FLEXIBLE-CONTINUUM ROBOT FOR BLADDER TISSUE DIAGNOSTICS

by

SAMSON ABIMBOLA ADEJOKUN

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2017

To my parents, Lara and Segun, for their love and sacrifice. To my siblings, Feyi, Funke, Yinka and Nike, for their undeniable friendship.

ACKNOWLEDGEMENTS

I would like to thank Dr. Panayiotis Shiakolas, my supervising professor for advising me on this thesis and for his invaluable guidance throughout the course of my research. I would also like to thank Dr. Alan Bowling and Dr. Tre Welch for serving on my defense committee. I would also express my gratitude to Christopher Abrego for proof reading my thesis document, to Shashank Kumat for his impactful assistance and the entire MARS Lab members for their valuable comments about my research through out its course.

Finally, I am deeply indebted to my parents and siblings for their interest in my education and encouragement. Thank you.

August 8, 2017

ABSTRACT

FLEXIBLE-CONTINUUM ROBOT FOR BLADDER TISSUE DIAGNOSTICS

Samson Abimbola Adejokun, Msc.

The University of Texas at Arlington, 2017

Supervising Professor: Dr Panayiotis Shiakolas

The aim of this thesis is to investigate and develop a robotic system capable of a transurethral palpation of any targeted area on the bladder interior wall tissue to determine the biomechanical properties of the tissue considering the urinary tract geometric constraints and to demonstrate the motion kinematics of such robot to achieve a desired robot pose normal to any localized region throughout the bladder workspace.

Current technologies have, to varied degree of success, provide approximate, global diagnostics information to bladder tissue elasticity. However, no direct access qualitative methods to measure the bladder tissue properties are known. For this reason, a survey of robotic systems applied to minimally invasive surgery was performed with the aim of repurposing existing robotic systems for bladder elasticity dysfunction diagnostics. The result demonstrated their limitations and a requirement for a procedure specific solution.

In the first part, this thesis examines the advantages of flexible robotic manipulators over rigid link robotic manipulators; and the design, actuation and modeling principles of flexible robotic manipulators; the relationship between bladder tissue elasticity and the health condi-

tion of a patient. In the subsequent parts, a conceptualized design of the robotic system which comprised of a flexible-continuum module, a rigid tube and tendon actuation mechanism, and a hyper-spherical actuation base was proposed. Furthermore, the Modified Denavit-Hartenberg parameter approach was applied to obtain the robot forward kinematics motion while a close-loop inverse differential kinematics that makes use of a Jacobian relationship between the joint space and the cartesian space to obtain a desired robot pose. Consequently, mechanical stress analyses of the structural components of the flexible-continuum module are provided to determine the fabrication material(s) using FDA approved standards. Finally, structural components of the flexible-continuum module were prototyped using 3D printing technologies to visualize the proposed robot and its function.

The results show a proposed robot capable of reaching a desired pose at any targeted location normal to the bladder surface with observable position and orientation errors, throughout the bladder workspace. Also, the evaluation of the flexible-continuum module proves the functionality of coupling rigid-flexible components to obtain structural stiffness while also maintaining dexterity.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

The human urinary tract consist of two kidneys, one on either side of the body. The kidneys are each connected to the bladder by a tube called the ureter. The bladder is a round, bag-like organ with a wall thickness of 3.0 ± 1.4 mm. [1] According to the National Institute of Diabetes and Digestive and Kidney Diseases, NIDDK, at the base of the bladder, another tube called the urethra connects the bladder to the prostate and the penis opening in a male but directly to the external urethral oriffice in a female. The urinary tract is tasked with the formulation of urine and export out of the human body. This is accomplished when excess waste is filtered from the blood by the kidneys in form of urine. The urine travels through the ureter to the bladder which in turns stores urine until it is full and then empties itself through the urethra. [2] The average length of the male urethra is 20 cm long while that of the female is 4 cm. Also, the external urethra orifice diameters are 5 mm for the male and uniformly 6 mm for the female. [3]

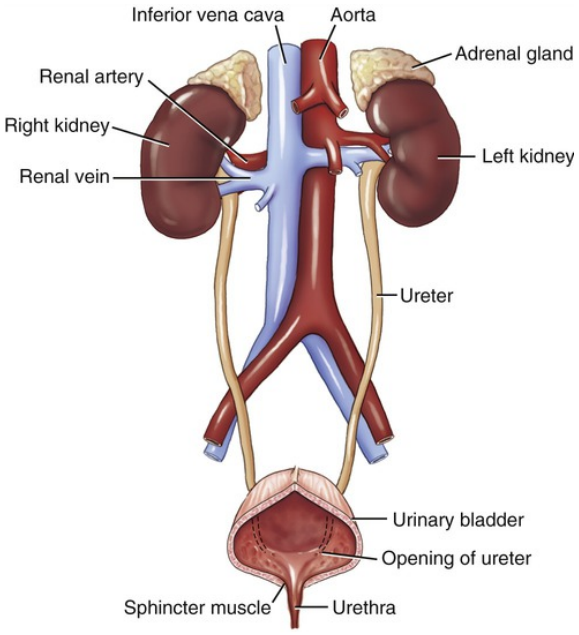


Figure 1.1: The Human Urinary Tract. [4]

According to O'Brien et al, the bladder can usually hold about $8 - 14$ ounces ($0.23 - 0.40$ kilograms) of urine before the brain signals the bladder's detrusor muscle to contract for urination. [3] The volumetric rate at which the bladder fills up depends on the rate of fluid intake. The average human urinates every $2 - 3$ hours during the day and about once at night. However, these urinary pattern such as urinary frequency, urgency, nocturia and post void fullness can be influence by conditions that affect the bladder's operations. Neurologic conditions, infection, loss of bladder tissue elasticity (i.e an increase in tissue stiffness) or tumor are some of these conditions. Clinical studies show bladder elasticity loss is more common, especially (but not always) as it concerns frequency and urgency over time. [5] Studies have also shown a decrease in bladder elasticity in human over time. [3]

According to the NIDDK, 25 to 45% of all females and 11 to 34% of older males in the United State alone suffer from urinary incontinence, high-pressure upper tract changes or overactive

bladder due to bladder elasticity dysfunction. [6] Established techniques to diagnose the causes of the symptoms of bladder elasticity dysfunction involve palpation of the abdomen to rule out the existence of an enlargement of abdominal organs. While in females, the structures around the bladder are checked for any enlargement by examining the pelvic region. For men, a possibility of an infection, inflammation of the prostate gland or prostate cancer are checked by digitally examining the rectum. In other cases, possibilities of a bladder infection is determined using a urine culture. Other methods such as Cystometry, Elastography (ultrasound-based elastography and magnetic resonanance imaging- (MRI-) based elastography) are currently applied to the diagnosis and evaluation of treatment responses of bladder elasticity. Experimental studies have also been carried out using Laser-induced Surface Acoustic Waves (SAWs). [7, 8]

1.1 State of the Art in Non-Invasive Bladder Elasticity Diagnostic Approaches

1.1.1 Laser-induced Surface Acoustic Waves (SAWs)

This technique quantitatively measures the bladder elasticity by using laser-induced surface acoustic waves, SAWs. A pulsed laser is used to energize the surface acoustic waves and then propagated to the bladder wall surface. The corresponding SAWs is then remotely recorded and manipulated to estimate the elasticity properties of the different layers of the bladder tissue. However, some of the shortcomings of this method are that there's an induced measurement error of approximately 9% due to a Poisson ratio and density assumption and data noise due to thermal reactions of the bladder tissue from the laser pulse incidence which invariably lead to tissue damage from thermal exposure. Also, the measurements are limited to averaging the elasticity of regional areas of the bladder which includes the trigone, midsection and dome as oppose to more targeted areas. [8]

1.1.2 Cystometry

A cystometry is performed when a catheter is connected to a cystometer (a device that measures bladder pressure) and inserted through the urethra to the bladder of a patient. A fluid or gas is then slowly introduced to the bladder (the filling phase) through the catheter while the bladder pressure is recorded and the patient describes when he or she gets an initial urge to urinate and when he or she must urinate. Also, when the bladder is completely full, the patient is asked to empty his/her bladder (the voiding phase) while the bladder pressure recording continues. [9] This process is used to determine the average bladder pressure at different volumes in order to match overall but not targeted bladder elasticity using the volumetric pressure information. However, cystometric examination has been documented to only successfully find signs and symptoms of an overactive bladder in less than 47% of patients, especially when it relates to the detrusor muscle overactivity. [10, 11, 12, 13, 14] Studies have also suggested inaccuracies in the diagnosis and symptom management using cystometry, therefore, the cystometry technique is perceived to act as a clinical test to ascertain whether the attributed symptoms of the problems could be reproduced during the filling phase, and whether it abates during the voiding phase. [15, 16]

1.1.3 Elastography

Ophir et al. developed a quatitative method for imaging the elasticity of biological tissues. [17] The procedure uses medical imaging to determine the elastic properties of a soft tissue. It uses stiffness (hardness or softness) data to diagnose the presence or development of a disease. A stress-strain data is obtained by physically distorting the tissue by pushing on the surface of the tissue and then observing the corresponding response. The most prominent elastography techiques are the Ultrasound-Based Elastography (UBE) which compares the before and after ultrasound images of the tissue experiencing a distortion, and the Magnetic Resonance Imaging- (MRI-) Based Elastography (MRIBE) which works similar to the UBE but

employs MRI scanners. Some other elastography base techniques are Vibro-Acoustography and Quasi-Static Method.

The disadvantages with elastography technique are that the stiffness data of the bladder wall suffers from high inter- and intra-observer variability as measurements require a precise bladder volume filling phase with a catheter and a high-resolution ultrasound or a MRI machine. Furthermore, this variability of measurement is relative to the experience or expertise of the imaging technician performing the procedure. [18] Also, the stiffness data is base on a measurement mean of multiple patients, however, studies have shown that this data is inconsistent even during follow-up of the same patient because the bladder wall thickness is dependent on the status of the bladder filling. [19]

1.1.4 Survey of Flexible Robots for Medical Applications

In recent years, medical interventions have demanded advancement in medical procedures and instruments to reducing variability, improve precision, and surgery success while also reducing cost, patient trauma and procedure invasiveness in a bid to provide direct access to internal human features. [20] Minimally invasive surgery has been developed as a vehicle for this advancement. Jaffray defines minimally invasive surgery as any surgical technique which provides access to in-vivo procedures using small size incisions or natural orifice while avoiding traditional healing time and the accompanying pain from large size incisions. [21] Laparascopy, endovascular techniques, natural orifice transluminal endoscopy surgical techniques (NOTES) and catheter-based cardiac techniques are examples of minimally invasive surgery techniques. [22] Central to minimally invasive surgery is the deployment of robotic devices and instruments.

Even though rigid link robots have found applications in medical surgery, difficulties still abound. Difficulties like miniaturizing both robot links and actuation mechanisms, the non-compliance of the materials used in their fabrication have been found to cause damage and/or

puncture to surrounding organs during surgical procedures owing to the limited intramural workspace of a patient. To a lesser degree, rigid link robots still produce some variability due to their noncompliance in the highly unstructured and dynamic environment that is the internal human body. [23] To this effect, there has been a quest for efficient alternatives while not completely discarding the benefits associated with robotic devices in the form of teleoperation and immersive visualization of the intramural workspace, repeatability, precision and so forth. Flexible and continuum robots were first developed in the 1960s and were bioinspired robotic manipulators which mimicked elephant trunks, snakes or octupus tentacles. These robotic manipulators have proven to be a better alternative than traditional rigid body robots. [24] Continuum robots differ from rigid link robots as characterized by the possibility of their large number of joints and scale-able size. They are also very compliant and flexible due to the ease of actuation and the material used in their development.

1.1.4.1 Classification of Flexible-Continuum Robots

Continuum robots are usually categorized, as shown in figure 1.2, using their design structure and means of actuation. Established research has shown that actuation means have proven to be well suited when design size is an issue and space confinement is necessary. [24] Flexibility and dexterity of continuum robots can be optimized by the use of design, actuation and modeling principles as discussed in sections (1.1.4.2, 1.1.4.3 and 1.1.4.4).

