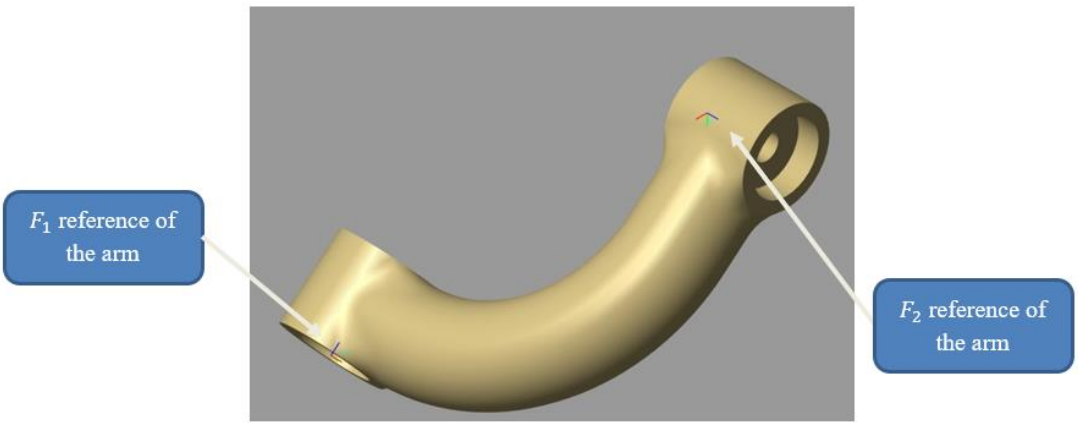


Figure 5. 10: Reference frames of arm

The reference of the supported arm gear F_1 and the reference of the arm F_2 are connected.

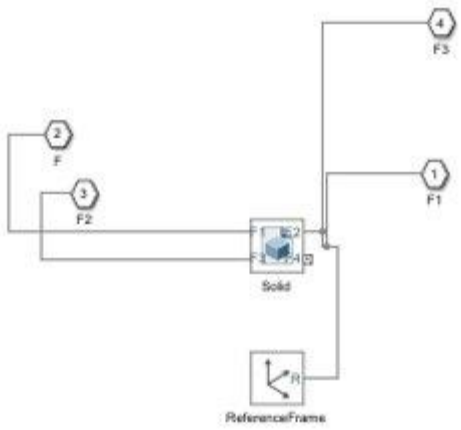


Figure 5. 11: Subsystem of top plate

The subsystem of the top plate part is shown in **Figure 5.11**.

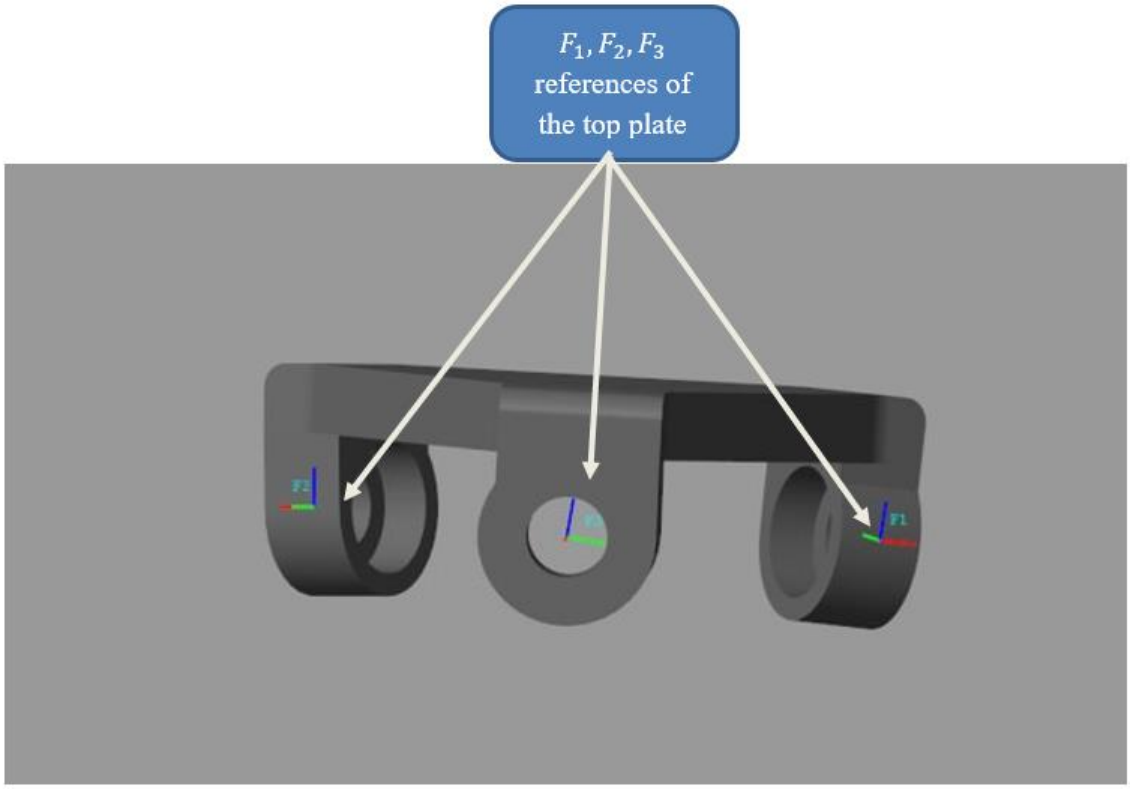


Figure 5. 12: Reference frames of top plate

The references of each arm F_2 and the references of the top plate F_1, F_2, F_3 are connected to each other.

5.3 Transform Sensor

The Transform sensor was connected between the F and B ports. It shows the calculation results of time-varying values, such as quaternion matrix, position and homogeneous matrix.

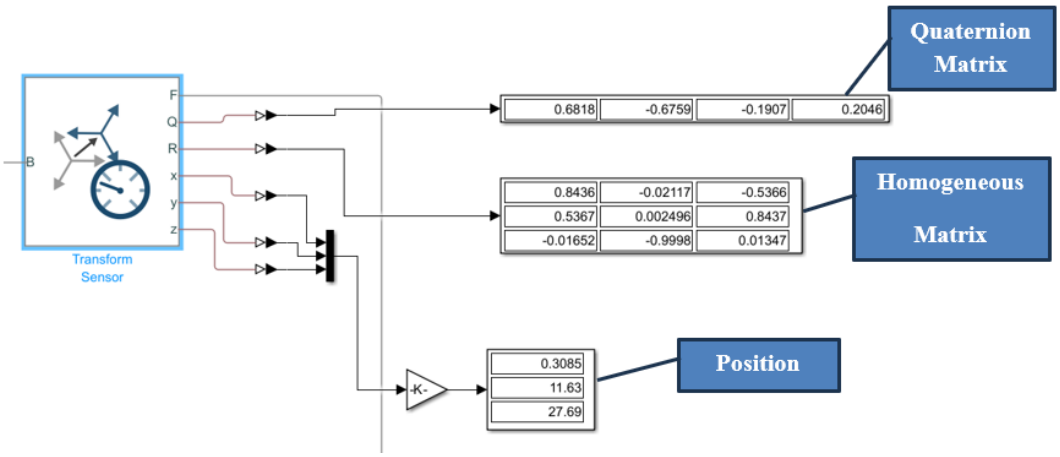


Figure 5. 13: Transform sensor outputs

In the thesis, MATLAB was used to implement the mathematical formulas for the forward and inverse kinematics computations. MATLAB and Simulink were chosen for this project because they are easy to use, can solve complex equations easily, and have high simulation capacity.

The written code was compared and tested via Simulink. According to the calculations, the 3 input values given and the simulation gives the exact same values as the output. Therefore, the forward and inverse kinematics calculations included in MATLAB functions are entirely accurate. The calculations of the motor input angle, as observed in the output values, also verify their correctness when examining the general equations as shown in **Figure 5.14**.

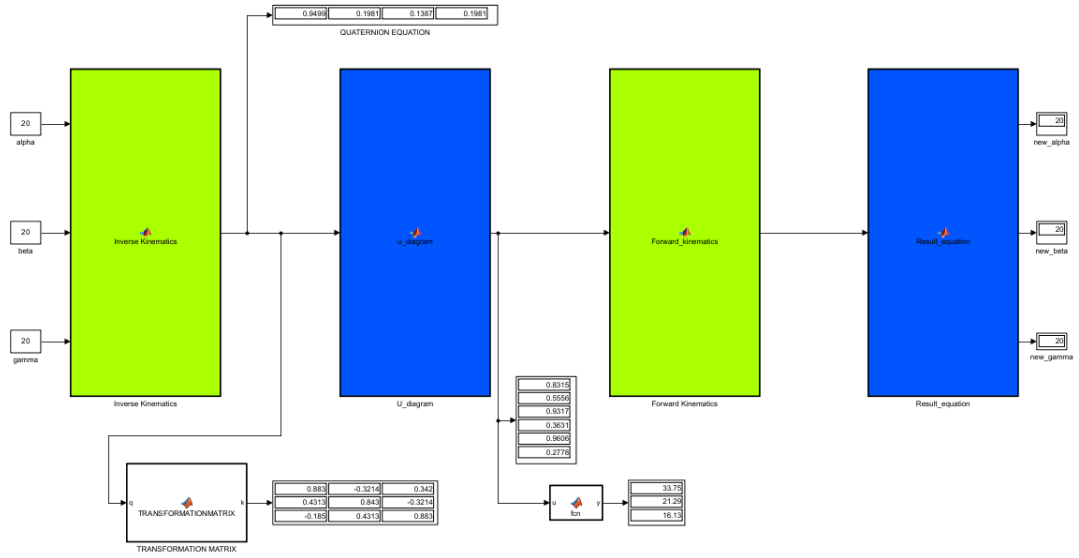


Figure 5. 14: SPM-Simulation blocks

Quaternion calculations [28] of alpha beta and gamma written in the MATLAB function are shown in **Figure 5.15**. The kinematic calculations were examined in detail, and some parameters were changed (link length, etc.) for the optimized prototype SPM and were adapted for the prototype SPM. Detailed codes can be found in **appendix H.**

```
function q = TBzyx2q( alpha, beta, gamma )
%------------- BEGIN CODE --------------
alpha = alpha*(pi/180);
beta = beta*(pi/180);
gamma = gamma*(pi/180);

e0 = cos(alpha/2)*cos(beta/2)*cos(gamma/2) - sin(alpha/2)*sin(beta/2)*sin(gamma/2);
e1 = cos(alpha/2)*cos(beta/2)*sin(gamma/2) + cos(gamma/2)*sin(alpha/2)*sin(beta/2);
e2 = cos(alpha/2)*cos(gamma/2)*sin(beta/2) - cos(beta/2)*sin(alpha/2)*sin(gamma/2);
e3 = cos(beta/2)*cos(gamma/2)*sin(alpha/2) + cos(alpha/2)*sin(beta/2)*sin(gamma/2);

% T01 = [cos(Alpha) sin(Alpha) 0 0 ; -sin(Alpha) cos(Alpha) 0 0 ; 0 0 1 0 ; 0 0 0 1];
% T12 = [cos(Beta) 0 -sin(Beta) 0 ; 0 1 0 0 ; sin(Beta) 0 cos(Beta) 0 ; 0 0 0 1];
% T23 = [cos(Gamma) sin(Gamma) 0 0 ; -sin(Gamma) cos(Gamma) 0 0 ; 0 0 1 0 ; 0 0 0 1];
% T03 = T01*T12*T23;
%
% C01=[cos(Alpha) sin(Alpha) 0;-sin(Alpha) cos(Alpha) 0;0 0 1];
% C12=[cos(Beta) 0 -sin(Beta);0 1 0;sin(Beta) 0 cos(Beta)];
% C23=[cos(Gamma) sin(Gamma) 0;-sin(Gamma) cos(Gamma) 0;0 0 1];
% C03=C01*C12*C23;

q = [e0,e1,e2,e3];
end
%------------- END OF CODE --------------
```

Figure 5. 15: Quaternion calculations in MATLAB

Inverse kinematic calculations [28] were made with quaternion inputs, the MATLAB function shown in **Figure 5.15**. For detailed information see a**ppendix H**.

```
function u  = SPM_invKin_pos(q)

%% Mechanical constants
alphaB = -pi/2;
alphaP = 65*pi/180;
alphaD = -pi/2;
eta = [0 , 2*pi/3 , -2*pi/3];
epsMax = 67*pi/180;

e0 = q(1);
e1 = q(2);
e2 = q(3);
e3 = q(4);

M=zeros(3);
N=zeros(3);
P=zeros(3);
c=zeros(3);
s=zeros(3);
%% Loop for the three kinematic chains
for i = 1:3
    %% Calculate Constant parameters
    M(i)=sin(alphaP)*(2*e0*e3-2*e1*e2*cos(2*eta(i))+(e1^2-e2^2)*sin(2*eta(i)));
    N(i)=sin(alphaP)*(-e0^2+e3^2+(e1^2-e2^2)*cos(2*eta(i))+2*e1*e2*sin(2*eta(i)));
    P(i)=2*cos(alphaP)*((e0*e1+e2*e3)*cos(eta(i))+(e0*e2-e1*e3)*sin(eta(i)));

    %% Solve quadratic equation (Roots positive is the correct answer)
    s(i)=(+sqrt(M(i)^2+N(i)^2-P(i)^2)*M(i)-P(i)*N(i))/(M(i)^2+N(i)^2);
    c(i)=(-sqrt(M(i)^2+N(i)^2-P(i)^2)*N(i)-P(i)*M(i))/(M(i)^2+N(i)^2);
end
%% Affect output
u = [c(1),s(1),c(2),s(2),c(3),s(3)];
```

Figure 5. 16: Forward kinematics calculations MATLAB code

A small part of forward kinematic calculations is shown in **Figure 5.16**. For detailed information see **appendix H.**

```
function q  = SPM_dirKin_pos(u)
  coder.extrinsic('quatexp')
%% Load Constants
iMax = 10;
ikMax = 4;
kini = 1;
eps = 5.9921e-06;
dk = 0.5;

%% Create Vectors
k = kini;
i = 0;
ik = 0;
ue = u;
ud = zeros(1,6);
du = zeros(1,6);
qip1 = zeros(1,4);

%% Create Initial Conditions
% Calculate mean angle off actuators

cmean = (u(1)+u(3)+u(5));
smean = (u(2)+u(4)+u(6));

divnorm = 1/sqrt(smean^2+cmean^2);

smean = smean*divnorm;
cmean = cmean*divnorm;
```

Figure 5. 17: Quatexp MATLAB function

The "quatexp" function or block is not a built-in feature or block in Simulink by default. If quaternion exponentiation is calculated in Simulink, a designed custom block or MATLAB Function blocks need to be used.

Quaternion exponentiation is the mathematical action of increasing a quaternion to a power, which may be performed in Simulink by building a custom MATLAB function. An example shown in **Figure 5.18**.

```matlab
% Define a quaternion
q = quaternion(1, 2, 3, 4); % Example quaternion

% Exponentiate the quaternion
q_exp = exp(q);
```

Figure 5. 18: Defining quaternion function

The final MATLAB function resulting from the kinematic calculations is shown in **Figure 5.19**.

```matlab
function [ alpha, beta, gamma ] = q2TBzyx(q)
e0 = q(1);
e1 = q(2);
e2 = q(3);
e3 = q(4);

alpha = atan2(2*(e0*e3-e1*e2),1-2*(e2^2+e3^2));
beta = asin(2*(e0*e2+e1*e3));
gamma = atan2(2*(e0*e1-e2*e3),1-2*(e1^2+e2^2));

alpha = alpha*(180/pi);
beta = beta*(180/pi);
gamma = gamma*(180/pi);
end
```

Figure 5. 19: Resulting function

5.4 Electrical Diagram of the Spherical Parallel Manipulator

The spherical parallel manipulator (SPM) electrical diagram shows the electrical connections and components used to enable the movement of the SPM. The components used in the prototype may vary in the future works. The power supply, control circuit and step motors required to move the joints can be seen in the diagram in **Figure 5.20**.

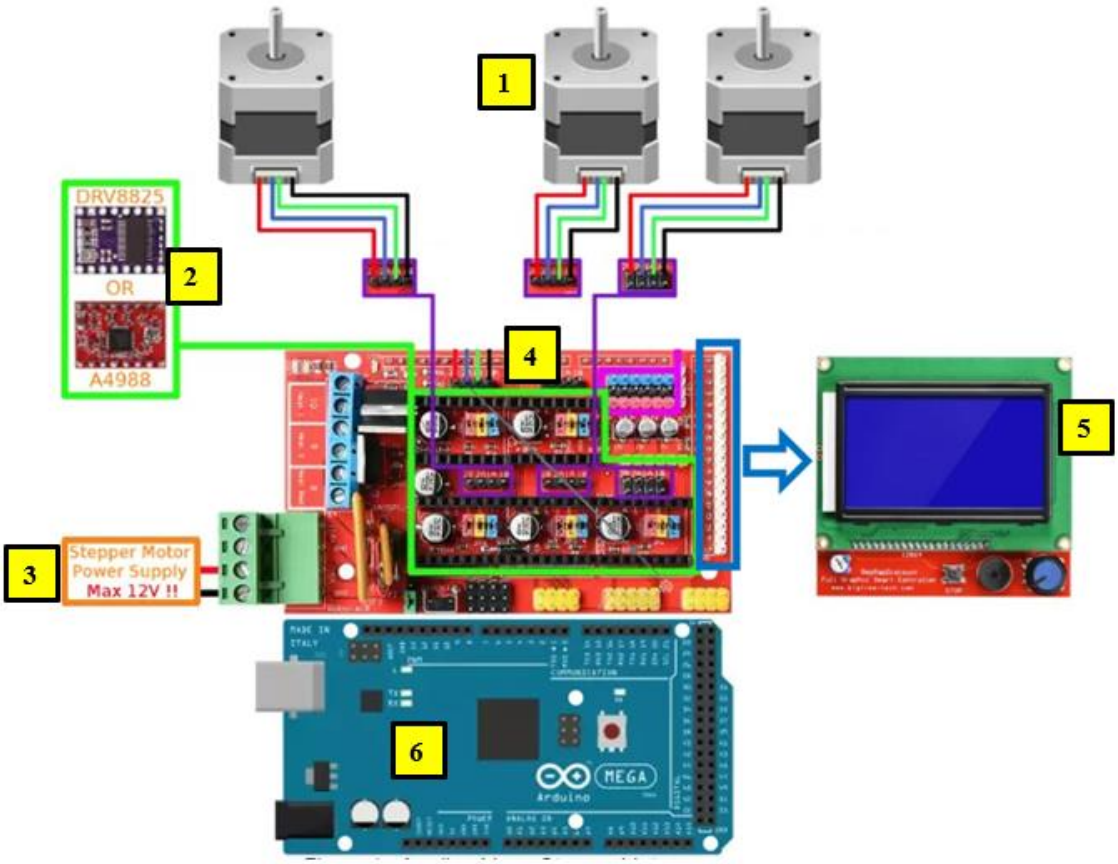


Figure 5. 20: Electric Diagram of SPM

Table 5.1: Components

Schematic Number	Used Electrical Parts
1	Stepper Motor Nema 17
2	A4988 Stepper Motor Drivers
3	Power Supply
4	RepRap Ramps 1.4
5	LCD Screen
6	Arduino Mega
7	Marlin Software

5.5 Modified Sections of Marlin Software for Spherical Parallel Manipulator

The motor drivers used have been selected in the Configuration.h [33] files, shown in **Figure 5.21** and detailed code explained in **appendix I.1**.

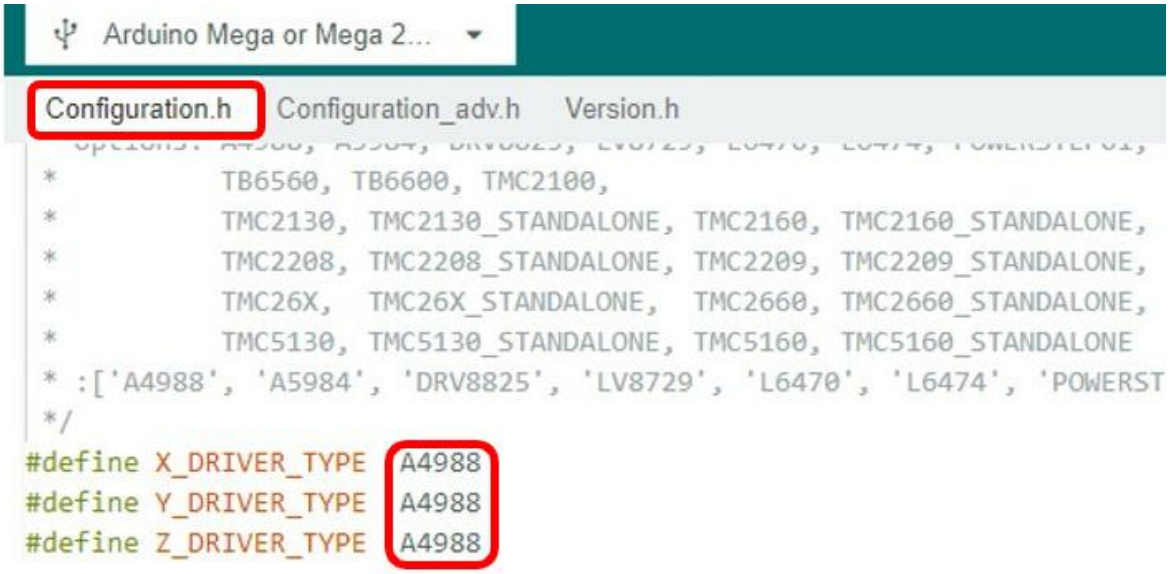


Figure 5. 21: Stepper code insertion on Marlin software

Since the temperature sensor was not used, it was written as 0 in the Configuration.h files, shown in **Figure 5.22** and detailed code explained in **appendix I.2.**

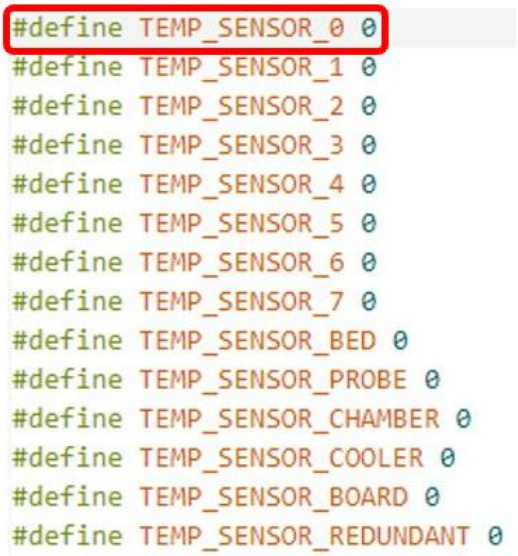


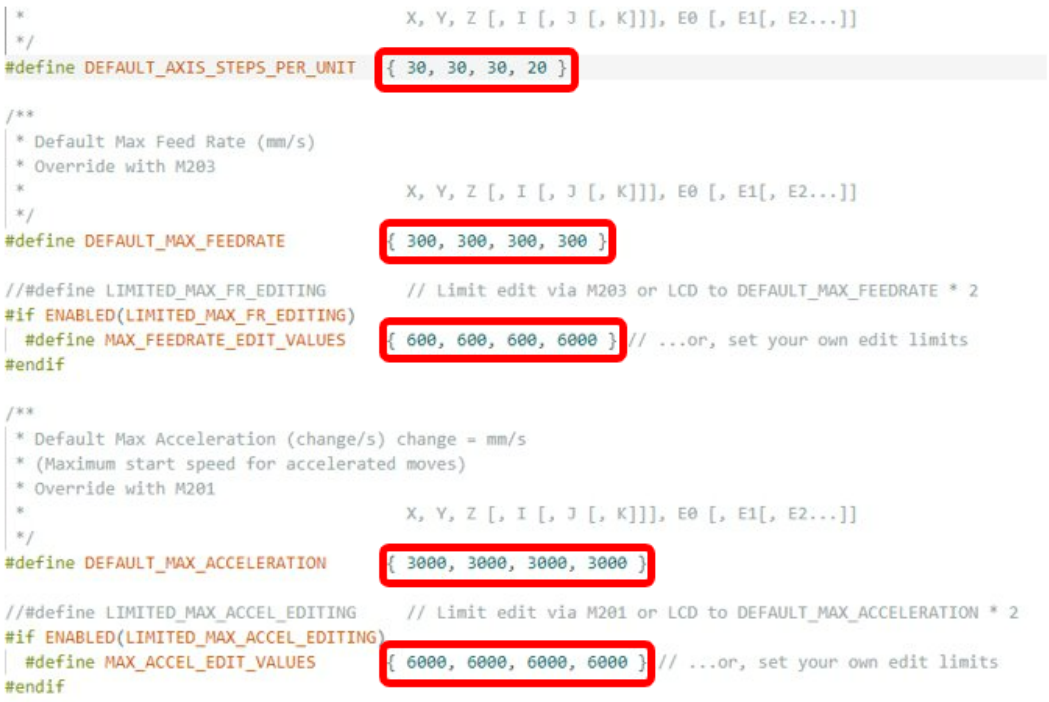
Figure 5. 22: Defining temperature sensor on Marlin software

"ENSTOPPULLDOWNS" is turned on as protection software on 3D printers. It prevents the motor from going in the negative direction. This command has been closed in Configuration.h files, shown in **Figure 5.23** and detailed code explained in **appendix I.3.**

```
// Enable pulldown for all endstops to prevent a floating state
//#define ENDSTOPPULLDOWNS
#if DISABLED(ENDSTOPPULLDOWNS)
  // Disable ENDSTOPPULLDOWNS to set pulldowns individually
  //#define ENDSTOPPULLDOWN_XMIN
  //#define ENDSTOPPULLDOWN_YMIN
  //#define ENDSTOPPULLDOWN_ZMIN
  //#define ENDSTOPPULLDOWN_IMIN
  //#define ENDSTOPPULLDOWN_JMIN
  //#define ENDSTOPPULLDOWN_KMIN
  //#define ENDSTOPPULLDOWN_XMAX
  //#define ENDSTOPPULLDOWN_YMAX
  //#define ENDSTOPPULLDOWN_ZMAX
  //#define ENDSTOPPULLDOWN_IMAX
  //#define ENDSTOPPULLDOWN_JMAX
  //#define ENDSTOPPULLDOWN_KMAX
  //#define ENDSTOPPULLDOWN_ZMIN_PROBE
#endif
```

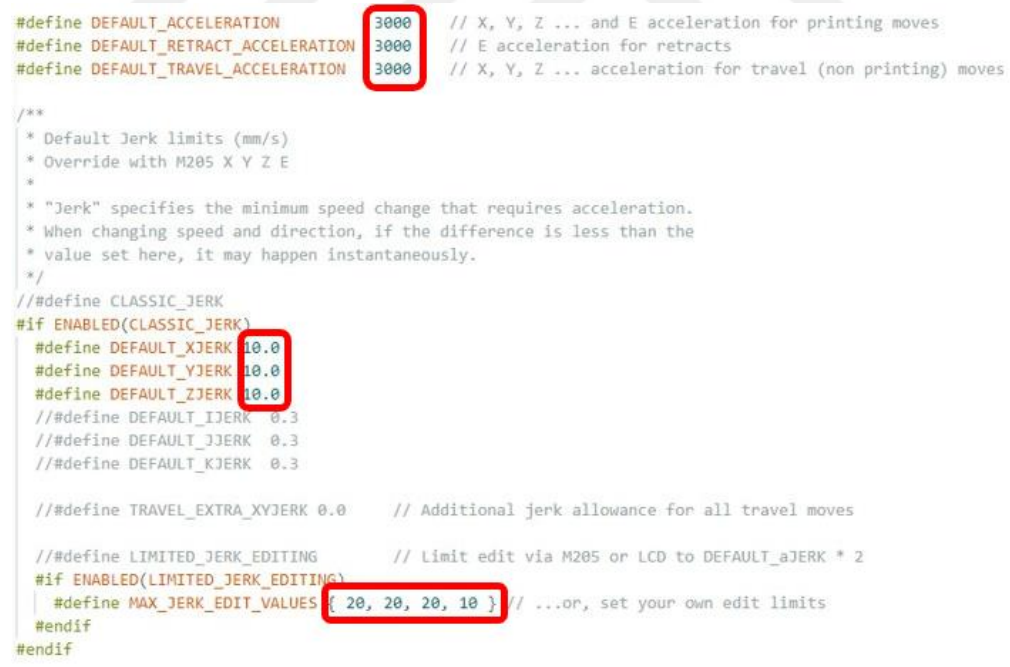
Figure 5. 23: Function that enables negative direction

Speed, acceleration, and step settings of the stepper motor were made in the Configuration.h files, shown in **Figure 5.24** and **Figure 5.25,** detailed code explained in **appendix I.4, appendix I.5** and **appendix I.6.**



```
 *                                    X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */
#define DEFAULT_AXIS_STEPS_PER_UNIT   { 30, 30, 30, 20 }

/**
 * Default Max Feed Rate (mm/s)
 * Override with M203
 *                                    X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */
#define DEFAULT_MAX_FEEDRATE          { 300, 300, 300, 300 }

//#define LIMITED_MAX_FR_EDITING       // Limit edit via M203 or LCD to DEFAULT_MAX_FEEDRATE * 2
#if ENABLED(LIMITED_MAX_FR_EDITING)
  #define MAX_FEEDRATE_EDIT_VALUES    { 600, 600, 600, 6000 } // ...or, set your own edit limits
#endif

/**
 * Default Max Acceleration (change/s) change = mm/s
 * (Maximum start speed for accelerated moves)
 * Override with M201
 *                                    X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */
#define DEFAULT_MAX_ACCELERATION      { 3000, 3000, 3000, 3000 }

//#define LIMITED_MAX_ACCEL_EDITING     // Limit edit via M201 or LCD to DEFAULT_MAX_ACCELERATION * 2
#if ENABLED(LIMITED_MAX_ACCEL_EDITING)
  #define MAX_ACCEL_EDIT_VALUES       { 6000, 6000, 6000, 6000 } // ...or, set your own edit limits
#endif
```

Figure 5. 24: Stepper motor settings on Marlin software



```
#define DEFAULT_ACCELERATION          3000    // X, Y, Z ... and E acceleration for printing moves
#define DEFAULT_RETRACT_ACCELERATION  3000    // E acceleration for retracts
#define DEFAULT_TRAVEL_ACCELERATION   3000    // X, Y, Z ... acceleration for travel (non printing) moves

/**
 * Default Jerk limits (mm/s)
 * Override with M205 X Y Z E
 *
 * "Jerk" specifies the minimum speed change that requires acceleration.
 * When changing speed and direction, if the difference is less than the
 * value set here, it may happen instantaneously.
 */
//#define CLASSIC_JERK
#if ENABLED(CLASSIC_JERK)
  #define DEFAULT_XJERK 10.0
  #define DEFAULT_YJERK 10.0
  #define DEFAULT_ZJERK 10.0
  //#define DEFAULT_IJERK  0.3
  //#define DEFAULT_JJERK  0.3
  //#define DEFAULT_KJERK  0.3

  //#define TRAVEL_EXTRA_XYJERK 0.0     // Additional jerk allowance for all travel moves

  //#define LIMITED_JERK_EDITING        // Limit edit via M205 or LCD to DEFAULT_aJERK * 2
  #if ENABLED(LIMITED_JERK_EDITING)
    #define MAX_JERK_EDIT_VALUES { 20, 20, 20, 10 } // ...or, set your own edit limits
  #endif
#endif
```

Figure 5. 25: Stepper motor settings 2 on Marlin software

In 3D printers, the Z axis works slower than the X and Y axis. To fix that problem, some parts were changed in the Marlin software, and it was made in the

Configuration.h and Configuration_adv.h files. Z_PROBE_FEEDRATE_SLOW command in Configuration_adv.h [33] is the reason that the manipulator's rotation is slow, this issue solved with the Z_PROBE_FEEDRATE_FAST command written in Configuration.h. the manipulator's rotation shown in **Figure 5.26** and detailed code explained in **appendix I.6 and appendix I.7**.