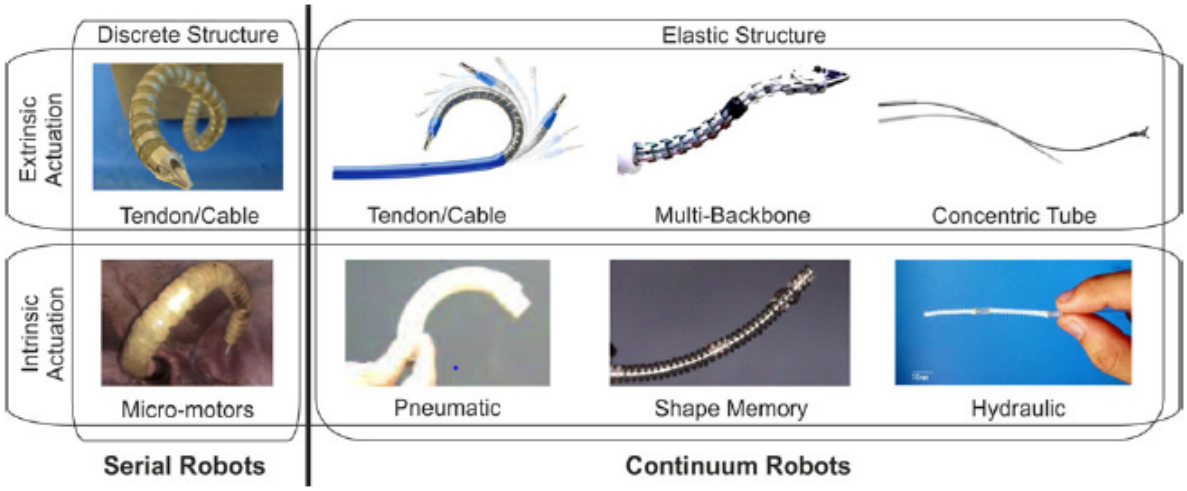


Figure 1.2: Classification of Continuum Robots by Actuation means and Structure. [24]

1.1.4.2 Design principles

Continuum robots comprise of multiple segments called backbones (or vertebrae) that make up a central elastic structure; or just a single elastic structure. The structures are made in a form to allow for routing of signal and power wires to procedure tools as well as the passage of actuation components in a way that doesn't compromise the structure stiffness requirements. [24]

Increasing dexiterity and stiffness are some of the primary design goals in the development of a continuum robot. However, as a manipulator diameter decreases and flexibility increases, stiffness decreases. [25] In essence, there is a trade-off. Therefore, high flexibility and stiffness of continuum manipulators can be achieved if the vertebrae design material, length, diameter and the vertebra-vertebra spacing are effectively resourced. [26]

1.1.4.3 Actuation principles

As shown in figure 1.2, actuation means of continuum robots are either intrinsic (having the actuation mechanism in the continuum structure) or extrinsic (having the actuation mechanism outside the continuum structure). Extrinsic actuation means are currently tendon or cable driven, or shape memory alloys where the lengths of these materials are shorten to create a motion of the structures of a manipulator. [27, 28, 29, 30] Meanwhile intrinsic actuation means are hydraulic, pneumatic and magnetic fields using magnets embedded in the robot structure. Both extrinsic and intrinsic actuation means could be combined as established by Kim's research group. [31, 32] There is no comparable advantage of one actuation means over the other. The determinant of the best suited actuation means is usually determined by the desired actuation and operational force, manipulator geometry, actuation speed, motion parameters, compatibility with other medical equipment and especially, the intended diameter of the robot.

1.1.4.4 Modeling principles

For continuum manipulators, many kinematic and mechanical models have been investigated. Briefly, kinematics approaches involving Denavit-Hartenberg (D-H) parameters [28, 33], constant-curvature [34, 35] and variable curvature [36] have been proposed. While lumped parameter, energy methods and classical elasticity theories are some of the standard approaches for modeling mechanics. [37, 38, 39, 40]

1.1.5 Current Medical Robot Applications

Current research and development of continuum robots for medical applications suggest a growth. Continuum robots could be re-purposed for diagnosing bladder elasticity dysfunction as a means to obtain direct access qualitative measurements of the bladder tissue properties.

Beyond theoretical developments, the survey of continuum manipulators currently used in the medical field or undergoing clinical trials and experimentation have reinforced their usefulness. Therefore, the shortcomings of some of these devices are investigated and presented in this research. Also, the intended contribution this research offer as an alternative will be discussed.

The major shortcomings of existing continuum manipulators lie in their control, structural stiffness, active force and force sensing due to their size, compliance and other design and development trade offs. A intracerebral hemorrhage evacuation concentric-tube continuum manipulator (figure 1.3A) developed by Burgner et al., comprises of a straight outer tube (of outer and inner diameter 1.14 mm and 0.91 mm respectively) and a circular pre-curved tube inserted into the outer tube and linearly actuated extrinsically. [41] The circular pre-curved tube is attached to an aspiration mechanism such that when the curved tube is actuated, its curvature changes and the end effector position can be controlled to where the brain clot is located. However, one of the challenges of the robot design is that the tube is precurved and the motion of the side of the tube can not be explicitly controlled. The clinical experiment shows a significant amount of surrounding brain tissue disturbance and deformation due to this drawback. As a result, procedures are vulnerable to tissue injury.

Yoon et al. (see figure 1.3B) introduced a dual master-slave robotic manipulator for maxillary sinus surgery which constitutes a $4-$DoF continuum manipulator with a 4 mm diameter. [42] The continuum manipulator structure is made of six nodes using a cylinder-spring-cylinder coupling and 4 tendons routed through the cylinder structure that allows a 90^0 bending(right/left and up/down). The continuum manipulator is equipped with either a gripper for ablation or a CMOS camera for investigation. Current work suggests a need to incorporate a navigation means for target detection and end-effector position monitoring in real-time.

Simaan et al.(see figure 1.3C) developed a tele-robotic system suitable for intracorporeal suturing in the throat with the aim of proffering solutions to the dexterity and size limitations of other robotic systems. [43] Their solution is a 4.2 mm diameter flexible-continuum robot with 20 joint space DoFs that can be inserted $180 - 250$ mm down the throat. The 20 joint space DoFs structure of the manipulator is made from an aluminum disk 4.2 mm in diameter actuated by NiTi Tubes. However, the challenges of their design were modeling uncertainties due to the long and flexible backbones that include friction and flexibility of the NiTi tubes actuating the robot. Also, NiTi wires are prone to temperature increase when actuated with electricity. This leads to a high time constant for motion.

Kensner et al. (see figure 1.3D) proposed a robotic catheter for intracardiac tissue ablation using 3D ultrasound image guidance and force control for tracking moving tissue surface when contact is initiated. [44] The 5 mm diameter robotic catheter consists of a force sensor end effector, actuation system and image guidance. Experimentation showed that the system was able to maintain a 1 N consistent contact force with a 0.08 N RMS error. However, their robotic catheter demonstrated sliding and buckling during tissue contact. Hansen Medical (see figure 1.3E) developed a robotic catheter system for endovascular application. [24] Hansen Medical's system consists of a leader and guide part. The leader part is 210 mm long and 2.13 mm in diameter, while the guide part is 3.25 mm in diameter and 500, 650 or 800 mm long depending on desired configuration. The leader part has a bending angle and bending radius of 90^0 and 70 mm respectively.

In accessing the bladder during transurethral surgery, Goldman et al., (see figure 1.3F) provided more dexterity in form of a continuum robot that traditional tools like a resectoscope. [45] The robotic system consists of a distal dexterous manipulator, resectoscope mockup, manual adjustment arm, flexible section, an actuation unit, an end-effector equipped with a laser energy cutter and a gripper. The structure distal end is made of multiple backbones of 0.4064 mm diameter, 25 mm long and is actuated by NiTi wires. Goldman et al. stated

that there is a need for improvement in which accuracy in positioning and dexterity of the end-effector should be pursued.

Hendrick et al., (see figure 1.3G) developed an alternative robotic system for transurethral surgery comprising of two 5 mm diameter concentric tube manipulators extrinsically actuated, two light sources and an endoscope lens in an outer stainless steel tube with a diameter of 8.3 mm. [46] However, the diameter of the endoscope could lead to discomfort due to stretching of the urethra. Also, there is a limitation in the precise control of the concentric tube end effector.

Other U.S. Food and Drug Administration (FDA) approved surgical robots are the Da Vinci Surgical systems and the Zeus Surgical systems as shown in figure 1.3H and 1.3I respectively. The Da Vinci Surgical system is a robotic system with four rigid link robotic arms equipped with interchangeable tools at the end effector. Some of these specialized tools are an endowrist instrument with seven degrees of freedom to mimic the human hand and wrist dexterity for grasping and manipulation, a laparascope tube with a camera and light to enable in-vivo video monitoring of the patient. [47, 48] The Zeus robotic surgical system has three robotic arms equipped with surgical instruments much like the Da Vinci Surgical system. [49] However, the FDA has warned about complications such as bleeding, cuts, tears and punctures to nearby organs. [50] These complications have been attributed to the noncompliance and rigidity of the rigid links and instruments.

In summary, there are several medical robots specifically proposed for different minimally invasive surgeries. Continuum robots have greatly improve the way surgeries are being performed. However, current continuum robot design materials, size, actuation means and operation differs, and are a contributing factor to their limitations. So much so that most applications or procedures requires a specifically designed continuum robotic system considering the constraints of the intermural human workspace.

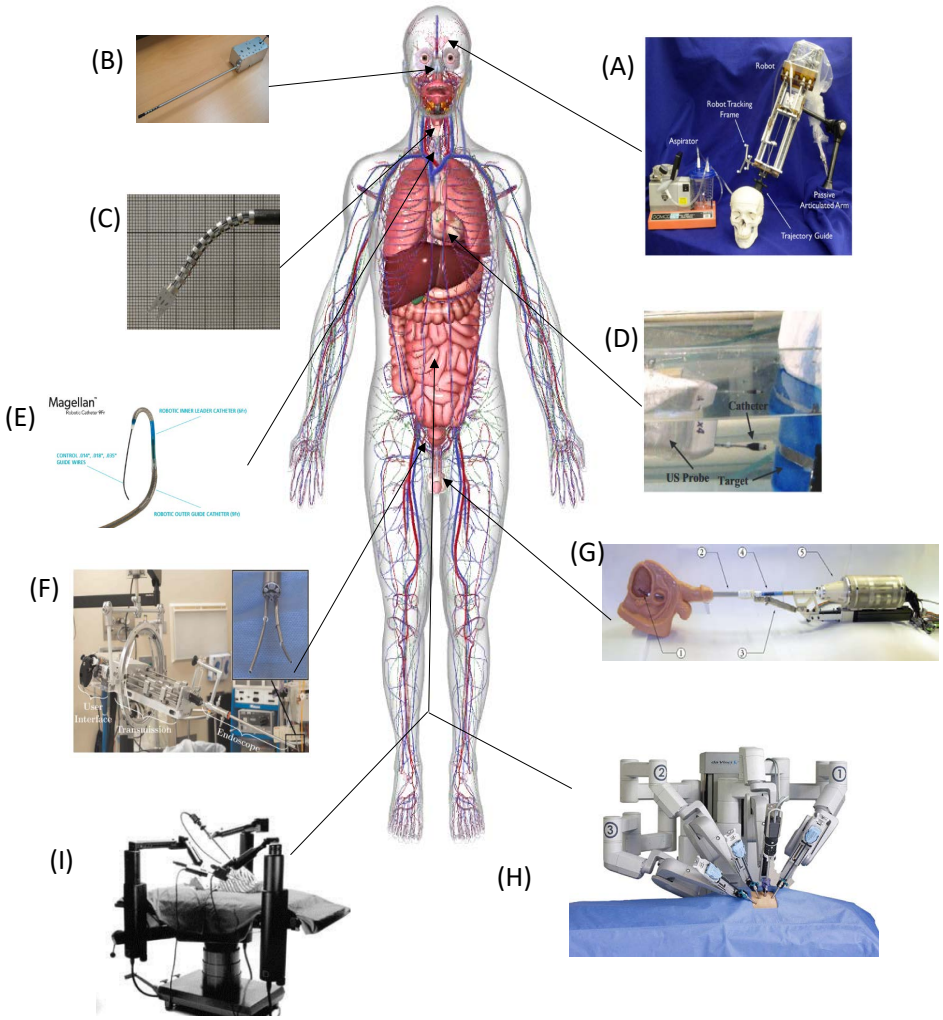


Figure 1.3: Medical Robots and Applications. (A) Intracerebral Hemorrhage Evacuation Concentric-tube Continuum Manipulator by Burgner et al. [41] (B) Robotic Manipulator for Maxillary Sinus Surgery by Yoon et al. [42] (C) Telerobotic System for Intracorporeal Suturing by Simaan et al. [43] (D) Robotic Catheter for Intracardiac Tissue Ablation by Kensner et al. [44] (E) Robotic Catheter System for Endovascular Application by Hansen Medical. [45] (F) Continuum Robot for Transurethral Surgery by Goldman et al. [46] (G) Concentric Tube Manipulator for Transurethral Surgery by Hendrick et al. [46] (H) The Da Vinci Surgical System. [51] (I) The Zeus Surgical System. [52]

1.2 Medical Imaging and 3D Reconstruction

With the introduction of minimally invasive surgery and robotic instruments for medical intervention, there has been a demand for imaging to obtain navigation information, an intramural workspace visualization and geometrical information of organs and/or tissue to improve accuracy and efficiency of the procedure and manipulation of the robotic devices as shown in figure 1.4. [53]