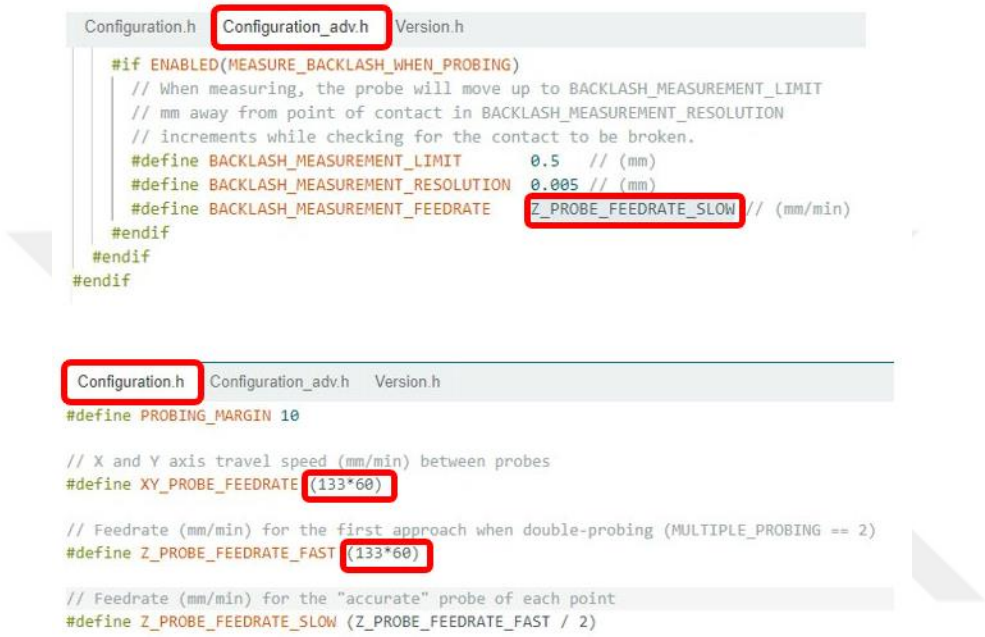


Figure 5. 26: Speed settings on Z-axis

CHAPTER 6

RESULTS

6.1 IMU Results

IMU results include data from sensors such as accelerometers and gyroscopes. This data is processed to determine the position, velocity, and orientation of the device. Accurate and reliable IMU results are a fundamental requirement for many technological applications. The IMU shown in **Figure 6.1** and **Figure 6.2** was used to track the position and motion of a spherical manipulator.

For graphs in the results, the angles given to the motors are not set at the same time, so the values in the graph remain constant for a certain period. The most important part here is that the results must give the desired value, and there may be %15 minor error consideringly the work plate working angle (max 30^{o}), the max error will be $\pm 4.5^{o}$ due to the prototype SPM was produced with 3D printer, PLA.

Figure 6. 1: SPM with IMU

The BWT901CL [32] is a multi-sensor device that can measure angle, angular velocity, acceleration, and magnetic field. AHRS IMU sensor is the scientific name for the BWT901CL. A sensor records acceleration, magnetic field, angular velocity, and 3-axis angle. It is utilized when the highest level of measurement precision is necessary.

Figure 6. 2: SPM with BWT901CL IMU

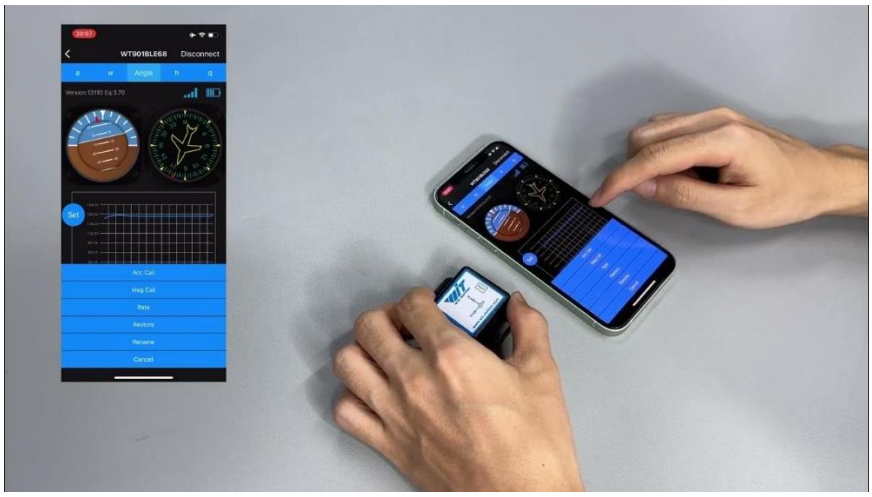


Figure 6. 3: BWT901CL IMU app

6.1.1 0^o 0^o 0^o Position

To read the angles more accurately, an IMU was placed on the top table of the prototype manipulator and the values read from the IMU in the initial position are shown in **Figure 6.4** and **Figure 6.5**.

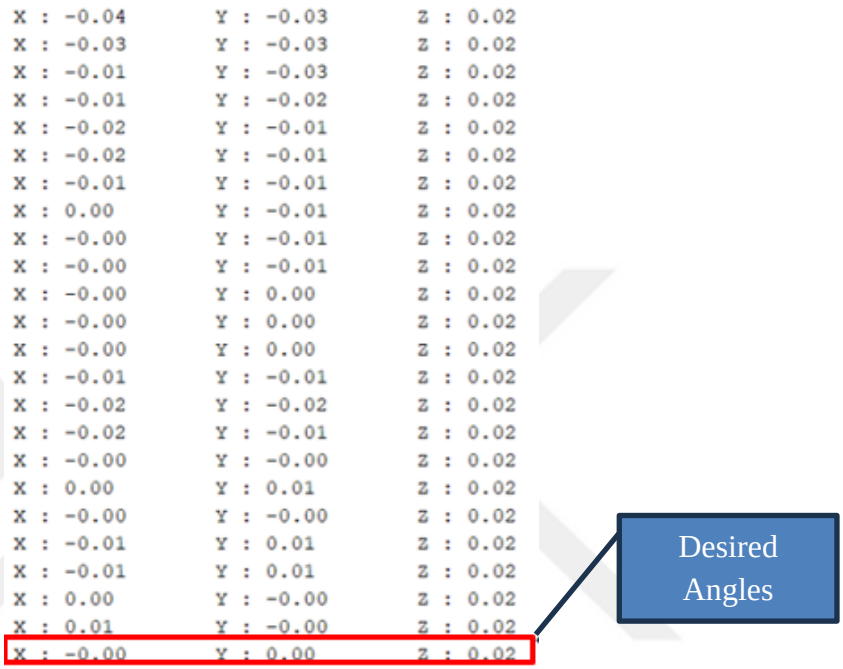


Figure 6. 4: 0^o 0^o 0^o Position IMU values of SPM

Figure 6. 5: 0^o 0^o 0^o Position IMU graph

6.1.2 Yaw Angle

To read the angles more accurately, an IMU was placed on the top table of the prototype manipulator and the values read from the IMU in the Yaw angle are shown in **Figure 6.6** and **Figure 6.7,** and graph shown in **Figure 6.8**.

X: -20,78	Y: 6.08	Z: 0.81
X: -20,79	Y: 5.41	Z: 1.07
X: -20,8	Y: 4.63	Z: 1.06
X: -20,81	Y: 3.44	Z: 1.22
X: -20,82	Y: 2.93	Z: 1.24
X: -20,83	Y: 1.99	Z: 1.52
X: -20,84	Y: 1.37	Z: 1.78
X: -20,85	Y: 0.19	Z: 1.71
X: -20.86	Y: 0.04	Z: 1.52
X: -20.87	Y: 0.67	Z: 1.37
X: -20.88	Y: -0.14	Z: 1.48
X: -20.89	Y: 0.09	Z: 1.51
X:-20.9	Y: -0.03	Z: 1.69
X: -20.91	Y: 0.54	Z: 1.21
X: -20.92	Y: 0.41	Z: 1.31
X: -20.93	Y: 0.38	Z: 1.06
X: -20.94	Y: 0.42	Z: 1.36
X: -20.95	Y: 0.48	Z: 1.25
X: -20.96	Y: 0.62	Z: 1.16
X: -20.97	Y: 0.46	Z: 1.11
X: -20.98	Y: 0.66	Z: 1.03
X: -21	Y: 0.66	Z: 1.19
X: -21.01	Y: 0.66	Z: 1.11
X: -21.02	Y: 0.59	Z: 1.05

Desired Angles

Figure 6. 6: Yaw Angle IMU values of SPM

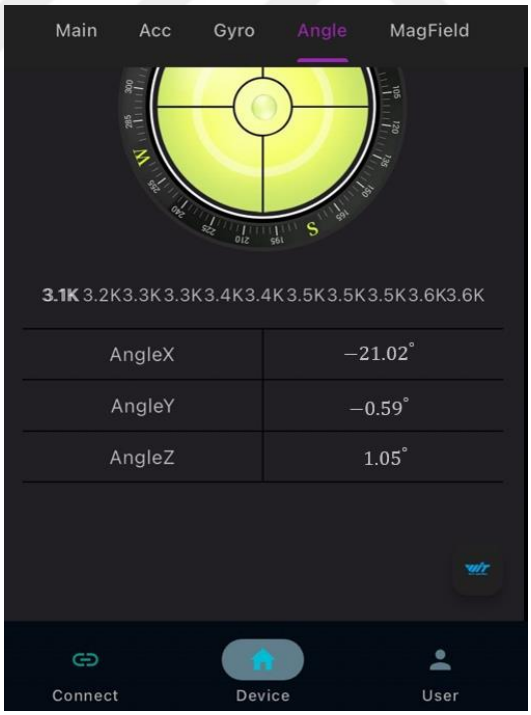


Figure 6. 7: BWT901CL IMU Yaw Angle

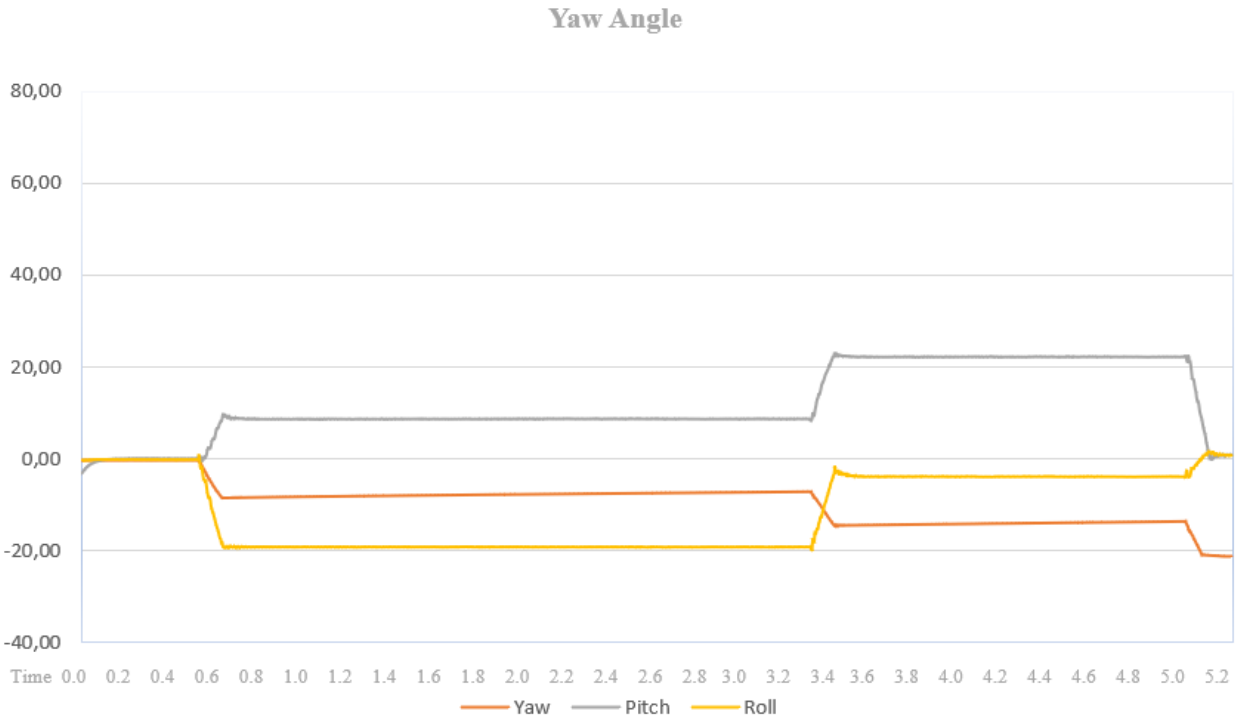


Figure 6. 8: Yaw Angle IMU graph

6.1.3 Pitch Angle

To read the angles more accurately, an IMU was placed on the top table of the prototype manipulator and the values read from the IMU in the Pitch angle are shown in **Figure 6.9, Figure 6.10,** and graph is shown in **Figure 6.11**.

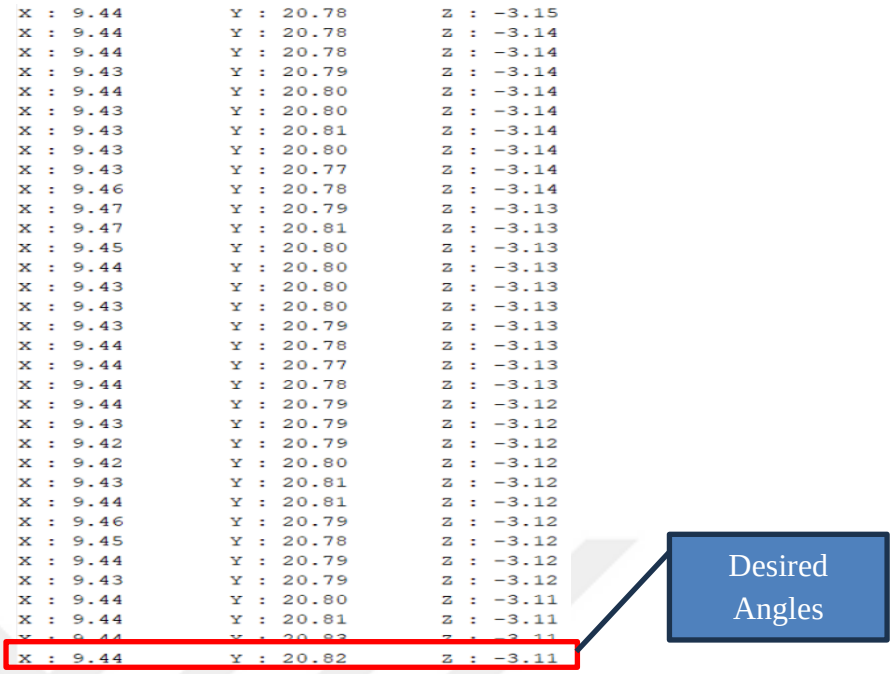


Figure 6. 9: Pitch Angle IMU values of SPM

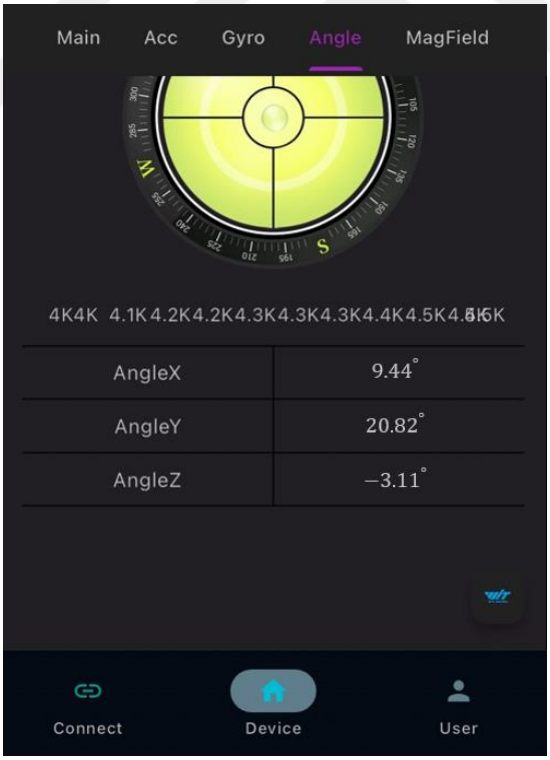


Figure 6. 10: BWT901CL IMU Pitch Angle

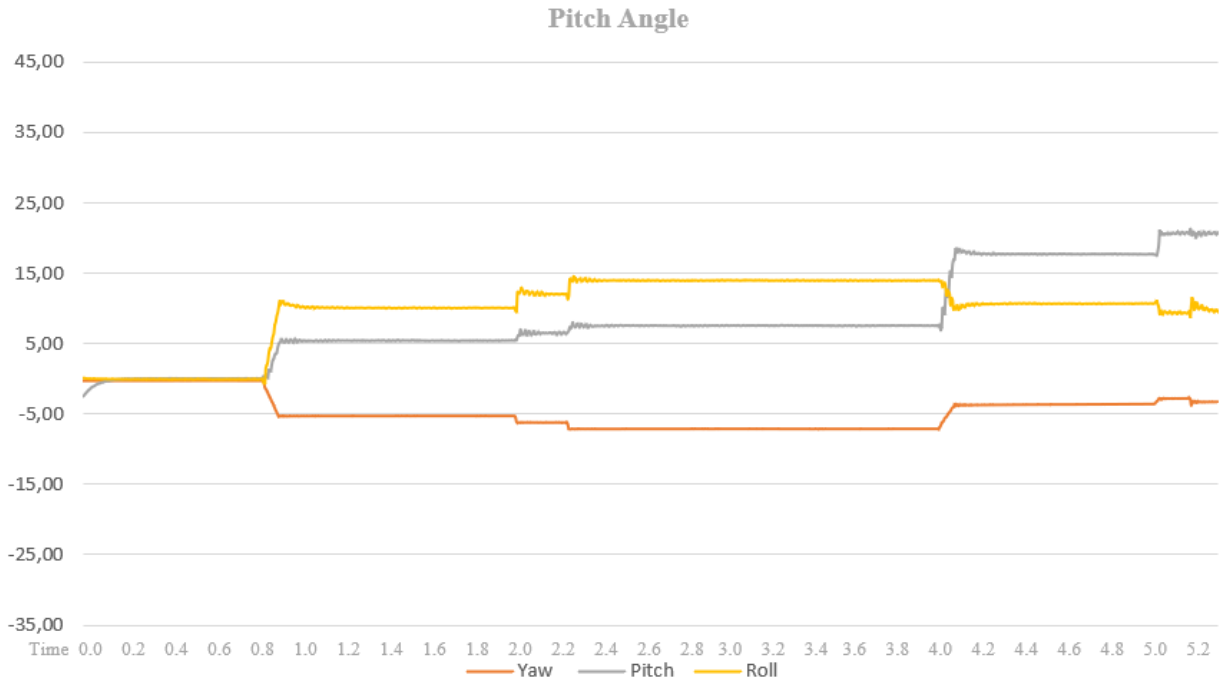


Figure 6. 11: Pitch Angle IMU graph

6.1.4 Roll Angle

To read the angles more accurately, an IMU was placed on the top table of the prototype manipulator and the values read from the IMU in the Roll angle are shown in **Figure 6.12, Figure 6.13,** and graph is shown in **Figure 6.14**.

X : -0.46	Y : 7.84	Z : -28.49
X : -0.46	Y : 7.82	Z : -28.48
X : -0.45	Y : 7.80	Z : -28.48
X : -0.45	Y : 7.80	Z : -28.50
X : -0.45	Y : 7.82	Z : -28.50
X : -0.45	Y : 7.83	Z : -28.50
X : -0.44	Y : 7.82	Z : -28.49
X : -0.44	Y : 7.82	Z : -28.49
X : -0.44	Y : 7.81	Z : -28.49
X : -0.44	Y : 7.80	Z : -28.49
X : -0.44	Y : 7.78	Z : -28.50
X : -0.44	Y : 7.78	Z : -28.49
X : -0.44	Y : 7.80	Z : -28.49
X : -0.44	Y : 7.83	Z : -28.48
X : -0.44	Y : 7.82	Z : -28.49
X : -0.44	Y : 7.81	Z : -28.48
X : -0.43	Y : 7.80	Z : -28.48
X : -0.43	Y : 7.80	Z : -28.49
X : -0.43	Y : 7.81	Z : -28.51
X : -0.43	Y : 7.80	Z : -28.53
X : -0.43	Y : 7.80	Z : -28.52
X : -0.43	Y : 7.80	Z : -28.51
X : -0.42	Y : 7.80	Z : -28.50
X : -0.42	Y : 7.81	Z : -28.52

Desired Angles

Figure 6. 12: Roll Angle IMU values of SPM

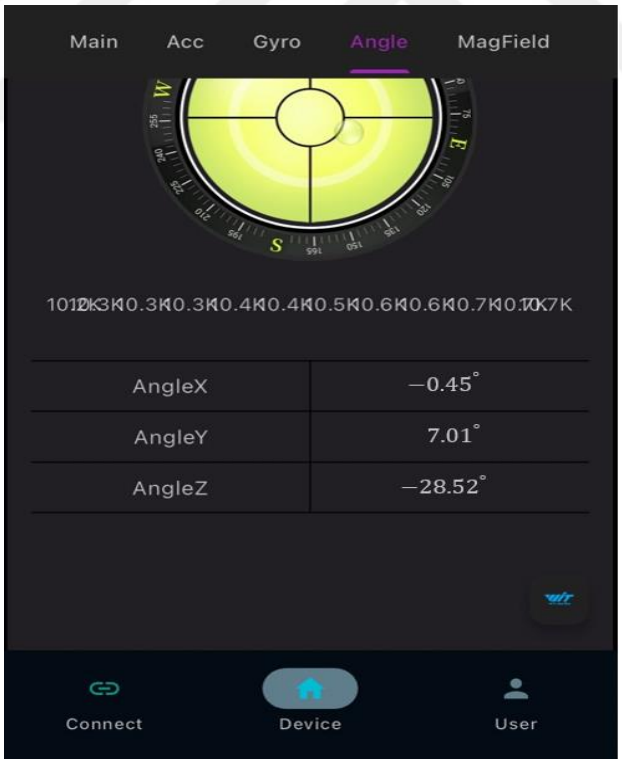


Figure 6. 13: BWT901CL IMU Roll Angle

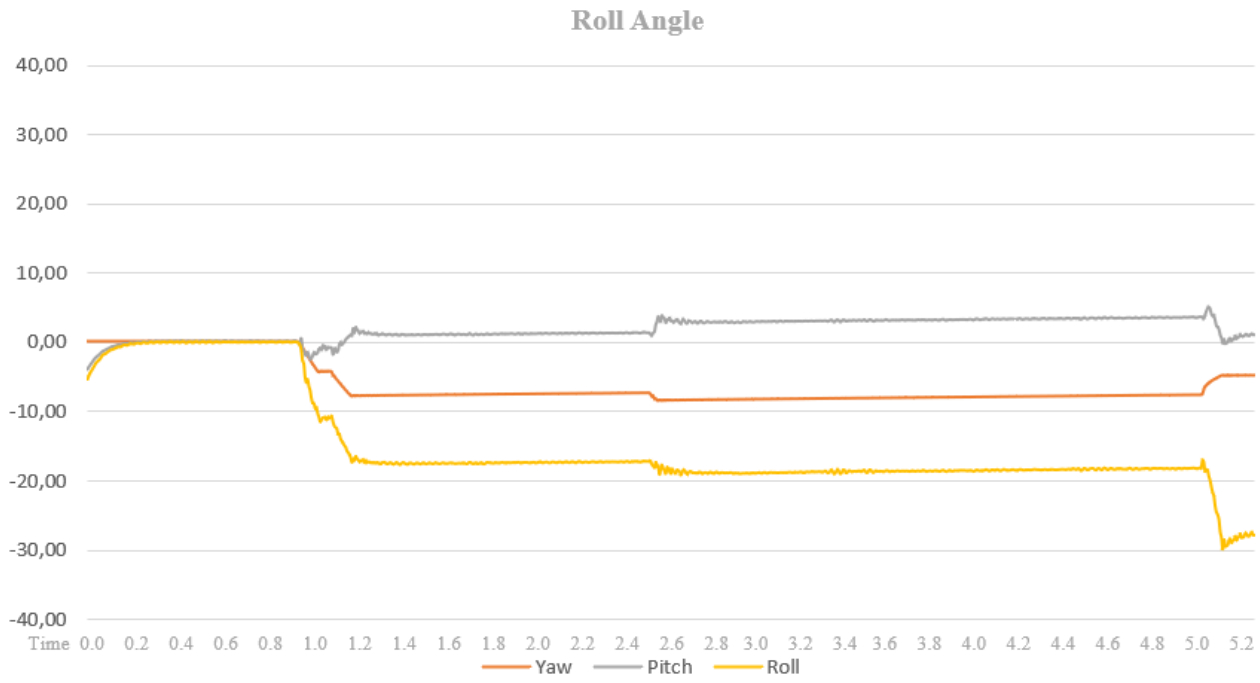


Figure 6. 14: Roll Angle IMU graph

6.1.5 For Three Angle

To read the angles more accurately, an IMU was placed on the top table of the prototype manipulator and the values read from the IMU in the multiple angles are shown in **Figure 6.15, Figure 6.16,** and graph is shown in **Figure 6.17**.

```
X : 4.80        Y : 18.29       Z : -10.05
X : 4.79        Y : 18.29       Z : -10.05
X : 4.79        Y : 18.28       Z : -10.05
X : 4.79        Y : 18.28       Z : -10.04
X : 4.79        Y : 18.28       Z : -10.04
X : 4.79        Y : 18.28       Z : -10.04
X : 4.78        Y : 18.28       Z : -10.04
X : 4.78        Y : 18.29       Z : -10.04
X : 4.78        Y : 18.29       Z : -10.04
X : 4.78        Y : 18.29       Z : -10.04
X : 4.78        Y : 18.30       Z : -10.04
X : 4.79        Y : 18.31       Z : -10.04
X : 4.78        Y : 18.31       Z : -10.04
X : 4.78        Y : 18.30       Z : -10.03
X : 4.77        Y : 18.30       Z : -10.03
X : 4.78        Y : 18.30       Z : -10.03
X : 4.78        Y : 18.31       Z : -10.03
X : 4.77        Y : 18.31       Z : -10.03
X : 4.77        Y : 18.31       Z : -10.03
X : 4.78        Y : 18.30       Z : -10.02
X : 4.79        Y : 18.30       Z : -10.02
X : 4.79        Y : 18.29       Z : -10.02
X : 4.78        Y : 18.28       Z : -10.02
X : 4.76        Y : 18.28       Z : -10.02
X : 4.75        Y : 18.30       Z : -10.02
X : 4.76        Y : 18.30       Z : -10.02
X : 4.75        Y : 18.31       Z : -10.02
X : 4.75        Y : 18.31       Z : -10.02
X : 4.74        Y : 18.31       Z : -10.01
X : 4.75        Y : 18.31       Z : -10.01
X : 4.75        Y : 18.31       Z : -10.01
X : 4.76        Y : 18.31       Z : -10.01
X : 4.77        Y : 18.31       Z : -10.01
X : 4.77        Y : 18.31       Z : -10.01
X : 4.78        Y : 18.31       Z : -10.01
X : 4.78        Y : 18.31       Z : -10.01
X : 4.76        Y : 18.31       Z : -10.00
X : 4.76        Y : 18.32       Z : -10.00
```

Desired Angles

Figure 6. 15: Three Angle IMU values of SPM

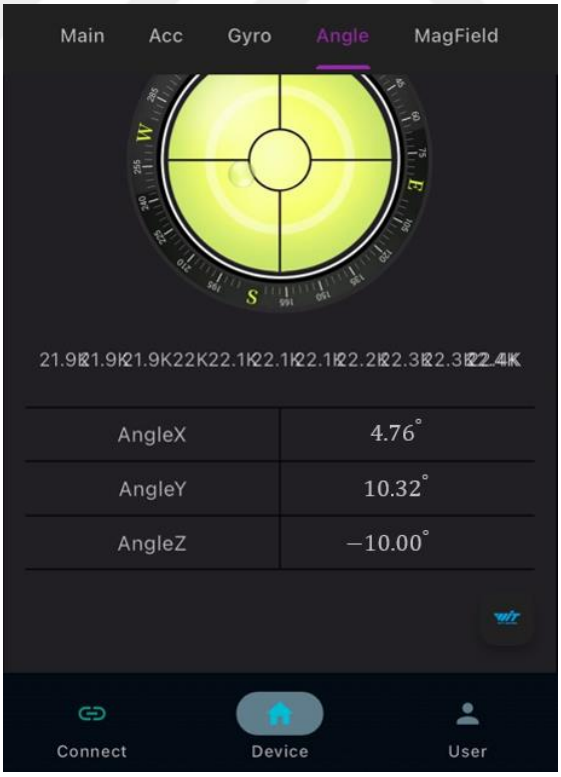


Figure 6. 16: BWT901CL IMU Three Angle

Figure 6. 17: Three Angle IMU graph

6.2 Experiments on Prototype Spherical Parallel Manipulator

6.2.1 0^o 0^o 0^o Position Example

When all angles are given 0^o, the output angles of the kinematic calculation are also 0^o as shown in **Figure 6.18** and **Figure 6.19**.

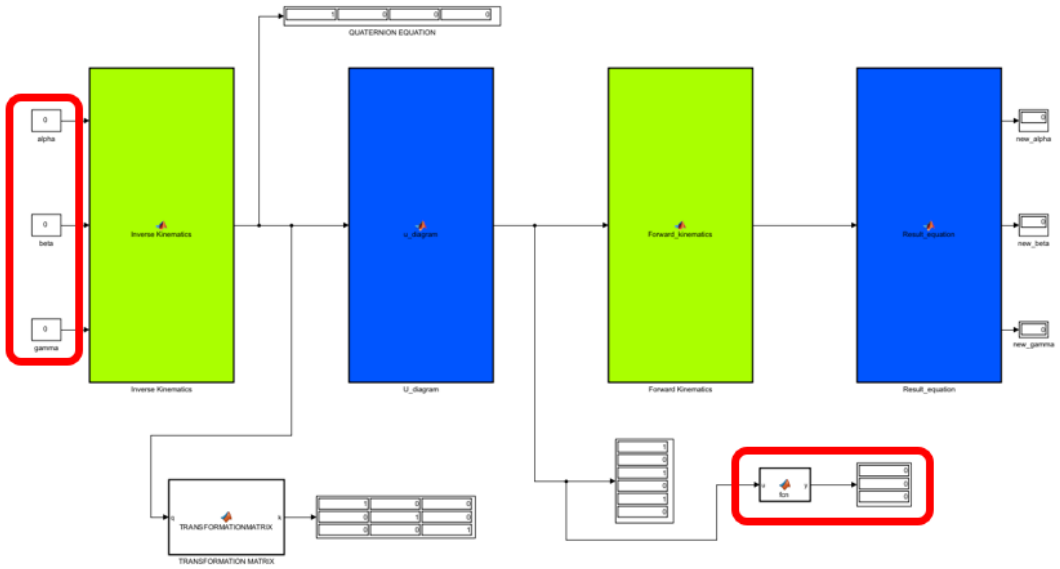


Figure 6. 18: 0^o 0^o 0^o Position Example

The angles provided as input are entered into the kinematic analysis blocks, resulting in the generation of three output angles. These outputs represent the stepper motor data utilized in the prototype of the spherical parallel manipulator. When the stepper motor angles were given for the case, the output on the manipulator became also.