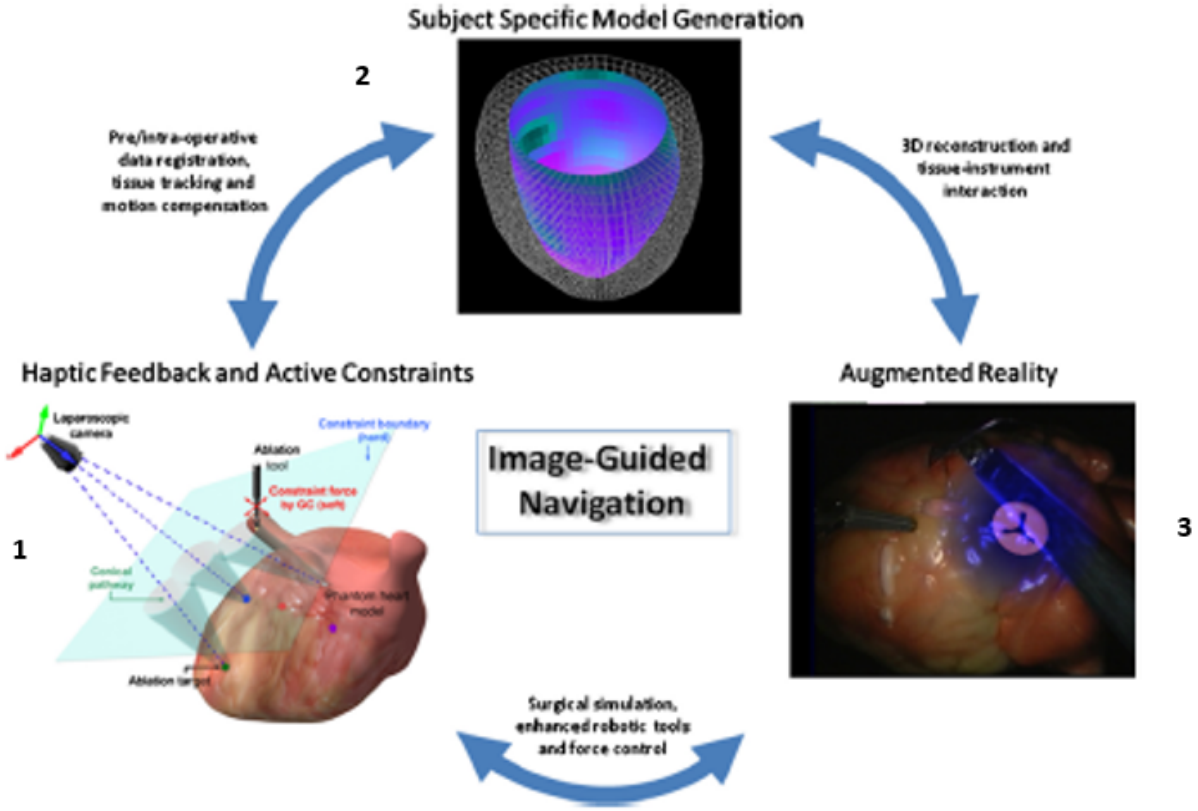


Figure 1.4: Flowchart showing Geometrical Infromation Extraction, 3D Reconstruction for Surgical Robot Navigation, Positioning and Orientation by Lee et al. [20]

Imaging guidance and 3D reconstruction are usually preoperative or intra-operative interventions where biomechanical modeling of tissue deformation or surgeon perception is paramount. Imaging and 3D reconstruction from MRI, Ultrasound, X-ray fluoroscopy are currently medical standards. [53] The first step to surgical robot navigation is to obtain an image either using MRI, ultrasound or X-ray to develop 3D models that describe the geometrical anatomy of the region(s) of interest either pre- or intra-operatively. The geometrical information of the region is then used simultaneously with the kinematic model of the robot for motion control and navigation either offline or in real time. [54] Abolmaesumi et al. developed an ultrasound imaging tracking and visual servoing for a robotic system that accommodates for patient motion. Abolmaesumi et al. suggested that their research provides a way to determine navigation, positioning and orientation for surgical robotic systems. [55]

1.3 The Bladder and Tissue Properties

In addition to the geometric properties of the bladder, there are other biomechanical properties that will pose design size, navigation and manipulation constraints to any minimally invasive surgical robotic systems. These contraints need to be understood. Studies provided by Rubod et al. from the biomechanical examination of five fresh female cadavers with a mean age of 75 years suggested that the bladder tissue is less stiffer than the other human pelvic organs such as the vagina and rectal tissue. [56] A 1 kN load cell was used to measure the stress level of the tissues during uniaxial tension. From uniaxial tension experimentation with monotonic and cyclic loading on test samples with size 25 mm $\times$ 4 mm, Rubod et al. obtained stress-strain relationships that demonstrated these tissue samples (bladder tissue included) exhibit nonlinear stress-strain relationships and visco-hyperelastic characteristics. During the loading and unloading phase from the application of uniaxial tension from 0 N to rupture, their research showed large hysteresis and an 80 % strain level before rupture. It was concluded that there was no significant difference in the stiffness of the bladder anterior

and posterior wall at low strain level, however it was anisotropic at high strain level. The results of this research show that the bladder tissue biomechanical or elastic properties can be modeled from the stress-strain relationship when a force is applied as shown in figure 1.5.

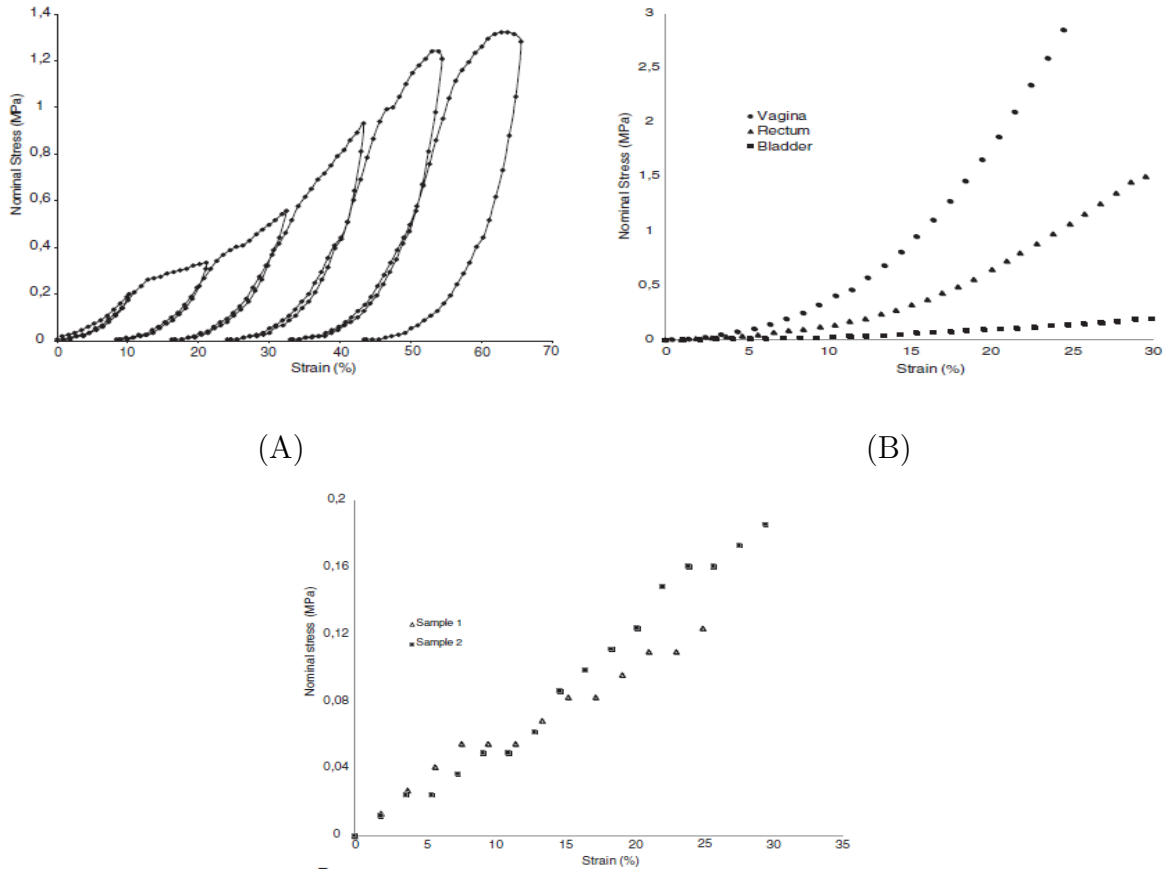
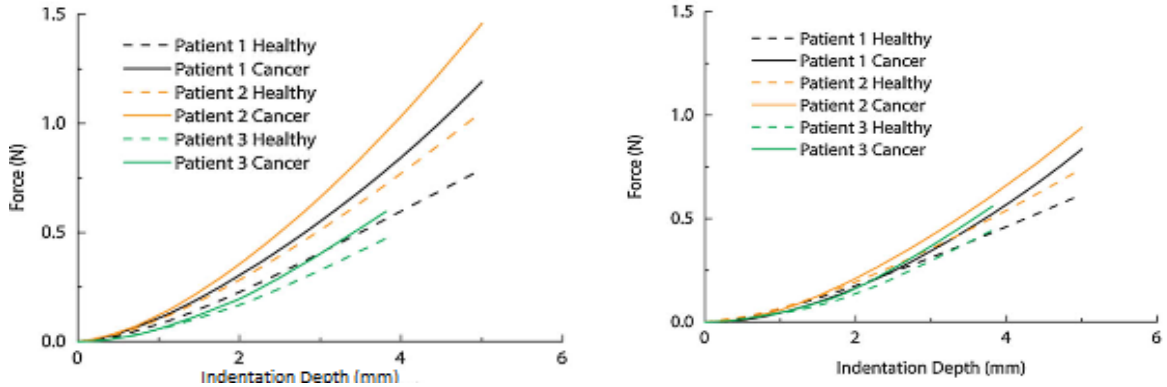


Figure 1.5: (A) Cyclic Loading Test on Bladder Tissue Showing Stress versus Strain, (B) Biomechanical Behavior of Vaginal, Rectal, and Bladder Tissue Showing Nominal Stress versus Strain during Uniaxial Tension Tests, (C) Bladder Rapture Test. [56]

Palacio-Torralba et al. also compared the palpation force-indentation depth relationships of the bladder tissue of healthy and cancer patients from palpation using a 5 mm radius spherical indenter when subjected to high and low bladder volumetric pressure. They found that the

sensitivity of the procedure was improved when the bladder was at a higher volumetric pressure and that the bladder tissue is more responsive as larger indentation depths were achieve when subjected to lesser forces in healthy patients than for cancer patients (as shown in figure 1.6). [57] This study proved that healthy bladder tissues have more elasticity.



(A) (B)

Figure 1.6: Palpation Force of three Patients Subjected to (A) Higher Bladder Volumetric Pressure, (B) Lower Bladder Volumetric Pressure. [57]

Konstantinova et al. suggest a palpation depth of 4 to 6 mm is required when the localized target area is been examined for shape and stiffness abnormality. [58] Chuong et al. showed that an applied pressure of 0.01 N/mm^2 is sufficient to achieve a soft tissue palpation depth of 6 mm. [59]

1.4 Contributions of this Research

The limitations of the techniques for bladder dysfunction diagnosis discussed in section (1.1) suggest a need for an alternative method which aims to provide more precise, unique biomechanical properties of the bladder tissue in a simple, easy to use and inexpensive way, especially for targeted locations on the interior bladder wall. This method will have to be a

minimally invasive, direct physical contact method which can help track the changes of the quantitative properties of the bladder tissue elasticity to adequately diagnose the problem and track the progress of the treatments. However, the existing minimally invasive surgical robots discussed in section (1.1.5) are well suited (in varying efficiency) for other surgical interventions involving ablation, visual inspection and so forth and, to a lesser degree, for palpation. Some of the robotic devices are also limited by their size and/or dexterity that will make them not feasible for their application in the characterization of the biomechanical properties of targeted areas of the bladder wall.

Therefore, the focus of the presented research is a robotic device to improve bladder wall tissue elasticity estimation by direct contact palpation. Using a force sensor mounted on an inexpensive, minimally invasive hybrid flexible-rigid link manipulator based on NOTES techniques. This robotic system will be capable of transurethral navigation and positioning. To accomplish this, an understanding of the human urinary anatomy must be achieved, a robotic system design must be conceptualized using the geometric constraints of the urethra and bladder placed on the robot size. The design information, along with the geometric information of the bladder, will be used to develop the robot kinematics for position and orientation control of the end-effector. As a matter of fact, the desired position and orientation information will be extracted from the geometry of the targeted area on the bladder wall relative to the insertion location in the bladder. Ultimately, an attempt to understand the reaction forces and generated stresses on the components of robotic system will improve the mechanical design. To that end, the following contributions are achieved:

1. Design a 3 mm outer diameter continuum robot manipulator that attempts to completely capture the biomechanical properties of the bladder with direct contact of any specified region of the bladder wall.
2. Considering the recommended palpation pressure of 0.01 N/mm^2 and a robot outer diameter of 3 mm, the robot will be capable of applying a palpation force of 0.1 N

without structural failure to meet the FDA approved medical devices factor of safety standard of ≥ 3.

3. Provide a method to automatically generate a body attached frame at the targeted area that gives the desired position and frame orientation of the end effector always normal to the bladder at the targeted area since the bladder tissue is anisotropic at high strain level.
4. Provide motion parameters for the robot using forward and a close-loop inverse differential kinematics modeling techniques and program algorithms.
5. Analytically develop the robot manipulator mechanics for mechanical design development. As a result, the tendon loading forces for a corresponding bending angle, tip force and the stress analysis on the joints are provided.
6. Development of a 3D solid model for preliminary prototyping of the various components of the conceptualized robot.

1.5 Thesis Outline

Chapter 2 provides a background to the design of the robotic system. Descriptions of various components of individual sub-sections of the robotic system are discussed. Additionally, a working principle of the robotic system is discussed. Furthermore, the process of prototyping the robotic system is introduced and compared to the initial design requirements.

Chapter 3 focuses on the kinematics of the robotic system. An analysis of the forward kinematics is provided using the Modified DenavitHartenberg (MDH) parameter approach. The inverse kinematics of the robot was accomplished using a close-loop inverse differential kinematic approach which allows for position and orientation control of the robot. Also, algorithms for the forward and inverse kinematic models were implemented in MATLAB and the output graphs showing the robot poses for arbitrary desired end effector position and orientation combinations were analyzed.

Chapter 4 shows analytically the static mechanics analysis for a robot flexible-continuum module using beam theory and assuming constant-curvature. Using this analysis, tendon loading and tip forces were obtained at 0^0 to 90^0 bending angle. Furthermore, standard clinically approved design materials for surgical instruments were consider to find an ideal structural design material for the concept robot using shear, bearing shear and torsional stresses on the pin joints.