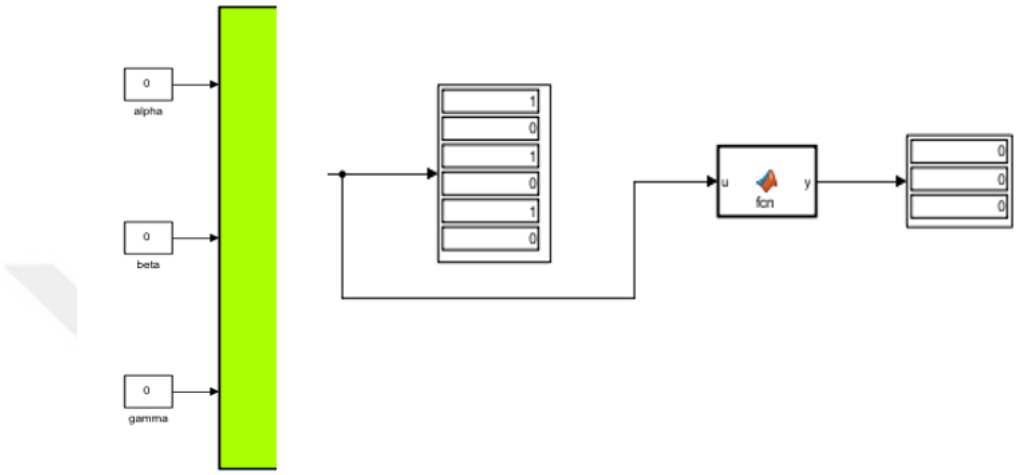


Figure 6. 19: Motor angles for 0^o 0^o 0^o position example

Subsequently, upon entering the angles 0^o 0^o 0^o , the top plate of the manipulator is measured using a protractor, and the angle on the protractor aligns with the input angles 0^o 0^o 0^o used in the simulation shown in the **Figure 6.20**.

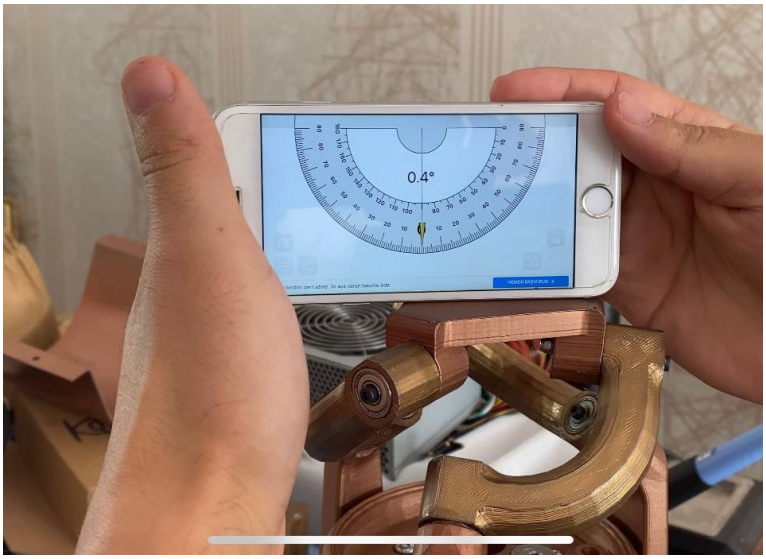


Figure 6. 20: Kinematic verification for 0^o 0^o 0^o position example

6.2.2 Yaw Angle Example

When yaw angle is given 20^o and the others given 0^o the output all motor angles will be 20^o 20^o 20^o as shown in **Figure 6.21**.

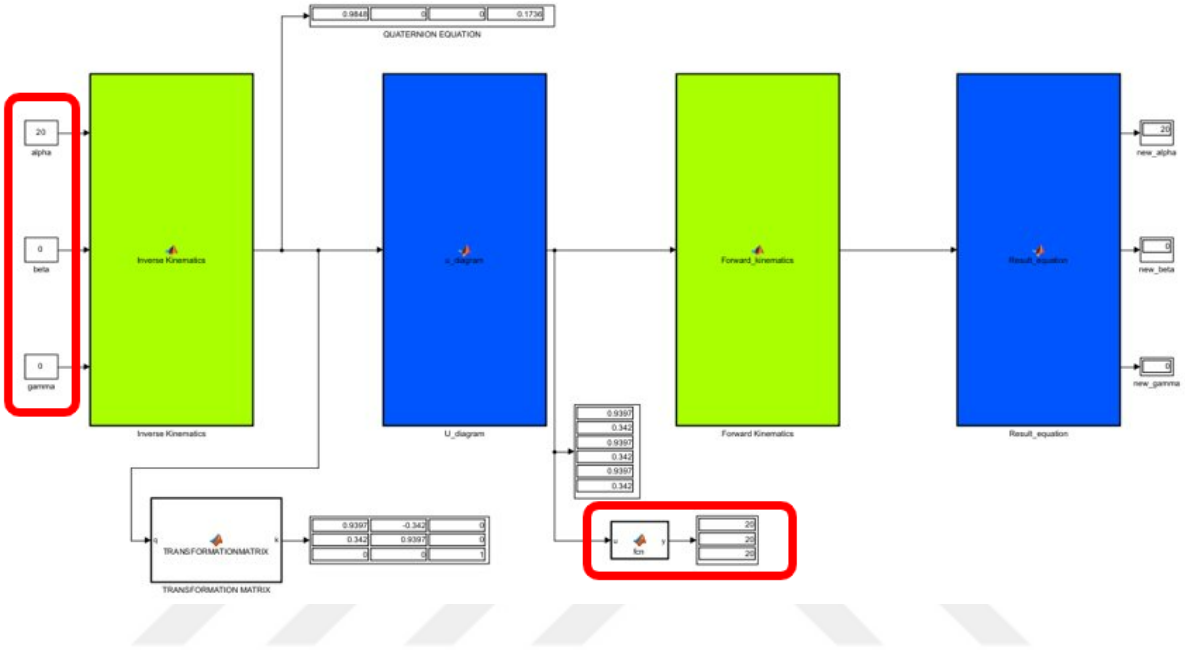


Figure 6. 21: Yaw Angle Example

The angles provided as input are entered into the kinematic analysis blocks, resulting in the generation of three output angles. These outputs represent the stepper motor data utilized in the prototype of the spherical parallel manipulator. Subsequently, upon entering the angles, the top plate of the manipulator is measured using a protractor, and the angle on the protractor aligns with the input angles used in the simulation. When the motor input was 20^o 20^o 20^o for the case, the output on the protractor angles 0^o became also, shown in **Figure 6.22**.

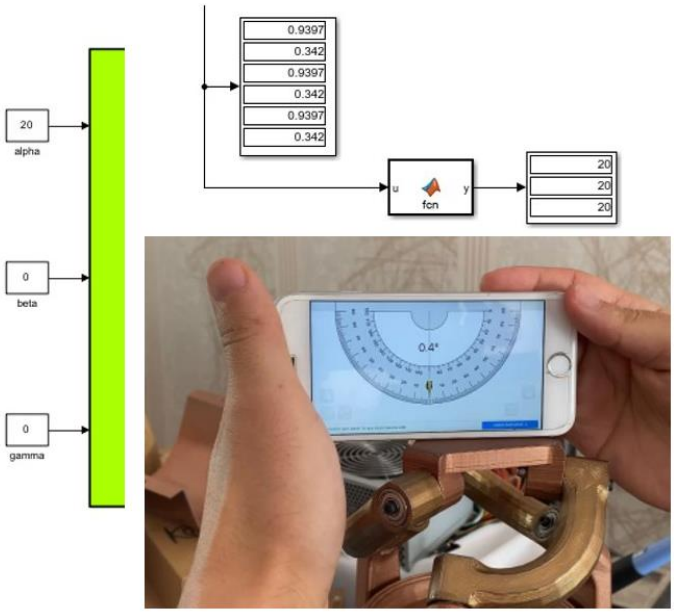


Figure 6. 22: Motor angles for Yaw Angle Example and kinematic verification

6.2.3 Pitch Angle Example

When the pitch angle is given $0^o\ 22^o\ 0^o$ then the stepper motor angles will be $0^o\ 11^o\ -11^o$ as shown in **Figure 6.23**.

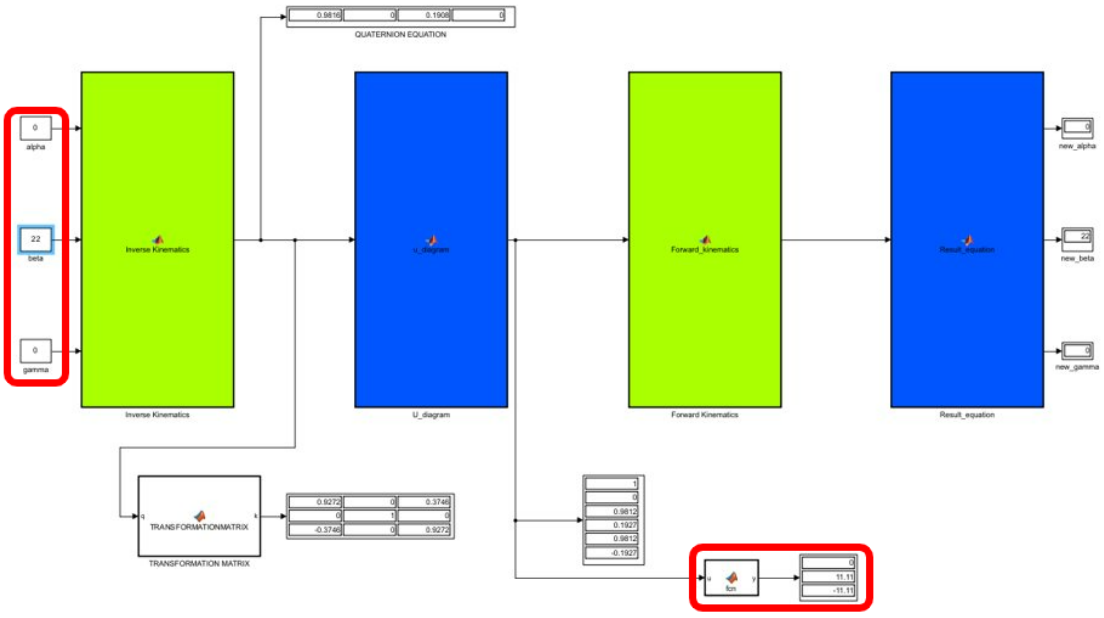


Figure 6. 23: Pitch Angle Example

The input angles are passed through the kinematic analysis blocks, the result will be three output angles. These outputs show the stepper motor data used in the spherical parallel manipulator prototype.

Following that, the top plate of the manipulator is measured using a protractor, and the angle on the protractor aligns with the input angles utilized in the simulation. When the motor angles were set to $0^o\ 11^o\ -11^o$ for the case, the output on the protractor was likewise set to 22^o, as shown in **Figure 6.24.**

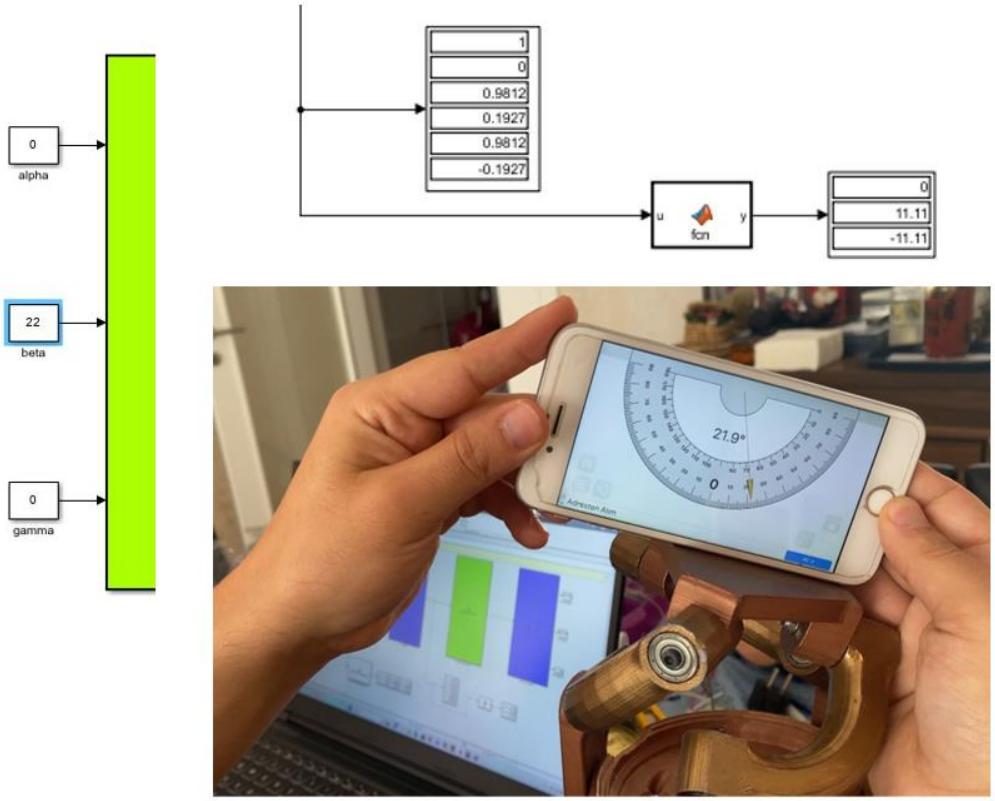


Figure 6. 24: Motor angles for Pitch Angle Example and kinematic verification

6.2.4 Roll Angle Example

When roll angle is given $0^o\ 0^o\ 30^o$ then the stepper motor angles $15^o - 10^o\ -3^o$. The input angles are transmitted through the kinematic analysis blocks, resulting in the derivation of three output angles. These output values represent the stepper motor data applied in the prototype of the spherical parallel manipulator, as shown in **Figure 6.25**.

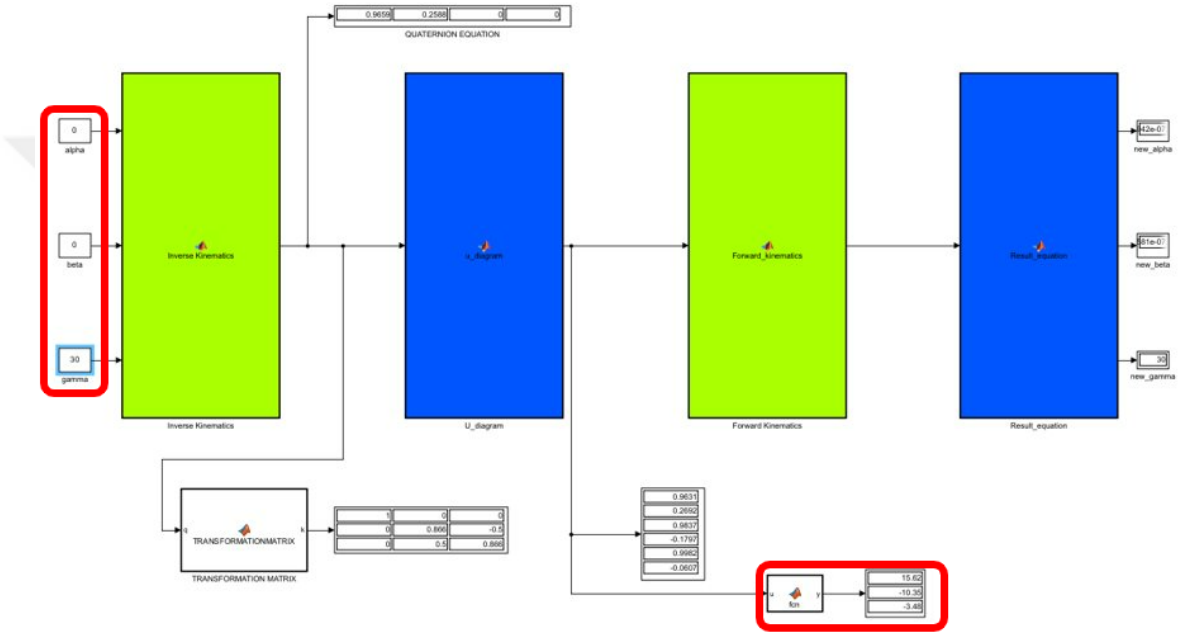


Figure 6. 25: Roll Angle Example

The top plate of the manipulator is measured using a protractor, and the protractor reading corresponds to the motor angles shown in the simulation shown in **Figure 6.26**.