Chapter 5 contains details of the methods used to prototype the flexible-continuum components of the proposed robotic system. Also provided is an assembly of the flexible-continuum components and functions. A detail working principle is enumerated.

Chapter 6 provides conclusions and recommendations for possible areas for future work and improvements. The appendices contain MATLAB programs and data of all analysis conducted in this research.

CHAPTER 2

ROBOT DESIGN AND PROTOTYPING

2.1 Design Considerations

As established in chapter 1, the bladder wall thickness and elasticity are dynamic and change with respect to the content of the bladder, the region measured and health of the bladder. Moreover, any minimally invasive way to get to the bladder in order to palpate the bladder wall with a force of 0.1 N has to be capable of a transurethra manipulation. Equivalently, there is a significant design size, workspace and actuation constraint. Therefore, the design objective of the proposed robot is flexibility, structural rigidity, miniaturization, under-actuation as well as being inexpensive.

The robot is conceptualized in multiple sections (as shown in figure 2.1); a flexible-continuum part with a force sensor tip, a rigid tube part and the actuation base.

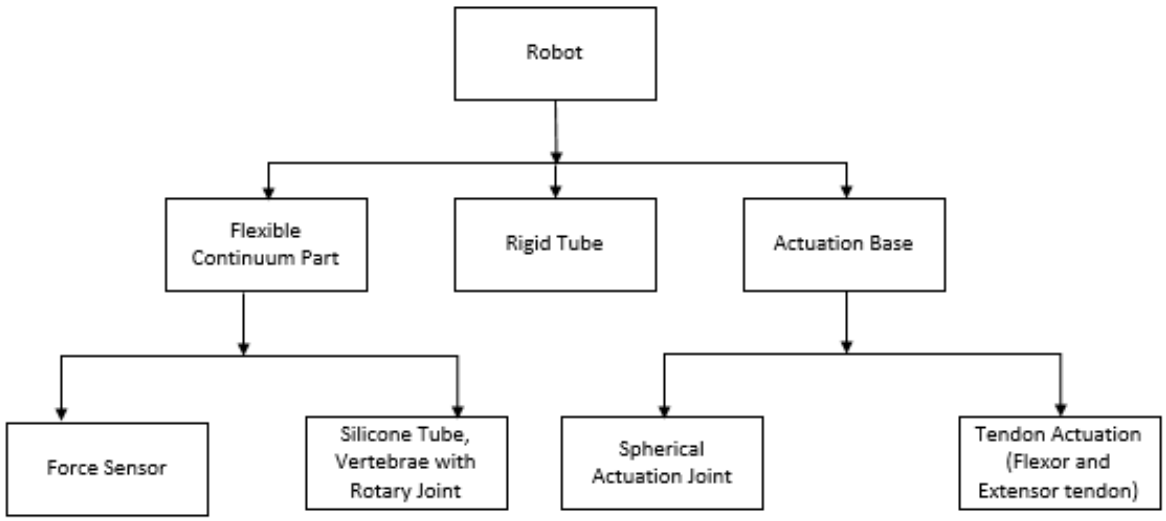


Figure 2.1: Robot Design Chart

2.2 General Description of the Concept Robot

The continuum robotic manipulator is presented in figure 2.2. The major structural components of the manipulator are the flexible-continuum part that is composed of multiple vertebra attached to each other at a pin joint, a silicone tube, a force sensor and several pins for the pin joint; a rigid tube with a tendon actuation mechanism that is composed of two V groove guide bearings and two actuators for the flexible-continuum part; a linear actuation mechanism, a hyper-spherical joint base that is composed of four actuators and a mounting base. The flexible-continuum part is attached to the rigid tube at a pin joint and the rigid tube is mounted to a linear motor which is in turn hosted by the hyper-spherical joint. The flexible-continuum part has a bending angle between -90^0 and $+90^0$ (when actuated by the tendon actuation mechanism) over a $\pm 300^0$ rotation. A linear and hyper-spherical actuation provide a means were the flexible-continuum part could be translated a distance of ± 60 mm and/or oriented at any point in three dimensional space. The hyper-spherical mechanism is mounted to a base frame which is intended to be fixed externally.

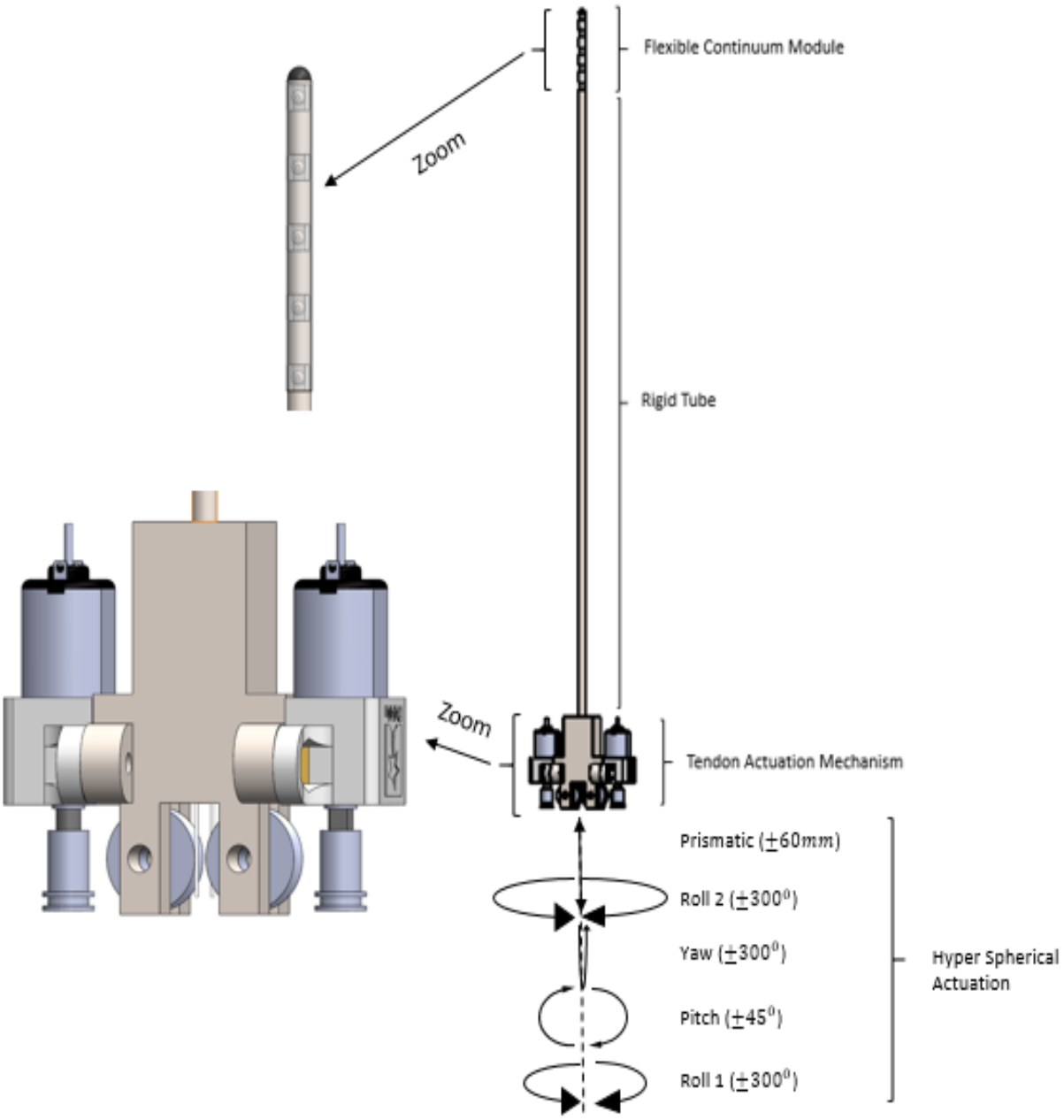


Figure 2.2: 3D of Assembled Robot

2.3 Robot Sub-Assembly Design Description

2.3.1 Flexible-Continuum Part

The flexible part of the robot (as shown in figure 2.3) comprises of multiple rigid links (called the vertebrae) attached at constrained rotational joints from a rigid tube to a sensor tip and enclosed in a flexible silicone tube. The flexible silicone tube allows actuation of the multiple links of this part of the robot by a single tendon, while taking advantage of the flexibility of the tube and the structural rigidity of the links when a pose is desired. The vertebrae are hollow cylinders each 2.95 mm in diameter and 5 mm long with rotary joint mount at the top and bottom face. The two rotary joint mounts are offset from each other to allow concentricity of vertebrae in the silicone tube. Each vertebra is attached to the next rotary joint with a pin. The hollow space in the cylindrical structure of each vertebra is used to route power and signal wires from the sensor at the tip of the continuum part to the base where all the control and actuation systems are located. Two tendons are also routed through individual vertebra through holes made in a 180^0 bi-configuration. One of these tendons is routed to an actuator and used as the flexor tendon to bend the continuum part when actuated while the other tendon is routed to another actuator and serves as extensor tendon to extend the continuum part to its initial straight pose. The flexible-continuum part is designed in such a way that it is modular and additional vertebrae could be added at each successive joint to increase the workspace of the robot. In this research, a design configuration with 4 successive vertebrae with a total length of 56 mm (also accounting for the force sensor geometry) are connected to the rigid tube is implemented and analyzed.

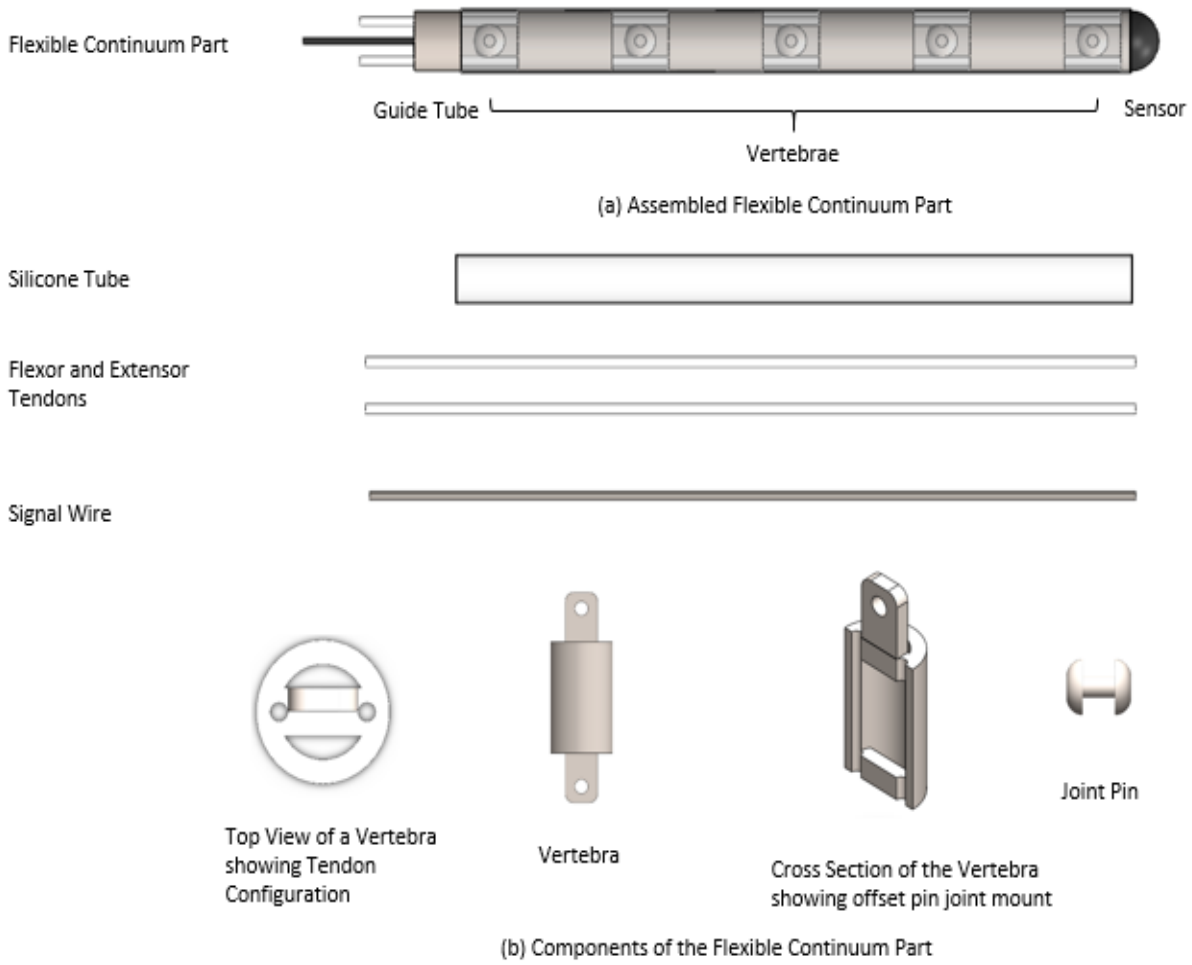


Figure 2.3: 3D Design of the Flexible-Continuum Part: (a) Assembled View, (b) Components of the Flexible-Continuum Part

2.3.1.1 Force Sensor

At the tip of the flexible-continuum part and attached at the rotary joint of the last vertebra is the force sensor (shown is figure 2.4). The force sensor measures the contact force when the robot palpates the bladder tissue wall. The force sensor is conceptualized in such a way that it provides adequate tissue-sensor contact surface area. A semi-spherical sensor

tip attached to a rotary joint mount is proposed as a smaller surface contact area from a conical, or tetrahedral surface may lead to a puncture in the bladder tissue when a force is applied on the bladder wall by the robot. [60]

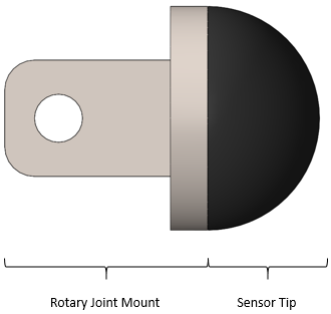


Figure 2.4: Geometric Concept of Force Sensor

A 3 mm force sensor where a strain gauge attached to a simple supported beam is deformed by load transmitting element to measure normal tissue reaction force has been proposed by Kumat at the University of Texas, Arlington MARS lab. [61] Therefore, the design and development of a 3 mm force sensor is not pursued in this research. However, a model was provided without any force measuring capacity to determine functionality and for use in analysis. This model has a semi-spherical sensor tip with a diameter of 2.95 mm as shown in figure 2.4. Also, the flexor and extensor tendons terminate at the base of the sensor, so when the tendons are actuated, the sensor rotates about its rotary joint.

2.3.2 Rigid tube

The flexible-continuum part of the robot is fixed to one end of the rigid tube at a rotary joint while the other end of the rigid tube has the flexor and extensor tendon actuation mount as well as a motorized lead screw mount for prismatic motion of the entire assembly (the

flexible-continuum part, the rigid tube and the rigid tube actuation mounts). As shown in figure 2.5, the rigid tube is 2.95 mm in diameter and 292.2 mm long. On either side of the tube, an actuator is mounted such that the extensor and flexor tendons could be routed from the flexible-continuum part to tendon pulleys through the V groove guide bearings to avoid rubbing against the internal wall of the tube, hence causing friction which will lead to an increase in required actuation force and wear of the tendon wires. The rigid tube provides structural support to the flexible-continuum part during manual insertion into the urethra. It also serve as a link to transfer manipulation of the end effector from the extrinsic hyper-spherical actuation mechanism.

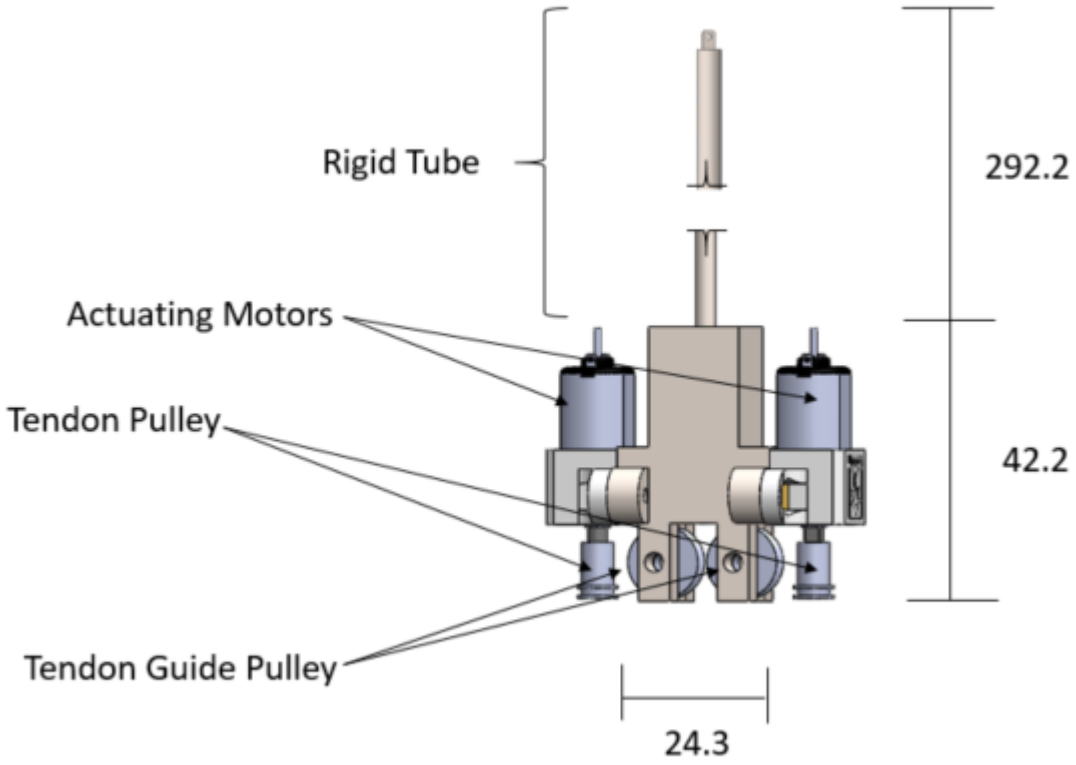


Figure 2.5: 3D Design Model of the Rigid tube and Overall Dimensions in mm

2.3.3 Actuation Base

Actuation means for the proposed robot is achieved in three parts; (1) the flexor and extensor tendon actuation, (2) a prismatic joint actuating the flexible-continuum module, the rigid tube and flexor-extensor tendon actuation mechanism and, (3) a hyper-spherical actuation joint actuating components (1) and (2). The flexor and extensor tendon actuation which is mounted on the rigid tube has already been described in section (2.3.2) while the hyper-spherical actuation joint consists of 2 roll, 1 yaw and 1 pitch joints. The 2 roll joints have a $\pm 300^0$ rotation limit to avoid tangling of electronic wires. The yaw and pitch joints each has a $\pm 45^0$ actuation limit so that discomfort in the patient is limited during the transurethra procedure. The prismatic joint is also limited to a $\pm$60mm linear motion. A kinematic analysis of the robotic system involving the hyper-spherical joint is investigated in this research but detailed mechanical design of this sub-part and the base frame has been omitted. The actuation base is illustrated in figure 2.5.

2.4 Chapter Summary

Chapter 2 provides details of the conceptualized robot intended for bladder elasticity dysfunction diagnosis. Firstly, a description of the design considerations and constraints used to arrive at the concept design was provided. Subsequently, details of the subparts of the robotic system were developed and described. Also, the actuation mechanisms were detailed including the robot motion limits. The resulting proposed robot has a flexible-continuum part with an outer diameter of 2.95 mm without the silicon tube (and 3 mm when included). The rigid tube also has an outer diameter of 2.95 mm. The overall length of the flexible-continuum and rigid tube part of the robot is 346.4 mm. Considering the geometric constraints of the bladder, the dimensions of the robot support an easy transurethra navi-

gation and manipulation. The hyper-spherical joint motion constraints was also considered to be important to reduce possible discomfort to patients during surgery.

CHAPTER 3

ROBOT KINEMATICS

3.1 Forward Kinematics

The mechanical design of the robot involves multi-bodies linked by several revolute and one prismatic joints. A total of ten joints are used, in which six joints serve as active joints and the remaining four serve as passive. This stems from the flexible-continuum part been driven on the principle of underactuation, allowing a limited number of joints to be actuatable directly (the sensor joint in this case) and the remaining flexible-continuum joints controlled by the structural elasticity of the encompassing silicone tube when actuated. Therefore, the kinematics of the robot was modeled as having 6 degrees of freedom (DoF) which is minimally required for arbitrary position and orientation control of the end-effector (sensor tip) of any robot in 3D space. [62]

3.1.1 Robot Joint Representation and Kinematics

The forward kinematics of the robot is essential to understanding the motion and behavior of the robot from specified joint parameters. To compute the forward kinematics of the robot, first, it is important to develop a kinematic diagram (shown in figure 3.1) of the robot so as to illustrate the configuration of the robot links and joints using body attached coordinate frames in Euclidean space. A global, or base, coordinate frame, $\{B\}$ is assigned at the center of the ellipsoid which will serve as a model of the bladder; a distance of 60 mm from the base of the ellipsoid (bladder opening) and would serve as a common reference for all other joints shown in section (3.1.2). Coordinate frames are assigned in such a way that the $z-$axis of each frame aligns with joint axis of motion and the $x-$axis of the current frame, x_C, is the

common perpendicular between the $z-$axis of the current frame, z_C, and the $z-$axis of the next frame, z_N, as shown in figure 3.1. This approach of assigning robot manipulator frames is called the Modified Denavit-Hartenberg, MDH approach. It is also important to state that all analysis in this research was performed using right-handed coordinate frames. [62]

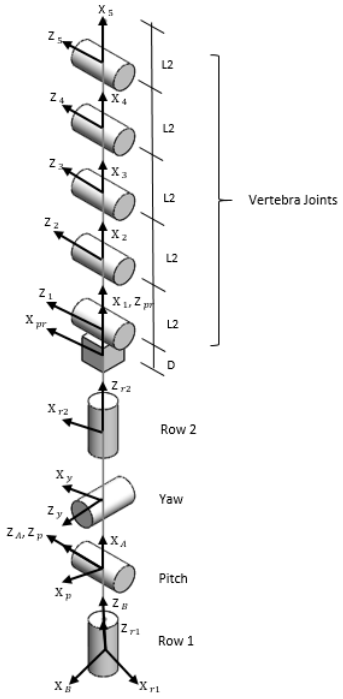


Figure 3.1: Robot Joint Representation and Frames

3.1.2 MDH Parameters and Table

A MDH table (shown in Table. 3.1) is used to describe the spatial relationship between successive link coordinate frames from only four parameters (a,α,d,θ) defined as:

1. a: is distance from z_C to z_N along x_C,
2. α: is the angle from z_C to z_N about x_C,

3. d: is the distance from x_C to x_N along z_C, and
4. θ: is the angle from x_C to x_N about z_C.

The parameters a and d define distance and α and θ define orientation between successive joint frames. In the case of revolute joints, parameters a, α, d are fixed while θ becomes the joint variable, and for prismatic joints, parameters a, α, θ are fixed while d becomes the joint variable.

Table 3.1: Robot MDH Table

Frame		Joints			MDH Parameters			
Current, C	Next, N	Frame #	Type	Variable	a	α	d	θ
B	R_1	1	R	θ	0	0	-60	θ_{R_1}
R_1	P	2	R	θ	0	90	0	θ_P
P	A	3	–	–	0	0	0	90
A	Y	4	R	θ	0	90	0	$\theta_Y + 90$
Y	R_2	5	R	θ	0	90	0	θ_{R_2}
R_2	Pr	6	P	d	0	0	D	0
Pr	1	7	R	θ	0	-90	0	$\theta/5 - 90$
1	2	8	R	θ	L_2	0	0	$\theta/5$
2	3	9	R	θ	L_2	0	0	$\theta/5$
3	4	10	R	θ	L_2	0	0	$\theta/5$
4	5	11	R	θ	L_2	0	0	$\theta/5$
5	6	12	–	–	0	0	L_2	0

3.1.3 Derivation of Link Transformations

The frame transformations are homogeneous transformation matrices that describe the orientation, R, and position vector, P, of each joint frame in 3D space with respect to the previous frame, or base frame, or any other frame. The position vector, P, is a 3×1 vector that locates any point or frame origin in space. Therefore, it is important to use proper notation that fully describes the point or frame origin and the reference from which it was measured. In this case, the notation similar to ${}^{A}_{B}P$ was adopted to represent the position vector of a point or frame origin, $\{B\}$, with respect to frame $\{A\}$, [62]

$$ {}^{A}_{B}P = \begin{bmatrix} P_x & P_y & P_z \end{bmatrix}^T \tag{3.1} $$

Where P_x, P_y and P_z are the x, y and z projections of the described point or frame origin on frame $\{A\}$.

Likewise, the orientation, R, describes the orientation of a body using a coordinate frame attached to the body. ${}^{A}_{B}R$ is a 3×3 rotation matrix that describes the orientation of a frame, $\{B\}$, with respect to another frame $\{A\}$,

$$ {}^{A}_{B}R = \begin{bmatrix} {}^{A}_{B}\hat{X} & {}^{A}_{B}\hat{Y} & {}^{A}_{B}\hat{Z} \end{bmatrix} \tag{3.2} $$

Where ${}^{A}_{B}\hat{X}$,${}^{A}_{B}\hat{Y}$ and ${}^{A}_{B}\hat{Z}$ are the projections of frame, $\{B\}$, onto frame, $\{A\}$, for the $x-$, $y-$ and $z-$axes respectively.

Individual link transformation matrices of the robot are computed by substituting values of a_C, α_N, d_N and θ_N from the MDH Table (Table 3.1) into the following homogeneous transformation matrix. The resulting transformations of frames 1 through 11 using equation (3.3) are provided in Appendix A.

$$
{}^{C}_{N}T = \begin{bmatrix} & {}^{C}_{N}R & & {}^{C}_{N}P \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta_N & sin\theta_N & 0 & a_C \\ \sin\theta_N \cdot \cos\theta_C & \cos\theta_N \cdot cos\theta_C & sin\theta_C & sin\theta_C \cdot d_N \\ \sin\theta_N \cdot sin\theta_C & \cos\theta_N \cdot sin\theta_C & \cos\theta_C & \cos\theta_C \cdot d_N \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}
$$

3.1.4 Robot Homogeneous Transformation Matrix

Therefore, the robot homogeneous transformation (a single transformation) matrix that relates the sensor tip (with attached frame 6) to the base frame, $\{B\}$, is obtained by post-multiplying all the transformation matrices of individual links.

$$
{}^{B}_{6}T = {}^{B}_{R_1}T \cdot {}^{R_1}_{P}T \cdot {}^{P}_{A}T \cdot {}^{A}_{Y}T \cdot {}^{Y}_{R_2}T \cdot {}^{R_2}_{Pr}T \cdot {}^{Pr}_{1}T \cdot {}^{1}_{2}T \cdot {}^{2}_{3}T \cdot {}^{3}_{4}T \cdot {}^{4}_{5}T \cdot {}^{5}_{6}T \tag{3.4}
$$

Using equation (3.4), the robot transformation matrix is computed as,

$$
{}^{B}_{6}T = \begin{bmatrix} R_{11} & R_{12} & R_{13} & P_x \\ R_{21} & R_{22} & R_{23} & P_y \\ R_{31} & R_{32} & R_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}
$$

Details of the entries for the individual elements of ${}^{B}_{6}T$ are also provided in Appendix A.