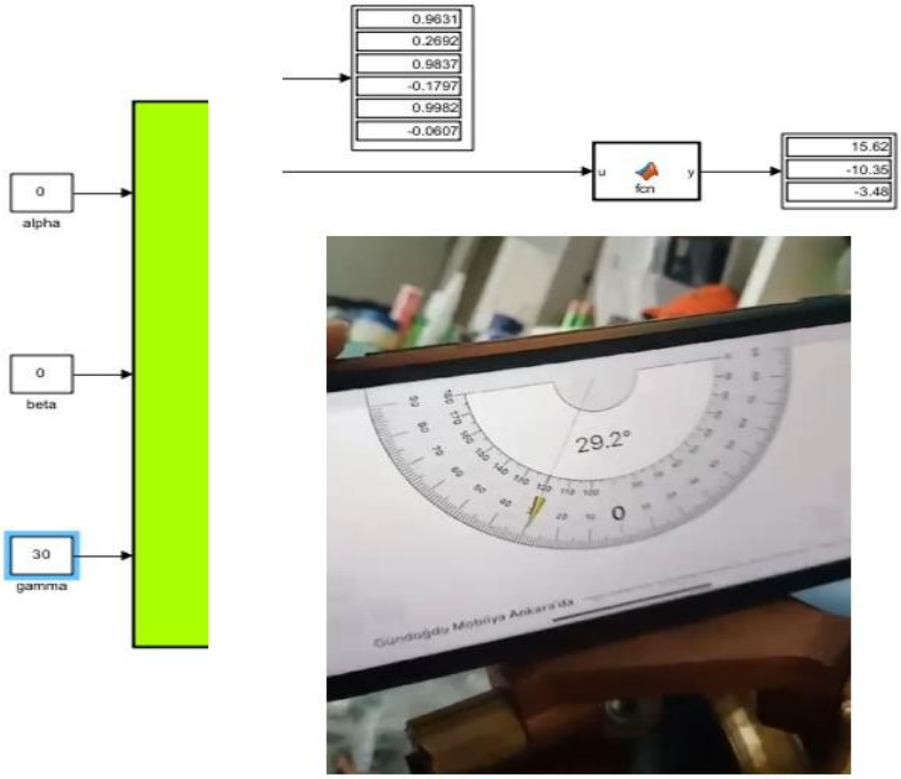


Figure 6. 26: Motor angles for Roll Angle Example and kinematic verification

In a specific scenario where the stepper motor angles were configured as $15^o - 10^o - 3^o$, the value was also read from protractor 30°.

6.2.5 Giving Values for Three Angle Example

When yaw and pitch angle are given as $10^o\ 20^o\ 0^o$ then the stepper motor angles will be $11^o\ 19^o\ 0^o$ as shown in **Figure 6.27**.

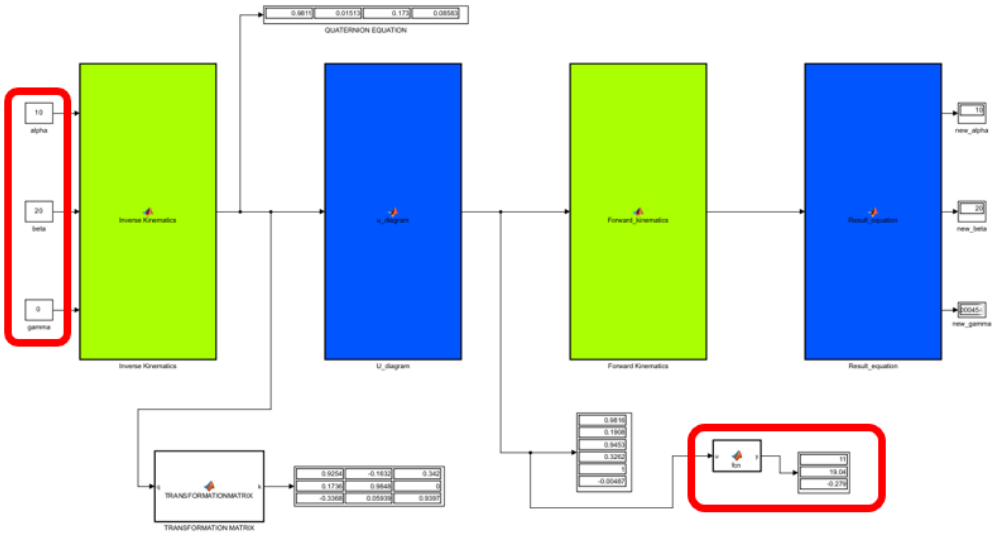


Figure 6. 27: Three Angle Example

The outputs represent the stepper motor data utilized in the prototype of the spherical parallel manipulator. Subsequently, upon entering the angles, the top plate of the manipulator is measured using a protractor, and the angle on the protractor aligns with the input angles used in the simulation. When the motor angles were given $11^o\ 19^o\ 0^o$ for the case, the output on the protractor angles became $11^o\ 19^o\ 0^o$, shown in **Figure 6.28.**

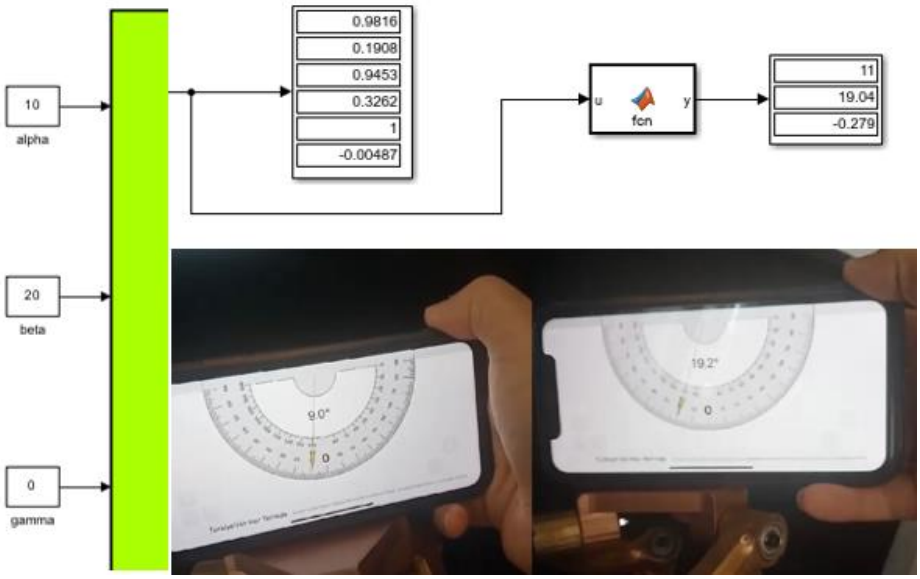


Figure 6. 28: Motor angles for Three Angle Example and kinematic verification

CHAPTER 7

CONCLUSION

In conclusion, this study addressed the design, manufacturing, and control processes of a 3-DOF spherical parallel manipulator. As a fundamental component of optimization, a comprehensive analysis was performed on various design parameters to enhance the manipulator's performance and efficiency. The design of a spherical parallel Manipulator was changed, designed in SolidWorks and Catia V5, and transferred to MATLAB/Simscape, and simulation was made in SimScape. During the design process, several considerations were considered, including Geometric Tolerance, Enhancement, Material Selection, Optimization of Range of Motion, Link Sizing, Load Distribution, and Balancing. To address these considerations, a large bearing was designed to facilitate smooth rotation. Furthermore, the arrangement of the middle body was adjusted to accommodate stepper motors, and couplings were integrated onto the motor shafts to enable their connection to the arms. Careful adjustments of relevant parameters enhanced the manipulator's ability to perform rapid and precise movements. This expands its range of applications in industrial and robotic systems. Additionally, a protective box was designed to house and safeguard the electrical components of the manipulator. This enclosure includes the control card, cables, and power supply. rigorous quality control during the manufacturing phase and appropriate material selection increased the durability and reliability of the manipulator. The designed manipulator was produced on a 3D printer as a prototype and PLA was chosen as the material. It is planned to be produced as metal in future work.

Establishing the validity of a kinematically verified spherical manipulator in real-world applications represents a critical phase in evaluating the usability and effectiveness of this robotic system. This work validated the accuracy of the manipulator's kinematics through physical experiments and verification. The main

purpose of this thesis is to confirm that the outputs of the prototype manipulator, generated through computer-based kinematic analysis, match one another. This study provides valuable insights to researchers, engineers, and industry professionals interested in the design, manufacturing, and control of 3-DOF spherical parallel manipulators. The results obtained can guide the design and optimization processes of similar manipulators and serve as a foundation for future research.

REFERENCES

[1] Sadeqi, S., Bourgeois, S. P., Park, E. J., & Arzanpour, S. (2017). Design and performance analysis of a 3-RRR spherical parallel manipulator for hip exoskeleton applications. Canada; Journal of Rehabilitation and Assistive Technologies Engineering.

[2] Gosselin, C., Perreault, L. L., & Vaillancourt, C. (2002). Simulation and computer-aided design of spherical parallel manipulators. IEEE. https://doi.org/10.1109/oceans.1993.326110

[3] Liu, X., Jin, Z., & Gao, F. (2000). Optimum design of 3-DOF spherical parallel manipulators with respect to the conditioning and stiffness indices. Mechanism and Machine Theory, 35(9), 1257–1267. https://doi.org/10.1016/s0094-114x(99)00072-5

[4] Bai, S. (2010). Optimum design of spherical parallel manipulators for a prescribed workspace. Mechanism and Machine Theory, 45(2), 200–211. https://doi.org/10.1016/j.mechmachtheory.2009.06.007

[5] Bulgarelli, A., Toscana, G., Russo, L. O., Indaco, M., Farulla, G. A., & Bona, B. (2009). A Low-Cost Open-Source 3D-Printable Dexterous Anthropomorphic Robotic Hand with a Parallel Spherical Joint Wrist for Sign Languages Reproduction. United Kingdom; International Journal of Advanced Robotic Systems.

[6] Essomba, T., Hsu, Y. C., Sandoval, J., Laribi, M. A., & Zeghloul, S. (2019). Kinematic optimization of a reconfigurable spherical parallel mechanism for Robotic-Assisted craniotomy. Journal of Mechanisms and Robotics, 11(6). https://doi.org/10.1115/1.4044411

[7] Dinh, T. V., & Kheylo, S. (2020). Solution of the Problem about Speeds and Special Positions of Spherical Parallel Manipulator. International Journal of Mechanical Engineering and Robotics Research, 38–43. https://doi.org/10.18178/ijmerr.10.1.38-43

[8] Vatsal, V., & Hoffman, G. (2021). The Wearable Robotic Forearm: Design and predictive control of a collaborative supernumerary robot. Robotics, 10(3), 91. https://doi.org/10.3390/robotics10030091

[9] Chen, Y. J., Tung, W., Lee, W., Patel, B., Bučinskas, V., Greitāns, M., & Lin, P. T. (2023). Designing and controlling a self-balancing platform mechanism based on 3-RCC spherical parallel manipulator. Robotic Systems and Applications, 3(1), 1–16. https://doi.org/10.21595/rsa.2023.23015

[10] Altuzarra, O., Tagliavini, L., Lei, Y., Petuya, V., & Ruiz-Erezuma, J. L. (2023). On constraints and parasitic motions of a Tripod parallel continuum manipulator. Machines, 11(1), 71. https://doi.org/10.3390/machines11010071

[11] Leal-Naranjo, J., Wang, M., Paredes-Rojas, J., & Rostro-González, H. (2019). Design and kinematic analysis of a new 3-DOF spherical parallel manipulator

for a prosthetic wrist. Journal of the Brazilian Society of Mechanical Sciences and Engineering, 42(1). https://doi.org/10.1007/s40430-019-2153-5

[12] Shintemirov, A., & Niyetkaliyev, A. (2014). An Approach for Obtaining Unique Kinematic Solutions of a Spherical Parallel Manipulator. France; IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM).

[13] Bozorgi, E. R. J., Yahyapour, I., Karimi, A., Masouleh, M. T., & Yazdani, M. (2014). Design, development, dynamic analysis, and control of a 2-DOF spherical parallel mechanism. IEEE. https://doi.org/10.1109/icrom.2014.6990942

[14] Li, T., & Payandeh, S. (2002). Design of spherical parallel mechanisms for application to laparoscopic surgery. Robotica, 20(2), 133–138. https://doi.org/10.1017/s0263574701003873

[15] ZHANG, P., & TLALE, N. S. (2005). Eaching the Design of Parallel Manipulators and Their Controllers Implementing MATLAB, Simulink, SimMechanics and CAD*. Great Britain; TEMPUS.

[16] Bai, S., Hansen, M. R., & Andersen, T. O. (2008). Modelling of a special class of spherical parallel manipulators with Euler parameters. Denmark; Robotica.

[17] Saafi, H., Laribi, M. A., & Zeghloul, S. (2020, August 6). Forward Kinematic Model Resolution of a Special Spherical Parallel Manipulator: Comparison and Real-Time Validation. Tunisia; MDPI.

[18] Chaker, A., Laribi, M. A., Zeghloul, S., & Romdhane, L. (2011). Design and optimization of spherical parallel manipulator as a haptic medical device. IEEE. https://doi.org/10.1109/iecon.2011.6119292

[19] Kizir, S., & Bingül, Z. (2012). Position control and trajectory tracking of the Stewart Platform. In InTech eBooks. https://doi.org/10.5772/35569

[20] Vidaković, J., Lazarević, M. P., Kvrgić, V., Dančuo, Z., & Ferenc, G. (2014). Advanced quaternion forward kinematics algorithm including overview of different methods for robot kinematics. FME-Transactions, 42(3), 189–199. https://doi.org/10.5937/fmet1403189v

[21] Tursynbek, I., & Shintemirov, A. (2020). Modeling and Simulation of Spherical Parallel Manipulators in CoppeliaSim (V-REP) Robot Simulator Software. School of Engineering and Digital Sciences, Nazarbayev University. https://doi.org/10.1109/nir50484.2020.9290227

[22] Güzin, D., & Gezgin, E. (2022). Development of a spherical parallel manipulator for brain surgery applications: preliminary study on dynamic analysis and verification. Robotica, 40(10), 3726–3750. https://doi.org/10.1017/s0263574722000522

[23] Zarkandi, S. (2020). Kinematic analysis and optimal design of a novel 3-PRR spherical parallel manipulator. Proceedings of the Institution of Mechanical

Engineers, Part C: Journal of Mechanical Engineering Science. https://doi.org/10.1177/0954406220938806

[24] Enferadi, J., & Jafari, K. (2020). A Kane's based algorithm for closed-form dynamic analysis of a new design of a 3RSS-S spherical parallel manipulator. Multibody System Dynamics, 49(4), 377–394. https://doi.org/10.1007/s11044-020-09736-y

[25] Crampette, A. (2020, December 18). Orbita is turning heads... literally. Medium. https://medium.com/pollen-robotics/orbita-is-turning-heads-literally-d10d378550e2

[26] Shape the world we live in: CATIA – dassault systèmes. Shape the world we live in | CATIA – Dassault Systèmes. (n.d.). Retrieved January 20, 2023, from https://www.3ds.com/products-services/catia/

[27] Corporation, D. S. S. W. (n.d.). 3D CAD-Design and Software. SOLIDWORKS. Retrieved January 20, 2023, from https://www.solidworks.com/tr

[28] Steiner, W., & Steiner, F. (2015). Commande d'un robot rotule à 3 degrés de liberté à très haute dynamique (modélisation et contrôle). Pasadena; Master of Science HES-SO in Engineering.

[29] Özgören M. Kemal. (2020). Kinematics of general spatial mechanical systems. Wiley.

[30] S. Särkkä, « Notes on Quaternions,» 2007.

[31] Matlab – El Lenguaje del Cálculo Técnico. El lenguaje del cálculo técnico – MATLAB & Simulink. (n.d.). Retrieved January 22, 2023, from https://la.mathworks.com/products/matlab.html

[32] Fazlul. (2021, July 22). Wit motion bluetootn 2.0 Inclinometer Sensor user manual. Manuals+ .https://manuals.plus/wit-motion/bluetootn-2-0-inclinometer-sensor-manual#bluetooth_20_inclinometer_sensor_user_manual

[33] GitHub. https://github.com/gorkemtaskirdi

APPENDICES

A. Bearing 624zz and 625zz

Specific ball bearing types are referred to as "Bearing 624zz" and "625zz". In the bearing industry, these names are frequently used to describe the size and properties of the bearings.

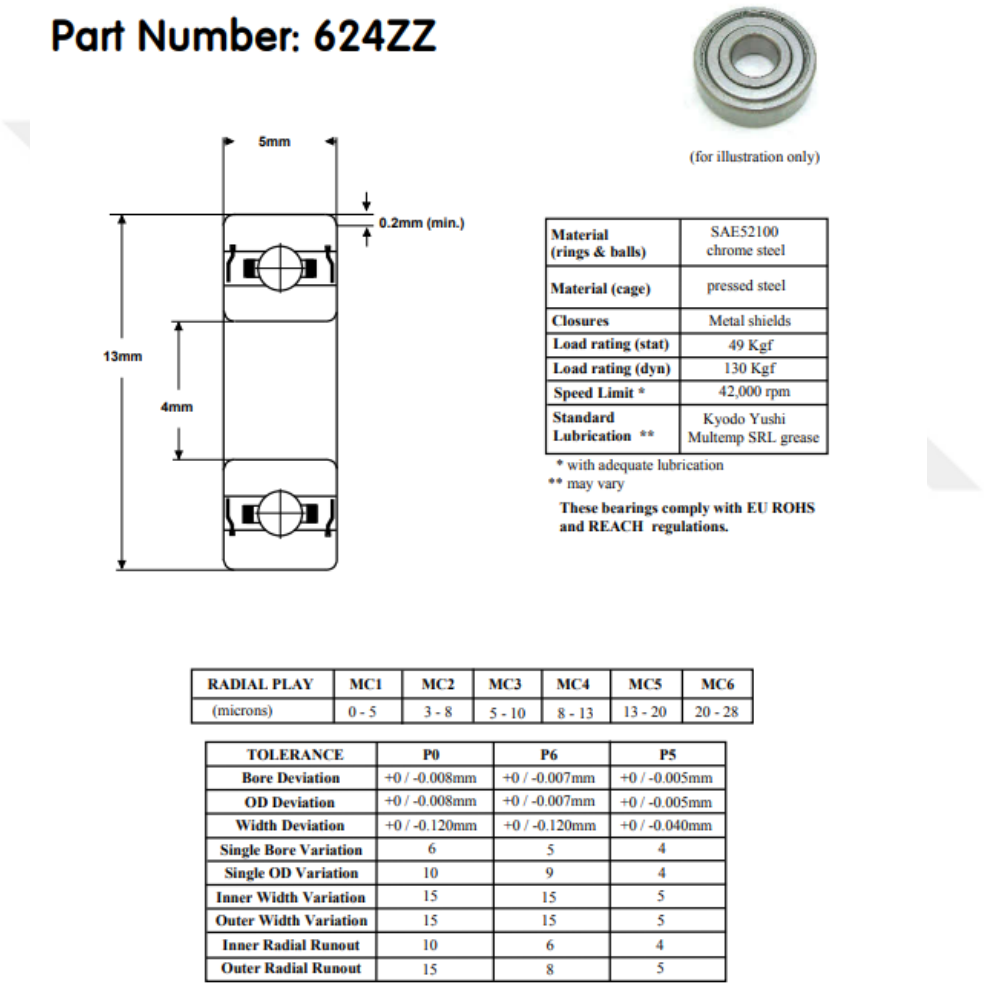


Figure A: 624zz Bearing

PRODUCT DEFINITION

Brand	SNR
d - Internal diameter	5 mm
D - External diameter	16 mm
B - Bearing/Inner ring width	5 mm
d1 - External diameter inner ring	7,7 mm
D1 - Inner diameter outer ring	13,8 mm
rs - Min fillet radius	0,3 mm
Radial clearance class	CN
Mass	0,005 kg

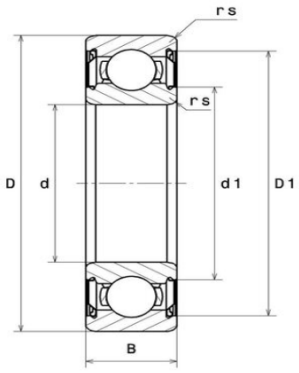


Figure B: 625zz Bearing

B. PLA-Filament

PLA filament (Polylactic Acid) is a popular 3D printing material because of its simplicity of use, biodegradability, and adaptability.

C. Supplies

In many mechanical and construction applications, washers, nuts, and bolts serve as vital elements. They occur in numerous varieties, each created for certain uses.

Figure C: Washers, nuts, and bolts

D. Electrical Cables and Jumpers

Electrical cables are an essential element of electrical and electronic systems, giving devices, parts, and structures a way to carry electrical power and communications. To suit requirements in diverse applications, they are available in a variety of kinds and combinations.

E. Spiral Cable Tray and Clips

An industrial and commercial environment can use a spiral cable tray, often referred to as a spiral wireway or spiral cable organizer, to arrange and support electrical cables, wires, and conduits. It is intended to give safety and flexibility while also giving wires an organized route.

F. Conversion from Solidworks to MATLAB

Solidworks and MATLAB have communication between them. Thus, using this feature it is easy to transfer a CAD assembly to MATLAB/Simulink. This feature provides the primary interface for exporting CAD assemblies into Simscape Multibody software.

First, open the assembly and click on the export box from the SimScape multibody link section in the tools table. A desired blank file is selected, and the tracks are exported as a step file shown in **Figure D.**

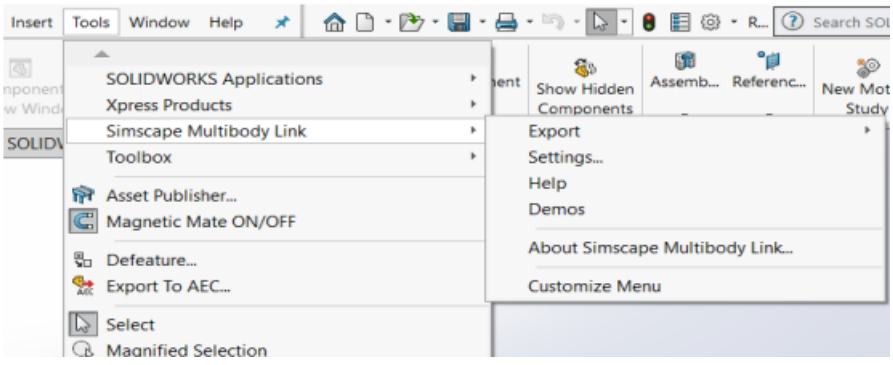


Figure D: Export Solidworks to MATLAB

Among the saved step files is an .slx file. To run this file in MATLAB, go to the command section in MATLAB and write the command smimport ('file name') as shown in **Figure E**. When the opened diagram is run, the system will appear.

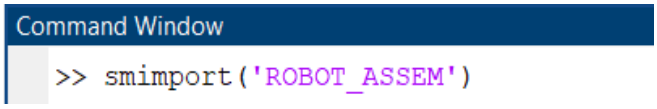


Figure E: Smimport command

G. Simscape Settings of the Manipulator

To adjust the angle in the joint, slider gain must be added. Then, Step signal added. Added degree to radian block to convert data from slider gain to radians. A simulink to physical converter block must be used to add the edited blocks to the joint block as shown in **Figure F.**

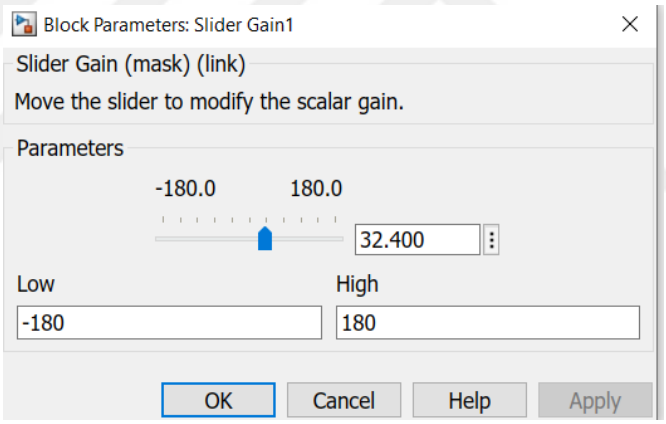


Figure F: Slider gain

To convert the value from gain to acceleration, adjustments must be made in the simulink to physical converter block. In the input handling tab, the filtering and derivatives option should be changed to filter input derivatives calculated. The input filtering order option must be set to second order filtering as shown in **Figure G**.

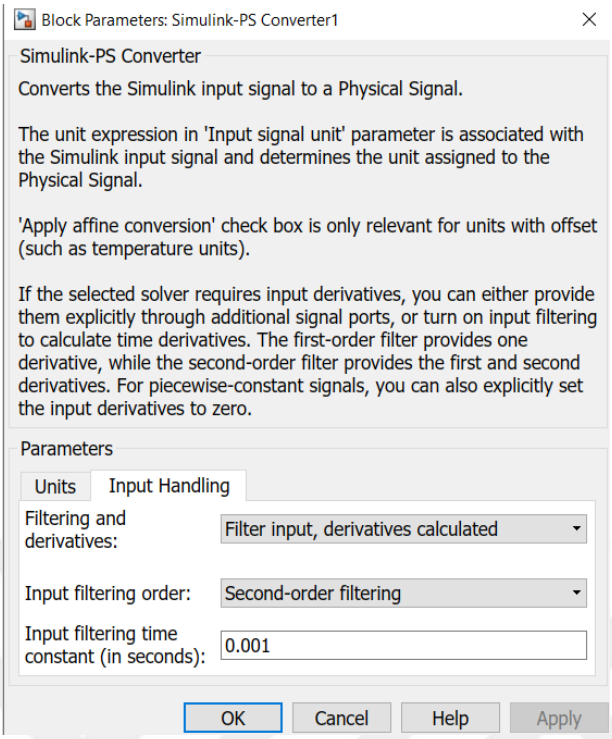


Figure G: Simulink to Ps converter

To be able to angle the joints, the revolute joint settings must be entered, and changes must be made for the actuation tab in these settings. In the Actuation tab, automatically computed should be selected in the torque section and provided by input should be selected in the motion section. To read the changing angle in the slider gain graphically from the scope, the position box must be checked in the sensing section of the joint as shown in **Figure H**.

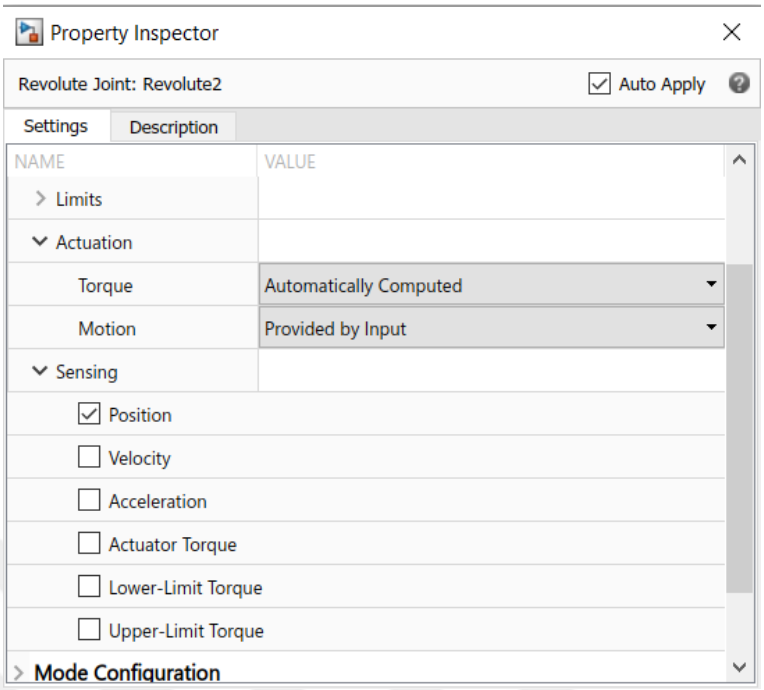


Figure H: Revolute joint

In addition to these, some adjustments were made in the configuration setting section. The solver part has been replaced with ode45 option as shown in **Figure I**.

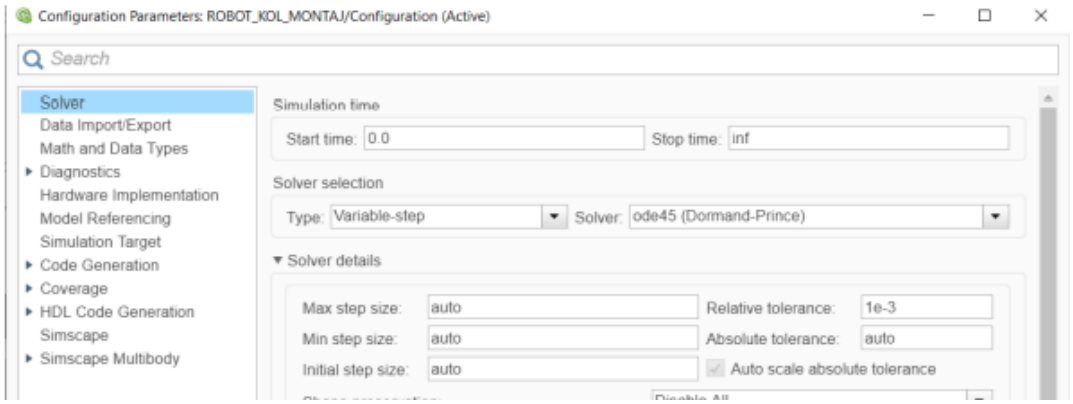


Figure İ: Solver configuration

H. Forward and Inverse Codes in Simulink

- Inverse Kinematics Block

alpha = alpha*(pi/180);

beta = beta*(pi/180);

```
gamma = gamma*(pi/180);

e0 = cos(alpha/2)*cos(beta/2)*cos(gamma/2) -
sin(alpha/2)*sin(beta/2)*sin(gamma/2);

e1 = cos(alpha/2)*cos(beta/2)*sin(gamma/2) +
cos(gamma/2)*sin(alpha/2)*sin(beta/2);

e2 = cos(alpha/2)*cos(gamma/2)*sin(beta/2) -
cos(beta/2)*sin(alpha/2)*sin(gamma/2);

e3 = cos(beta/2)*cos(gamma/2)*sin(alpha/2) +
cos(alpha/2)*sin(beta/2)*sin(gamma/2);

% T01 = [cos(Alpha) sin(Alpha) 0 0 ; -sin(Alpha) cos(Alpha) 0 0 ; 0 0 1 0 ; 0 0
0 1];

% T12 = [cos(Beta) 0 -sin(Beta) 0 ; 0 1 0 0 ; sin(Beta) 0 cos(Beta) 0 ; 0 0 0 1];

% T23 = [cos(Gamma) sin(Gamma) 0 0 ; -sin(Gamma) cos(Gamma) 0 0 ; 0 0 1
0 ; 0 0 0 1];

% T03 = T01*T12*T23;

% C01=[cos(Alpha) sin(Alpha) 0;-sin(Alpha) cos(Alpha) 0;0 0 1];

% C12=[cos(Beta) 0 -sin(Beta);0 1 0;sin(Beta) 0 cos(Beta)];

% C23=[cos(Gamma) sin(Gamma) 0;-sin(Gamma) cos(Gamma) 0;0 0 1];

% C03=C01*C12*C23;

q = [e0,e1,e2,e3];

end
```