3.1.5 Forward Kinematics Algorithm

The computation of the forward kinematics of the robot was implemented as a program (provided in Appendix A) using MATLAB R2016a in the following algorithm. Also, an

ellipsoid was plotted to closely model the bladder as the workspace of the robot for effective visualization, computing and verifying the inverse kinematics of the robot.

1. Declare symbolic variables,
2. Initialize variables,
3. Populate MDH table and save as a cell matrix, M_T,
4. For $i = 1$: number of rows of M_T,
   (a) Extract MDH parameters from cell matrix, M_T,
   (b) Compute homogeneous transforms of each joint MDH parameters, ${}^{i-1}_{i}T$.
5. Obtain homogeneous transforms of each joint with respect to base frame, ${}^{B}_{i}T$,
6. Using the position vector of each link transformation matrix with respect to base frame, ${}^{B}_{i}T$, plot the robot,
7. Plot ellipsoid,

3.2 Inverse Kinematics

So far, the mathematical model for the forward kinematics of the robot has been obtained. This model describes the robot end-effector as a function of the individual joint and geometric parameters given specified joint variables. However, suppose a goal or desired pose, a cartesian position and/or orientation of the end-effector is required. In this case, the analysis of the forward kinematics alone will be insufficient to solve this. This is because it is impossible to manually predict what joint configurations will give a solution (except in trivial cases). However, a solution could be obtained with inverse kinematics.

In this research, an inverse differential kinematics method is used (as described in sections 3.2.1 - 3.2.3) to control both the position and orientation of the end-effector (sensor tip) of the robot. Since the designed robot has 6 active joints and 4 passive joints, the position control utilizes 3 joint variables while the orientation control uses the remaining 3 joint variables of the active joints. Central to developing the mathematical model of the inverse

differential kinematics is the analytical Jacobian matrix, J_A. The analytical Jacobian matrix, J_A, is populated by two other Jacobians (i.e the translational velocity Jacobian, J_V, and the angular velocity Jacobian, J_ω). The translational velocity Jacobian, J_V, is computed for the position control phase and, likewise, the angular velocity Jacobian, J_ω, is computed for the orientation control phrase. [63]

3.2.1 Position Control

Suppose a manipulator end-effector position at a given joint configuration, $\vec{q}$, is $\vec{P}$. Also, suppose there is an infinitesimally small change in the joint configuration with respect to time denoted by $\frac{\delta\vec{q}}{\delta t}$, there is a corresponding change in the end-effector position denoted as $\frac{\delta\vec{P}}{\delta t}$. Establishing a relationship between $\frac{\delta\vec{q}}{\delta t}$ and $\frac{\delta\vec{P}}{\delta t}$ is the aim of position control. The time derivative of the position vector gives the velocity vector, $\vec{V}$ (as in Equation 3.6). However, the velocity vector can be expressed as a function of the joint configuration vector, $\vec{q}$. Therefore, a relationship describing how the change in joint space affects the change in position of the end-effector in cartesian space is obtained in Equation 3.6 and the proportionality term $\frac{\delta\vec{P}}{\delta\vec{q}}$ is called the velocity Jacobian, J_V. Once the velocity Jacobian is obtained, imagine that the desired position is provided and there is a need to know what joint configuration is required for the end-effector to reach the specified position from its current position, multiplying the inverse of the velocity Jacobian by the absolute position difference provides the needed solution (as shown in Equation 3.7). For non-square or non-invertible velocity Jacobian matrix, a pseudo-inverse of the velocity Jacobian is employed. It is also important to state that, unlike other methods of inverse kinematics computation, the inverse differential kinematics approach could provide a unique solution for position control in real time as the robot moves to the desired end effector pose. [64, 65]

$$\vec{V} = \frac{\delta\vec{P}}{\delta t} = \frac{\delta\vec{P}}{\delta\vec{q}} \cdot \frac{\delta\vec{q}}{\delta t} = J_V \frac{\delta\vec{q}}{\delta t} \tag{3.6}$$

Also,

$$\delta\vec{q}_{n\times 1} = {J_V}^{-1}{}_{n\times 3} \cdot \delta\vec{P}_{3\times 1} \tag{3.7}$$

Where n is the number of joints.

To compute the velocity Jacobian of the robot, it is necessary to extract the position vector of robot end-effector, ${}^{B}_{6}P$, from the robot homogeneous transformation matrix, ${}^{B}_{6}T$, (in Equation 3.5) and take the partial derivatives of the $x-$, $y-$ and $z-$ components with respect to each of the joint variables (θ_{R1}, θ_P, θ_Y, θ_{R2}, D and θ) symbolically (as shown in equation 3.8). [63]

$$J_V = \begin{bmatrix} \frac{\delta P_x}{\delta\theta_{R1}} & \frac{\delta P_x}{\delta\theta_P} & \frac{\delta P_x}{\delta\theta_Y} & \frac{\delta P_x}{\delta\theta_{R2}} & \frac{\delta P_x}{\delta D} & \frac{\delta P_x}{\delta\theta} \\ \frac{\delta P_y}{\delta\theta_{R1}} & \frac{\delta P_y}{\delta\theta_P} & \frac{\delta P_y}{\delta\theta_Y} & \frac{\delta P_y}{\delta\theta_{R2}} & \frac{\delta P_y}{\delta D} & \frac{\delta P_y}{\delta\theta} \\ \frac{\delta P_z}{\delta\theta_{R1}} & \frac{\delta P_z}{\delta\theta_P} & \frac{\delta P_z}{\delta\theta_Y} & \frac{\delta P_x}{\delta\theta_{R2}} & \frac{\delta P_z}{\delta D} & \frac{\delta P_z}{\delta\theta} \end{bmatrix} \tag{3.8}$$

3.2.2 Position Control Algorithm

The computation of the inverse position kinematics of the robot was implemented as a program (provided in Appendix A) using MATLAB R2016a according to the following algorithm. Also, an ellipsoid was plotted to closely model the bladder as the workspace of the robot from which the desired position is obtained, and also for effective visualization. A scalar called the step size, s, is used to divide the iterative change in joint parameters towards the desired pose. s is used to discretize the vector error between the current robot pose and the desired robot pose to improve the solution and is chosen to be small enough so that the end effector is restricted from 'jumping pass' the goal position and large enough so that the computation time of the program is not unnecessary longer. [63] $e = 0.01$ (which is a very small error for the purpose of this thesis) is the permissive error of the solution before the iteration is stopped.

1. Obtain desired position, ${}^{B}_{6}P_D = [D_x, D_y, D_z]^T$,
2. Obtain robot home position vector, ${}^{B}_{6}P_C = [P_x, P_y, P_z]^T$, at joint variables $\theta = 0$,
3. Initialize change in joint variables to zero, $\Delta\theta = 0$,
4. While $|{}^{B}_{6}P_D - {}^{B}_{6}P_C| > e$ do
   (a) $Dp = |{}^{B}_{6}P_D - {}^{B}_{6}P_C|/s$, where s is the desired precision of increment
   (b) Apply joint variable change, $\theta = \theta + \Delta\theta$
   (c) If θ exceeds joint range, apply joint limit
   (d) Update position vector of ${}^{B}_{6}P_C$ using new θ
   (e) Compute Jacobian, J_v
   (f) Invert Jacobian matrix using pseudo inverse of the Jacobian, $J_p = ({J_V}^T J_V)^{-1} {J_V}^T$
   (g) Compute change in joint variables, $\Delta\theta = J_p Dp$
5. Repeat until solution is within desired error, e.

Suppose an end effector desired position, ${}^{B}_{6}P_D = [D_x, D_y, D_z]^T = [-36.68, 88.26, 84.4]$ is required, the motion of the end effector to the position goal as function of iterations for each joint variable is shown in the figure 3.2. The position control behavior shows the robot joint variable motion history without discontinuity. Therefore, the position control algorithm creates a smooth motion path of the robot joint as the robot tends towards the desired robot pose.

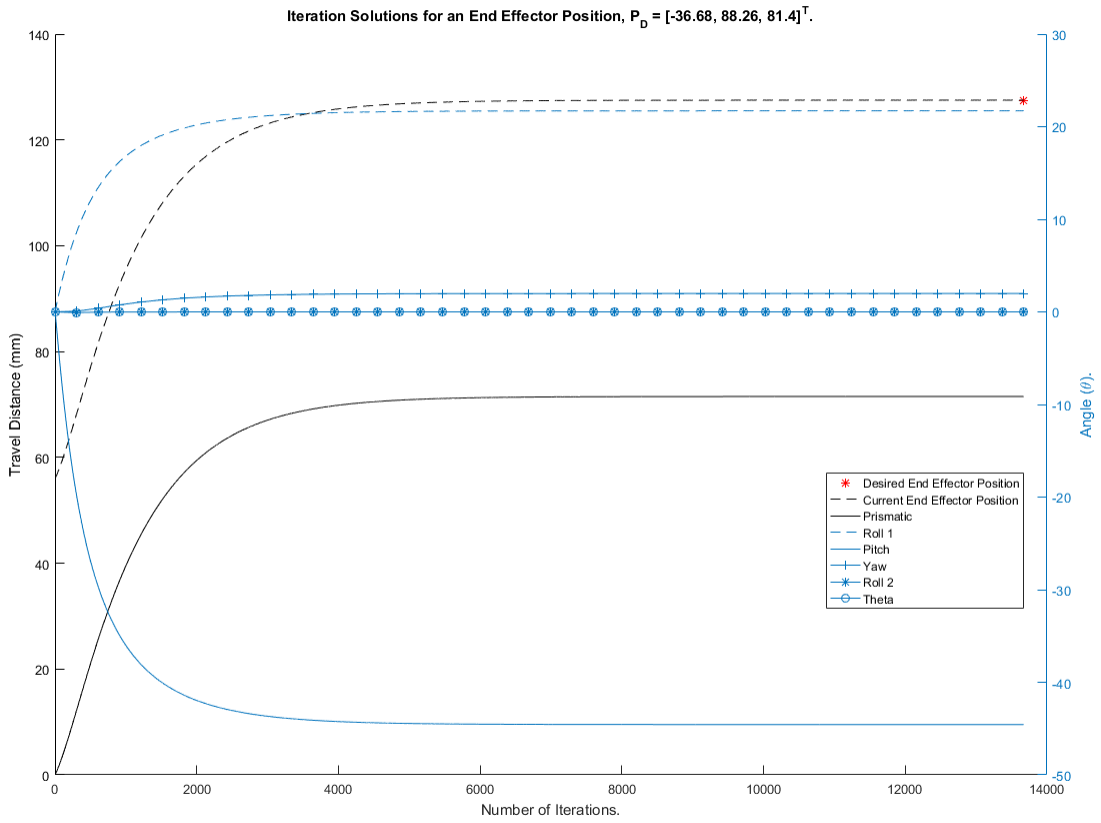


Figure 3.2: End Effector Motion to Desired Position, ${}^{B}_{6}P_{D} = [-36.68, 88.26, 84.4]^{T}$ with all Iterative Solutions

3.2.3 Orientation Control

Suppose a specified end-effector orientation is desired, a mathematical model similar to that used in the position control is required to develop a Jacobian relationship. However, this Jacobian relationship is not a straightforward process as, first, the angular velocity, $\vec{\omega}$, of the robot links must be obtained, followed by the angular velocity Jacobian, J_{ω}, then the end-effector orientation vector, $\vec{\alpha}$, is computed. The relationship between the angular velocity,

$\vec{\omega}$, and the end-effector orientation vector, $\vec{\alpha}$, helps obtain a useful Jacobian, J_α, that will be required to obtain the robot end-effector orientation in task space from its joint space. This is the relationship required for the orientation control. [66, 67, 68, 65, 69]

3.2.3.1 Angular Velocity, $\vec{\omega}$, and Augular Velocity Jacobian, J_ω

A manipulator with multiple link joints usually has a prismatic, or revolute joint or a combination of both as noted in Chapter 2. The $z-$ axis is defined as the axis of motion of the frames attached to individual joints with respect to the base frame of the robot. The contribution of the individual joints to the angular velocity of the robot end-effector is evaluated. The angular velocity for a prismatic joint is $\omega = 0$, while that of a revolute joint is $\omega = Z_i\dot{q}$. Where Z_i is the $z-$ axis projection of the ith joint with respect to the base frame. [63, 67] In general, the angular velocity, ${}^0_n\vec{\omega}$ of the end-effector of a robot with n degrees of freedom can be expressed as,

$$ {}^0_n\vec{\omega} = \rho_1 \, {}^0_0Z\dot{q}_1 + \rho_2 \, {}^0_1Z\dot{q}_2 + ... + \rho_n \, {}^0_{n-1}Z\dot{q}_n \tag{3.9} $$

such that,

$$ {}^0_n\vec{\omega} = \begin{bmatrix} \rho_1 \, {}^0_0Z & \rho_2 \, {}^0_1Z & \cdots & \rho_n \, {}^0_{n-1}Z \end{bmatrix} \cdot \begin{bmatrix} \dot{q}_1 & \dot{q}_2 & \cdots & \dot{q}_n \end{bmatrix}^T \tag{3.10} $$