- U_Diagram Block

```
function u  = SPM_invKin_pos(q)
```

```
%% Mechanical constants

alphaB = -pi/2;

alphaP = 65*pi/180;

alphaD = -pi/2;

eta = [0 , 2*pi/3 , -2*pi/3];       epsMax = 51*pi/180;

%% Load Quaternion

e0 = q(1);

e1 = q(2);

e2 = q(3);

e3 = q(4);

%% Create Vectors

M=zeros(3);

N=zeros(3);

P=zeros(3);

c=zeros(3);

s=zeros(3);

%% Loop for the three kinematic chains

for i = 1:3

   %% Calculate Constant parameters

   M(i)=sin(alphaP)*(2*e0*e3-2*e1*e2*cos(2*eta(i))+(e1^2-
e2^2)*sin(2*eta(i)));
```

```
    N(i)=sin(alphaP)*(-e0^2+e3^2+(e1^2-
e2^2)*cos(2*eta(i))+2*e1*e2*sin(2*eta(i)));

    P(i)=2*cos(alphaP)*((e0*e1+e2*e3)*cos(eta(i))+(e0*e2-e1*e3)*sin(eta(i)));

    %% Solve quadratic equation (Roots positive is the correct answer)

    s(i)=(+sqrt(M(i)^2+N(i)^2-P(i)^2)*M(i)-P(i)*N(i))/(M(i)^2+N(i)^2);

    c(i)=(-sqrt(M(i)^2+N(i)^2-P(i)^2)*N(i)-P(i)*M(i))/(M(i)^2+N(i)^2);

end

u = [c(1),s(1),c(2),s(2),c(3),s(3)];
```

- Forward Kinematics Block

```
function q  = SPM_dirKin_pos(u)

  coder.extrinsic('quatexp')

%% Load Constants

iMax = 10;              %Maximum Iterations

ikMax = 4;              %Maximum Step decrease Iterations

kini = 1;               %Initial Stepsize

eps = 5.9921e-06;       %Precision for end condition

dk = 0.5;               %Step Change

%% Create Vectors

k = kini;               % Step Size

i = 0;                  % Iterations

ik = 0;                 % Step decrease Iterations
```

```
ue = u;                % End Actuator angles
ud = zeros(1,6);       % Delta Actuator phasors
du = zeros(1,6);       % Acutator step
qip1 = zeros(1,4);     % New orientation
%% Create Initial Conditions
% Calculate mean angle off actuators
cmean = (u(1)+u(3)+u(5));
smean = (u(2)+u(4)+u(6));
divnorm = 1/sqrt(smean^2+cmean^2);
smean = smean*divnorm;
cmean = cmean*divnorm;
thetamean = atan2(smean,cmean);
% Set initial conditions with mean angle
ui = [cmean,smean,cmean,smean,cmean,smean];
qi = [cos(thetamean*0.5) 0 0 sin(thetamean*0.5)];
%% Start iterativ loop
while(1)
   % Calculate phasor difference
   for j=1:3
      is = 2*j;
      ic = is-1;
```

```
    % Calculate phasor difference
    % cd = ce*ci+se*si
    % sd = se*ci-ce*si
    ud(ic) = ue(ic)*ui(ic)+ue(is)*ui(is);
    ud(is) = ue(is)*ui(ic)-ue(ic)*ui(is);
end
% Calculate delta Angle
dtheta = u2theta(ud);
% Check if orientation meets requirements
if(abs(dtheta(1))<eps && abs(dtheta(2))<eps && abs(dtheta(3))<eps)
    break;
else
    for j=1:3
        % Calculate Step phasor;
        is = 2*j;
        ic = is-1;
        du(ic)=-ui(is)*dtheta(j);
        du(is)=ui(ic)*dtheta(j);
    end
    %% Calculate Step Quaternion with the direct Jacobian
    dq = SPM_dirKin_spd(qi, ui, du);
```

```
%% Loop for calculating new Position in Workspace
while(1)
    %% Calculate new Position
    qip1=quatmultiply(quatexp(quatmultiply(dq*k,quatinv(qi))),qi);
    % Check if new position is in Workspace
    if(SPM_qinWrkSpace(qip1))
        % Reset Step size and counter
        ik = 0;
        k = kini;
        break;
    else
        ik = ik+1;
        if(ik >=ikMax)
            error('Maximum Step Decrease Iterations reached');
        else
            % Decrease Step Size
            k=k*dk;
        end
    end
end
% Affect new position
```

```
        qi=qip1;
        % Calculate New phasor angles with inverse Kinematics
        ui = SPM_invKin_pos(qi);
        % Increase Iteration Counter
        i = i+1;
        %% Check for maximum iterations
        if(i >= iMax)
            error('Maximum Iterations reached');
        end
    end
end
q = qi;
```

- Result_Equation Block

```
function [ alpha, beta, gamma ] = q2TBzyx(q)
e0 = q(1);
e1 = q(2);
e2 = q(3);
e3 = q(4);
alpha = atan2(2*(e0*e3-e1*e2),1-2*(e2^2+e3^2));
beta = asin(2*(e0*e2+e1*e3));
gamma = atan2(2*(e0*e1-e2*e3),1-2*(e1^2+e2^2));
```

```
alpha = alpha*(180/pi);

beta = beta*(180/pi);

gamma = gamma*(180/pi);

end
```

I. Software

I.1 Marlin Configuration 1

```
* Stepper Drivers

* These settings allow Marlin to tune stepper driver timing and enable advanced options for

* stepper drivers that support them. You may also override timing options in Configuration_adv.h.

* Use TMC2208/TMC2208_STANDALONE for TMC2225 drivers and TMC2209/TMC2209_STANDALONE for TMC2226 drivers.

* Options: A4988, A5984, DRV8825, LV8729, L6470, L6474, POWERSTEP01,

*       TB6560, TB6600, TMC2100,

*       TMC2130, TMC2130_STANDALONE, TMC2160, TMC2160_STANDALONE,

*       TMC2208, TMC2208_STANDALONE, TMC2209, TMC2209_STANDALONE,

*       TMC26X,  TMC26X_STANDALONE,  TMC2660, TMC2660_STANDALONE,

*       TMC5130, TMC5130_STANDALONE, TMC5160, TMC5160_STANDALONE
```

* :['A4988', 'A5984', 'DRV8825', 'LV8729', 'L6470', 'L6474', 'POWERSTEP01', 'TB6560', 'TB6600', 'TMC2100', 'TMC2130', 'TMC2130_STANDALONE', 'TMC2160', 'TMC2160_STANDALONE', 'TMC2208', 'TMC2208_STANDALONE', 'TMC2209', 'TMC2209_STANDALONE', 'TMC26X', 'TMC26X_STANDALONE', 'TMC2660', 'TMC2660_STANDALONE', 'TMC5130', 'TMC5130_STANDALONE', 'TMC5160', 'TMC5160_STANDALONE']

#define X_DRIVER_TYPE A4988

#define Y_DRIVER_TYPE A4988

#define Z_DRIVER_TYPE A4988

I.2 Marlin Configuration 2

* Custom/Dummy/Other Thermal Sensors

* 0 : not used

* 1000 : Custom - Specify parameters in Configuration_adv.h

* !!! Use these for Testing or Development purposes. NEVER for production machine. !!!

* 998 : Dummy Table that ALWAYS reads 25°C or the temperature defined below.

* 999 : Dummy Table that ALWAYS reads 100°C or the temperature defined below

#define TEMP_SENSOR_0 0

#define TEMP_SENSOR_1 0

#define TEMP_SENSOR_2 0

#define TEMP_SENSOR_3 0

#define TEMP_SENSOR_4 0

#define TEMP_SENSOR_5 0

#define TEMP_SENSOR_6 0

#define TEMP_SENSOR_7 0

#define TEMP_SENSOR_BED 0

#define TEMP_SENSOR_PROBE 0

#define TEMP_SENSOR_CHAMBER 0

#define TEMP_SENSOR_COOLER 0

#define TEMP_SENSOR_BOARD 0

#define TEMP_SENSOR_REDUNDANT 0

I.3 Marlin Configuration 3

//#define ENDSTOPPULLDOWNS

#if DISABLED(ENDSTOPPULLDOWNS)

// Disable ENDSTOPPULLDOWNS to set pulldowns individually

//#define ENDSTOPPULLDOWN_XMIN

//#define ENDSTOPPULLDOWN_YMIN

//#define ENDSTOPPULLDOWN_ZMIN

//#define ENDSTOPPULLDOWN_IMIN

//#define ENDSTOPPULLDOWN_JMIN

//#define ENDSTOPPULLDOWN_KMIN

//#define ENDSTOPPULLDOWN_XMAX

//#define ENDSTOPPULLDOWN_YMAX

//#define ENDSTOPPULLDOWN_ZMAX

//#define ENDSTOPPULLDOWN_IMAX

//#define ENDSTOPPULLDOWN_JMAX

//#define ENDSTOPPULLDOWN_KMAX

//#define ENDSTOPPULLDOWN_ZMIN_PROBE

#endif

I.4 Marlin Configuration 4

* Default Axis Steps Per Unit (steps/mm)

* Override with M92

X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]

#define DEFAULT_AXIS_STEPS_PER_UNIT { 30, 30, 30, 20 }

* Default Max Feed Rate (mm/s)

* Override with M203

X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]

#define DEFAULT_MAX_FEEDRATE { 300, 300, 300, 300 }

//#define LIMITED_MAX_FR_EDITING // Limit edit via M203 or LCD to DEFAULT_MAX_FEEDRATE * 2

#if ENABLED(LIMITED_MAX_FR_EDITING)

#define MAX_FEEDRATE_EDIT_VALUES { 600, 600, 600, 6000 } // ...or, set your own edit limits

#endif

* Default Max Acceleration (change/s) change = mm/s

* (Maximum start speed for accelerated moves)

* Override with M201

X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]

#define DEFAULT_MAX_ACCELERATION { 3000, 3000, 3000, 3000 }

//#define LIMITED_MAX_ACCEL_EDITING // Limit edit via M201 or LCD to DEFAULT_MAX_ACCELERATION * 2

#if ENABLED(LIMITED_MAX_ACCEL_EDITING)

#define MAX_ACCEL_EDIT_VALUES { 6000, 6000, 6000, 6000 } // ...or, set your own edit limits

#endif

* Default Acceleration (change/s) change = mm/s

* Override with M204* M204 P Acceleration

* M204 R Retract Acceleration / M204 T Travel Acceleration

#define DEFAULT_ACCELERATION 3000 // X, Y, Z ... and E acceleration for printing moves

#define DEFAULT_RETRACT_ACCELERATION 3000 // E acceleration for retracts

#define DEFAULT_TRAVEL_ACCELERATION 3000 // X, Y, Z ... acceleration for travel (non printing) moves

I.5 Marlin Configuration 5

* Default Acceleration (change/s) change = mm/s

* Override with M204

* M204 P Acceleration

* M204 R Retract Acceleration

* M204 T Travel Acceleration

#define DEFAULT_ACCELERATION 3000 // X, Y, Z ... and E acceleration for printing moves

#define DEFAULT_RETRACT_ACCELERATION 3000 // E acceleration for retracts

#define DEFAULT_TRAVEL_ACCELERATION 3000 // X, Y, Z ... acceleration for travel (non printing) moves

* Default Jerk limits (mm/s)

* Override with M205 X Y Z E

* "Jerk" specifies the minimum speed change that requires acceleration.

* When changing speed and direction, if the difference is less than the

* value set here, it may happen instantaneously.

//#define CLASSIC_JERK

#if ENABLED(CLASSIC_JERK)

#define DEFAULT_XJERK 10.0

#define DEFAULT_YJERK 10.0

#define DEFAULT_ZJERK 10.0

//#define DEFAULT_IJERK 0.3

//#define DEFAULT_JJERK 0.3

//#define DEFAULT_KJERK 0.3

//#define TRAVEL_EXTRA_XYJERK 0.0 // Additional jerk allowance for all travel moves

//#define LIMITED_JERK_EDITING // Limit edit via M205 or LCD to DEFAULT_aJERK * 2

#if ENABLED(LIMITED_JERK_EDITING)

#define MAX_JERK_EDIT_VALUES { 20, 20, 20, 10 } // ...or, set your own edit limits

#endif

#endif

I.6 Marlin Configuration 6

#if ENABLED(BACKLASH_GCODE)

// Measure the Z backlash when probing (G29) and set with "M425 Z"

#define MEASURE_BACKLASH_WHEN_PROBING

#if ENABLED(MEASURE_BACKLASH_WHEN_PROBING)

// When measuring, the probe will move up to BACKLASH_MEASUREMENT_LIMIT

// mm away from point of contact in BACKLASH_MEASUREMENT_RESOLUTION

// increments while checking for the contact to be broken.

#define BACKLASH_MEASUREMENT_LIMIT 0.5 // (mm)

#define BACKLASH_MEASUREMENT_RESOLUTION 0.005 // (mm)

#define BACKLASH_MEASUREMENT_FEEDRATE Z_PROBE_FEEDRATE_SLOW // (mm/min)

#endif

#endif

I.7 Marlin Configuration 7

// Most probes should stay away from the edges of the bed, but

// with NOZZLE_AS_PROBE this can be negative for a wider probing area.

#define PROBING_MARGIN 10

// X and Y axis travel speed (mm/min) between probes

#define XY_PROBE_FEEDRATE (133*60)

// Feedrate (mm/min) for the first approach when double-probing (MULTIPLE_PROBING == 2)

#define Z_PROBE_FEEDRATE_FAST (133*60)

// Feedrate (mm/min) for the "accurate" probe of each point

#define Z_PROBE_FEEDRATE_SLOW (Z_PROBE_FEEDRATE_FAST / 2)

J. Weierstrass Method

To solve nonlinear equations. The Weierstrass method is used, which is a numerical technique. This technique is used to transform rational trigonometric function expressions into algebraic rational functions, which may make integration easier. The process begins with an initial guess, linearizes the equations around that point, solves the linearized equations to obtain a correction term, and then revises the initial guess to account for the updated correction.

This process is performed repeatedly until the result converges within a suitable tolerance, or a preset stopping threshold is met. However, depending on factors like the initial assumption made and the behavior of the equations close to the solution,

it can need adjustments or alternative techniques for some tough situations. Due to the inclusion of the half-angle approach in the Weierstrass method, this integration is also known as tangential half angle substitution, **Figure J**.

$\sin x = \frac{2\tan\frac{x}{2}}{1+\tan^2\frac{x}{2}} = \frac{2t}{1+t^2}$	$\cos x = \frac{1-\tan^2\frac{x}{2}}{1+\tan^2\frac{x}{2}} = \frac{1-t^2}{1+t^2}$
$\tan x = \frac{2\tan\frac{x}{2}}{1-\tan^2\frac{x}{2}} = \frac{2t}{1-t^2}$	$\cot x = \frac{1-\tan^2\frac{x}{2}}{2\tan\frac{x}{2}} = \frac{1-t^2}{2t}$
$\sec x = \frac{1+\tan^2\frac{x}{2}}{1-\tan^2\frac{x}{2}} = \frac{1+t^2}{1-t^2}$	$\csc x = \frac{1+\tan^2\frac{x}{2}}{2\tan\frac{x}{2}} = \frac{1+t^2}{2t}$

Figure J: Weierstrass Method