Where $\rho_i(i = 1, 2, ..., n) = 1$ for a revolute joint and 0 for a prismatic joint. ${}^0_0Z$ is always equal to $[0 \quad 0 \quad 1]^T$. Therefore, the angular velocity Jacobian, J_ω can be extracted as,

$$ \begin{aligned} {}^0_n\vec{\omega} &= J_\omega \cdot \dot{\vec{q}}, \\ J_\omega &= \begin{bmatrix} \rho_1 \, {}^0_0Z & \rho_2 \, {}^0_1Z & \cdots & \rho_n \, {}^0_{n-1}Z \end{bmatrix} \end{aligned} \tag{3.11} $$

In the case of the designed robot, there's one prismatic joint and five active revolute joints. Therefore, $\rho_1 = 1$, $\rho_2 = 1$, $\rho_3 = 1$, $\rho_4 = 1$, $\rho_5 = 0$, $\rho_6 = 1$ and the angular velocity Jacobian, J_ω is computed as, [62]

$$J_\omega = \begin{bmatrix} \rho_1 \, {}^{B}_{B}Z & \rho_2 \, {}^{B}_{R1}Z & \rho_3 \, {}^{B}_{P}Z & \rho_4 \, {}^{B}_{Y}Z & \rho_5 \, {}^{B}_{R2}Z & \rho_6 \, {}^{B}_{Pr}Z \end{bmatrix} \tag{3.12}$$

Where,

$$\rho_1 \, {}^{B}_{B}Z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{3.13}$$

$$\rho_2 \, {}^{B}_{R1}Z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{3.14}$$

$$\rho_3 \, {}^{B}_{P}Z = \begin{bmatrix} \sin\theta_{R1} & -\cos\theta_{R1} & 0 \end{bmatrix}^T \tag{3.15}$$

$$\rho_4 \, {}^{B}_{Y}Z = \begin{bmatrix} \cos\theta_P \cos\theta_{R1} & \cos\theta_P \sin\theta_{R1} & \sin\theta_P \end{bmatrix}^T \tag{3.16}$$

$$\rho_5 \, {}^{B}_{R2}Z = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{3.17}$$

$$\rho_6 \, {}^{B}_{Pr}Z = \begin{bmatrix} -\cos(\frac{\pi}{2} + \theta_Y)\sin\theta_{R1} - \cos\theta_{R1}\sin\theta_P \sin(\frac{\pi}{2} + \theta_Y) \\ \cos\theta_{R1}\cos(\frac{\pi}{2} + \theta_Y) - \sin\theta_P \sin\theta_{R1}\sin(\frac{\pi}{2} + \theta_Y) \\ \cos\theta_P \sin(\frac{\pi}{2} + \theta_Y) \end{bmatrix} \tag{3.18}$$

3.2.3.2 End-Effector Orientation Vector, $\vec{\alpha}$, and Orientation Jacobian, J_α

Assume that a manipulator end-effector orientation at a given joint configuration, $\vec{q}$, is $\vec{\alpha}$. The orientation vector, $\vec{\alpha}$ is a 3×1 vector obtained when the robot rotation matrix,

R (a 3×3 vector) is decomposed into 3 elementary rotations called Euler angles. The $Z(\phi) - Y(\theta) - Z(\psi)$ Euler angle convention used in this research is derived in the following order; [62]

$$ {}^{0}_{n}R \equiv R_Z(\phi) R'_Y(\theta) R''_Z(\psi) \tag{3.19} $$

$$ {}^{0}_{n}R \equiv \begin{bmatrix} \cos\phi & sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & \cos\theta \end{bmatrix} \times \begin{bmatrix} \cos\psi & sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.20} $$

Therefore,

$$ \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \equiv \begin{bmatrix} \cos\phi\cos\theta\cos\psi - \sin\phi\sin\psi & -\cos\psi\sin\phi - \cos\phi\cos\theta\sin\psi & \cos\phi\sin\theta \\ \cos\phi\sin\psi + \cos\theta\cos\psi\sin\phi & \cos\phi\cos\psi - \cos\theta\sin\phi\sin\psi & \sin\phi\sin\theta \\ -\sin\theta\sin\psi & \cos\psi\sin\theta & \cos\theta \end{bmatrix} \tag{3.21} $$

Comparing the elements of the LHS and RHS gives the extracted values of the 3 elementary rotations, ϕ, θ and ψ as follows,

If $\sin\theta \neq 0$,

$$ \begin{aligned} \theta &= \arctan 2\Big(\sqrt{r_{31}^2 + r_{32}^2}, r_{33}\Big), \\ \phi &= \arctan 2\Big(\frac{r_{23}}{\sin\theta}, \frac{r_{13}}{\sin\theta}\Big), \\ \psi &= \arctan 2\Big(\frac{r_{32}}{\sin\theta}, -\frac{r_{31}}{\sin\theta}\Big), \end{aligned} \tag{3.22} $$

If $\theta = 0.0$,

$$\begin{aligned} \theta &= 0.0, \\ \phi &= 0.0, \\ \psi &= \arctan 2(-r_{12}, r_{11}), \end{aligned} \tag{3.23}$$

If $\theta = 180^0$,

$$\begin{aligned} \theta &= 180^0, \\ \phi &= 0.0, \\ \psi &= \arctan 2(r_{12}, -r_{11}), \end{aligned} \tag{3.24}$$

so that the orientation as a function of the robot joint variables is,

$$ {}^0_n\vec{\alpha}(q) = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \tag{3.25}$$

Once this is evaluated, it is observed that an infinitesimal small change in the joint configuration with respect to time denoted by $\frac{\delta \vec{q}}{\delta t}$ will result in a corresponding change in the end-effector orientation, $\frac{\delta \vec{\alpha}}{\delta t}$. Also, a relationship between $\frac{\delta \vec{\alpha}}{\delta t}$ and $\frac{\delta \vec{q}}{\delta t}$ can be rewritten as,

$$\frac{\delta \vec{\alpha}}{\delta t} = \frac{\delta \vec{\alpha}}{\delta \vec{q}} \cdot \frac{\delta \vec{q}}{\delta t} = J_\alpha \cdot \frac{\delta \vec{q}}{\delta t} \tag{3.26}$$

Where $J_\alpha = \frac{\delta \vec{\alpha}}{\delta \vec{q}}$ is the orientation Jacobian.

Establishing a relationship between the time change in joint configuration, $\frac{\delta \vec{q}}{\delta t}$, the orientation Jacobian, J_α, the time change in the end-effector orientaton, $\frac{\delta \vec{\alpha}(q),}{\delta t}$ and the angular velocity, ${}^0_n\vec{\omega}$, will provide the orientation control solutions. The relationship between ${}^0_n\vec{\omega}$ and $\frac{\delta \vec{\alpha}(q)}{\delta t}$ is obtained as follows, [67] Let $S({}^0_n\vec{\omega}) = ({}^0_n\dot{R})({}^0_nR)^T$ be a skew-symmetric matrix. Such that,

$$ {}^0_n\dot{R} = \dot{\phi}\frac{d}{d\phi}(R_Z(\phi))R_{Y'}(\theta)R_{Z''}(\psi) + R_Z(\phi)\dot{\theta}\frac{d}{d\theta}(R_{Y'}(\theta))R_{Z''}(\psi) + R_Z(\phi)R_{Y'}(\theta)\dot{\psi}\frac{d}{d\theta}(R_{Z''}(\psi)) \tag{3.27}$$

and,

$$ {}^0_nR^T = R^T_Z(\phi)R^T_{Y'}(\theta)R^T_{Z''}(\psi) \tag{3.28}$$

Therefore,

$$S({}^{0}_{n}\vec{\omega}) = \begin{bmatrix} 0 & -(\dot{\psi}\cos\theta + \dot{\phi}) & \dot{\theta}\cos\phi + \dot{\psi}\sin\phi\sin\theta \\ \dot{\psi}\cos\theta + \dot{\phi} & 0 & \dot{\theta}\sin\phi - \dot{\psi}\cos\phi\sin\theta \\ -(\dot{\theta}\cos\phi + \dot{\psi}\sin\phi\sin\theta) & -(\dot{\theta}\sin\phi - \dot{\psi}\cos\phi\sin\theta) & 0 \end{bmatrix} \tag{3.29}$$

And the values for ${}^{0}_{n}\vec{\omega}$ are extracted as,

$${}^{0}_{n}\vec{\omega} = \begin{bmatrix} \dot{\psi}\cos\phi\sin\theta - \dot{\theta}\sin\phi \\ \dot{\psi}\sin\phi\sin\theta + \dot{\theta}\cos\phi \\ \dot{\psi}\cos\theta + \dot{\phi} \end{bmatrix} = \begin{bmatrix} \cos\psi\sin\theta & -\sin\psi & 0 \\ \sin\psi\sin\theta & \cos\psi & 0 \\ \cos\theta & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = A({}^{0}_{n}\vec{\alpha}) \cdot {}^{0}_{n}\dot{\vec{\alpha}} \tag{3.30}$$

Where $A({}^{0}_{n}\vec{\alpha})$ is a multiplying matrix which is a function of ${}^{0}_{n}\vec{\alpha}$.

Examing equations 3.11, 3.26 and 3.30, it is observed that ${}^{0}_{n}\vec{\omega} \equiv J_\omega \cdot \dot{\vec{q}} \equiv A({}^{0}_{n}\vec{\alpha}) \cdot {}^{0}_{n}\dot{\vec{\alpha}}$,

$$J_\omega \cdot \dot{\vec{q}} \equiv A({}^{0}_{n}\vec{\alpha}) \cdot {}^{0}_{n}\dot{\vec{\alpha}} \tag{3.31}$$

and,

$$\begin{gathered} J_\omega \cdot \dot{\vec{q}} \equiv A({}^{0}_{n}\vec{\alpha}) \cdot J_\alpha \cdot \dot{\vec{q}}, \\ J_\omega \equiv A({}^{0}_{n}\vec{\alpha}) \cdot J_\alpha \end{gathered} \tag{3.32}$$

Therefore,

$$J_\alpha \equiv A({}^{0}_{n}\vec{\alpha})^{-1} \cdot J_\omega \tag{3.33}$$

Once this is obtained, the inverse orientation kinematics is modeled using equations (3.26) and (3.33) by evaluating,

$$ {}^{0}_{n}\dot{\vec{\alpha}} = A({}^{0}_{n}\vec{\alpha})^{-1} \cdot J_{\omega} \cdot \dot{\vec{q}} \tag{3.34} $$

Yielding,

$$ \delta\vec{q}_{n\times 1} = (A({}^{0}_{n}\vec{\alpha}) \cdot {J_{\omega}}^{-1})_{n\times 3} \cdot \delta\, {}^{0}_{n}\vec{\alpha}_{3\times 1} \tag{3.35} $$

Equation 3.35 is a relationship describing how the change in joint space affects the change in orientation of the end-effector in cartesian space. Now assume that the desired orientation is provided and there is a need to know what joint configuration is required for the end-effector to be rotated to the specified orientation about the reference base frame. Multiplying the inverse of the angular velocity Jacobian and the $A({}^{0}_{n}\vec{\alpha})$ matrix by the absolute orientation difference provides the needed solution (as shown in equation 3.35). For non-square or non-invertible angular velocity Jacobian matrix, a pseudo-inverse of the angular velocity Jacobian is evaluated. This approach also provides an iterative solution for orientation control in real time as the robot moves to the desired end effector pose. [70, 66, 67]

3.2.4 Obtaining the Desired Orientation

As discussed, an ellipsoid is always plotted along with the pose of the robot in the inverse kinematics plots. When a point on the interior surface of the bladder is desired, the data cursor tool in MATLAB is used to click on the desired point and a body attached frame is computed and placed on the surface with its origin where that point is situated as describe in section 3.2.4.1. The desired orientation frame is such that the $x-$ axis is always collinear to the outward pointing normal of the ellipsoid.

3.2.4.1 Ellipsoid

First, an ellipsoid is modeled with the semi-principal axes of lengths $a = 60mm, b = 130mm, c = 60mm$ using the mean length, width and height of the bladder computed by Fay A. at 902.0 cm^3 fill volume. [71] The center point of the ellipsoid is at $(x_0, y_0, z_0) = (0, 0, 0)$ (shown in figure 3.3) and the ellipsoid equation represented by equation 3.36.

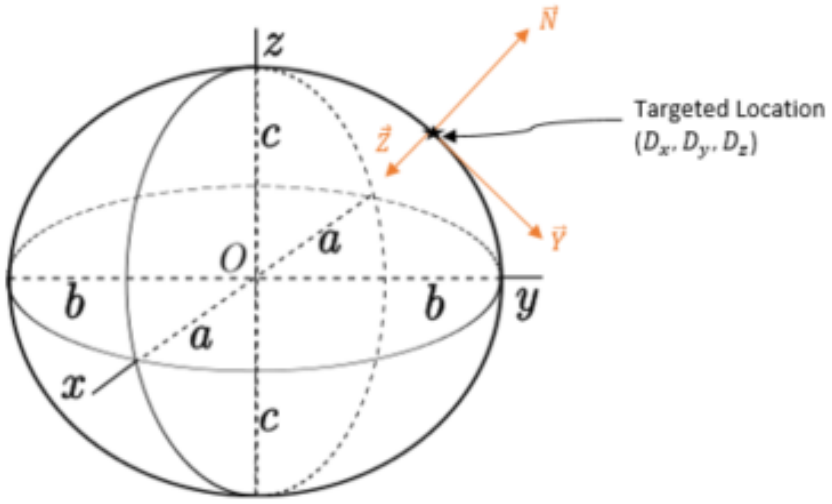


Figure 3.3: Ellipsoid with semi-principal axes[72]

$$\frac{(x - x_0)^2}{a^2} + \frac{(y + y_0)^2}{b^2} + \frac{(z - z_0)^2}{c^2} = 1 \tag{3.36}$$

3.2.4.2 Normal to an Ellipsoid, $\vec{N}$

Now, imagine a desired point defined by $D = [D_x, D_y, D_z]$ is selected on the surface of the ellipsoid, a frame with the desired orientation is evaluated after finding the outward pointing

normal vector at that point using equation 3.37 and its gradient and then substituting the desired point into equation 3.37 as shown in equation 3.38.

$$f(x,y,z) = \frac{x^2}{60^2} + \frac{y^2}{130^2} + \frac{z^2}{60^2} - 1 = 0 \tag{3.37}$$

$$\vec{N}(D_x, D_y, D_z) \equiv \nabla f(D_x, D_y, D_z) = 2\Big(\frac{D_x}{60^2}\hat{i} + \frac{D_y}{130^2}\hat{j} + \frac{D_z}{60^2}\hat{k}\Big) \tag{3.38}$$

3.2.4.3 Desired Frame from the Orthogonal Basis of the $\vec{N}$ Vector

The normal vector, $\vec{N}$, is assumed to be the vector representing the projection of the $x-$ axis of the desired orientation frame to the reference frame, $\{B\}$. Since this normal has an infinite number of perpendicular axes such that the dot product of any perpendicular axis and the normal projections is zero. Therefore, the $y-$ axis projection of the desired orientation frame, $\vec{Y}$, to the reference frame, $\{B\}$, is obtained by finding the orthogonal basis of $\vec{N}$ using null space, so that;

$$\vec{Y} = y_1\hat{i} + y_2\hat{j} + y_3\hat{k} \tag{3.39}$$

and

$$\vec{N} \cdot \vec{Y} = N_x y_1 + N_y y_2 + N_z y_3 = 0 \tag{3.40}$$

also, the $z-$ axis projection of the desired orientation frame, $\vec{z}$, to the reference frame, $\{B\}$, is obtained by finding cross product of $\vec{N}$ and $\vec{Y}$ to get,

$$\vec{z} \equiv \vec{N} \times \vec{Y} = \begin{bmatrix} \hat{i} & \hat{j} & \hat{k} \\ N_x & N_y & N_z \\ y_1 & y_2 & y_3 \end{bmatrix} = (N_y y_3 - N_z y_2)\hat{i} - (N_x y_3 - N_z y_1)\hat{j} + (N_x y_2 - N_y y_1)\hat{k} \tag{3.41}$$

and the rotation matrix representing the orientation of the desired frame orientation at the desired point is populated as,

$$ {}^{B}_{n}\alpha_{(desired)} = \begin{bmatrix} N_x & N_y & N_z \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{bmatrix} \tag{3.42} $$

3.2.5 Orientation Control Algorithm

The computation of the inverse orientation kinematics of the robot was implemented as a program (provided in Appendix A) using MATLAB R2016a in the following algorithm.

1. Obtain desired position, ${}^{B}_{6}P_D = [D_x, D_y, D_z]^T$,
2. Obtain robot home position vector, ${}^{B}_{6}P_C = [P_x, P_y, P_z]^T$ at joint variables, $\theta = 0$,
3. Initialize change in joint variables to zero, $\Delta\theta = 0$,
4. Compute current rotation matrix, ${}^{B}_{6}R$ of the robot extracted from the forward kinematics,
5. Compute current orientation, ${}^{B}_{6}\alpha_c$ of the robot,
6. Compute the normal vector and desired orientation, ${}^{B}_{6}\alpha_d$ at the desired point using desired position,
7. While $|{}^{B}_{6}\alpha_d - {}^{B}_{6}\alpha_c| > e$ do
   (a) $Do = \frac{|{}^{B}_{6}\alpha_d - {}^{B}_{6}\alpha_c|}{s}$, where s is the desired step size
   (b) Apply change joint variables, $\theta = \theta + \Delta\theta$
   (c) If θ exceeds joint range, apply joint limit
   (d) Update orientation vector of ${}^{B}_{6}\alpha_c$ using new θ
   (e) Compute Orientation Jacobian, J_α
   (f) Invert Orientation Jacobian matrix using pseudo inverse of the Jacobian, $J_{\alpha p} = (J_\alpha^T J_\alpha)^{-1} J_\alpha^T$

(g) Compute change in joint variables, $\Delta\theta = J_{\alpha p} Do$

8. Repeat until solution is within desired error, e.

3.2.6 Position and Orientation Control

Obtaining position and orientation control of the end-effector requires a combination of both the position and orientation control components derived in equations 3.6 and 3.34. Using these equations, the robot pose inverse differential kinematics equation can be presented in the following,

$$\dot{\vec{x}}_G = \begin{bmatrix} {}^0_n\vec{V} & {}^0_n\vec{\omega} \end{bmatrix}^T = J_G(q) \cdot \dot{q} = \begin{bmatrix} J_V & J_\omega \end{bmatrix}^T \cdot \dot{q} \tag{3.43}$$

Where $\dot{\vec{x}}_G$ is the geometric solution, and $J_G(q)$ is the geometric Jacobian. These are computed using the geometric procedure as a result of the joint velocity contributions of each joint to the end-effector translational and angular velocities.

Also,

$$\dot{\vec{x}}_{A(n\times 1)} = \begin{bmatrix} {}^0_n\vec{V} \\ \\ {}^0_n\vec{\alpha} \end{bmatrix} = J_A(q) \cdot \dot{\vec{q}} = \begin{bmatrix} J_V \\ \\ A({}^0_n\vec{\alpha})^{-1} \cdot J_\omega \end{bmatrix} \cdot \dot{\vec{q}}_{(n\times 1)} \tag{3.44}$$

Where $\dot{\vec{x}}_A$ is the analytical solution, and $J_A(q)$ is the analytical Jacobian, computed as a result of the joint velocity contributions of each joint to the end-effector translational velocities and orientation change. Again, equation 3.44 was used to obtain the overall pose of the robot because the robot position and orientation control are of concern. Multiplying the change in position and orientation vector, $\dot{\vec{x}}_A$, and the pseudoinverse analytical Jacobian, $J_A(q)$, produces the desired joint configuration, $\dot{\vec{q}}$.

3.2.6.1 Enhancing the Position and Orientation Control Solution

As the robot kinematics is computed by mapping task space velocities into joint velocities iteratively, there's a likely introduction of task error from large joint parameter gains, especially, when the current pose of the end-effector is far from the goal. Therefore, an improved pseudoinverse Jacobian proposed by Chan et al. to impose an error damping condition on the pseudoinverse Jacobian. The improved pseudoinverse Jacobian is called Error Damped Pseudoinverse Jacobian, $J^{\dagger ED}$ defined as, [73, 70, 74]

$$J^{\dagger ED} = J^T(JJ^T + EI_m)^{-1} \tag{3.45}$$

Where $E = \frac{1}{2}[D\vec{p}, D\vec{\alpha}] \cdot [D\vec{p}, D\vec{\alpha}]^T$.

Also, a close loop error was implemented to impose an iterative process until a solution is obtained where the final position is within a sphere with a radius error of 0.03 MSE from the desired position and a final orientation which has an error less than 0.03^0 MSE. [73]

3.2.6.2 Model Solution Error, E_m

In some cases, a solution is not possible and the solution iteration goes into an infinite loop. Therefore, an iteration time limit of 400 seconds was implemented in MATLAB to terminate execution of the program loop and the current iteration solution is acquired. To determine the effectiveness of the mathematical solution for the position and orientation inverse kinematics, it was necessary to observe the solution error as suggested by Colome et al.; i.e how close is the final end effector pose from the desired one. [73] The position goal error, P_G, and orientation goal error, O_G are found respectively as,

$$P_G = \|D - P\|^2, \tag{3.46}$$

$$O_G = (\phi_{(current)} \cdot \phi_{(desired)} - 1)^2 + (\theta_{(current)} \cdot \theta_{(desired)} - 1)^2 + (\psi_{(current)} \cdot \psi_{(desired)} - 1)^2, \tag{3.47}$$

and model solution error, E_m, is

$$E_m = \begin{bmatrix} P_G \\ \\ O_G \end{bmatrix} \tag{3.48}$$

3.2.7 Improved Position and Orientation Control Algorithm

To enhance the solution towards the desired pose, the position and orientation algorithm develops a damping error, E, for every $(i-1)$th iteration to compute the ith iteration Jacobian called the error damped Jacobian. This Jacobian is different from the former Jacobian used in section .

1. Obtain Transformation matrix of the robot. See equation 3.5,
2. Initialize current joint variables to zero, $\theta = 0$,
3. Initialize change in joint variables to zero, $\Delta\theta = 0$,
4. Compute initial rotation matrix, ${}^{B}_{6}R(\theta)$ of the robot extracted from the forward kinematics,
5. Compute initial orientation from the rotation matrix using Euler angles, ${}^{B}_{6}\alpha_c(\theta)$ of the robot. See equation 3.25,
6. Compute the normal vector and desired orientation, ${}^{B}_{6}\alpha_d(\theta)$ at the desired point using desired position and ${}^{0}_{n}R_D(\theta)$. See equations 3.37, 3.38, 3.42 and 3.25,

7. Obtain desired position, ${}^{B}_{6}P_D(\theta) = [D_x D_y D_z]^T$,
8. While $|{}^{B}_{6}\alpha_d(\theta) - {}^{B}_{6}\alpha_c(\theta)| > e$ and $|{}^{B}_{6}P_D(\theta) - {}^{B}_{6}P_C(\theta)| > e$ do
   (a) $Dp(\theta) = {}^{B}_{6}P_D(\theta) - {}^{B}_{6}P_C(\theta)$
   (b) $Do(\theta) = |{}^{B}_{6}\alpha_d(\theta) - {}^{B}_{6}\alpha_c(\theta)|$
   (c) Find model solution error, E_m
   (d) Apply joint angle change, $\theta = \theta + \Delta\theta$
   (e) If θ exceeds joint range, apply joint limit
   (f) Update position vector of ${}^{B}_{6}\vec{P}_c(\theta)$ using new θ
   (g) Update orientation vector of ${}^{B}_{6}\vec{\alpha}_c(\theta)$ using new θ
   (h) Compute velocity Jacobian, J_V, See equation 3.8
   (i) Compute angular velocity and orientation Jacobian, J_α. See equations 3.9, 3.12 and 3.33
   (j) Populate analytical Jacobian matrix, J_A with the velocity and orientation Jacobian. See equation 3.33
   (k) Invert analytical Jacobian vector and find the Error Damped (ED) pseudo inverse. See equation 3.45
   (l) Compute change in joint angles, $\Delta\theta = J^{\dagger ED} \cdot [Dp(\theta), Do(\theta)]^T$
9. Repeat until solution is within desired error, e or iterative time limit is reached.

3.3 Verification

Using the equations and algorithms discussed, the forward and inverse kinematics are verified by inputting arbitrary joint variables into the forward kinematics program and arbitrary desired robot poses into the inverse kinematics program while the output results are observed.

3.3.1 Forward Kinematics

To verify the results of the forward kinematics, a few sets of joint configurations were chosen and the robot pose was visually verified. For example, suppose a joint configuration where all joints were initialized to zero, it is expected that the robot should have its initial pose as seen in figure 3.1. This is confirmed by the MATLAB output graph in figure 3.4. Also, choosing a vertebrae angle, $\theta = 150^0$ produces figure 3.5 and if the vertebrae angle θ is set to 150^0 and the prismatic joint is increased from 0 to 30 mm figure 3.6 is produced.

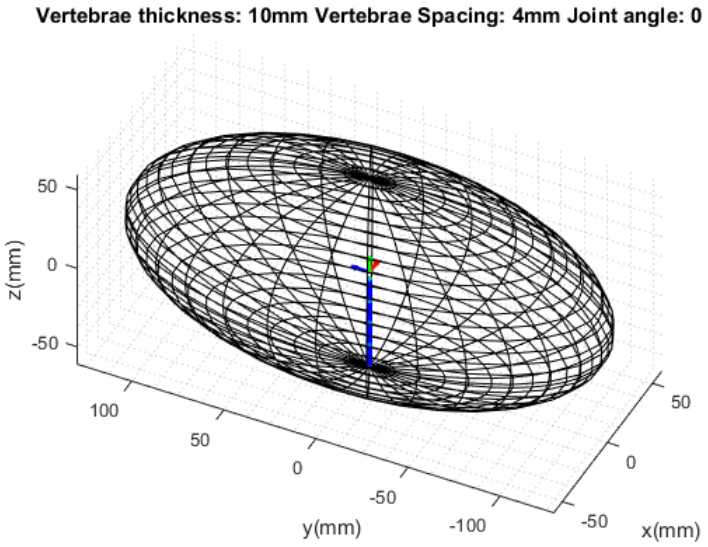


Figure 3.4: Joint Configurations Initialized to Zero

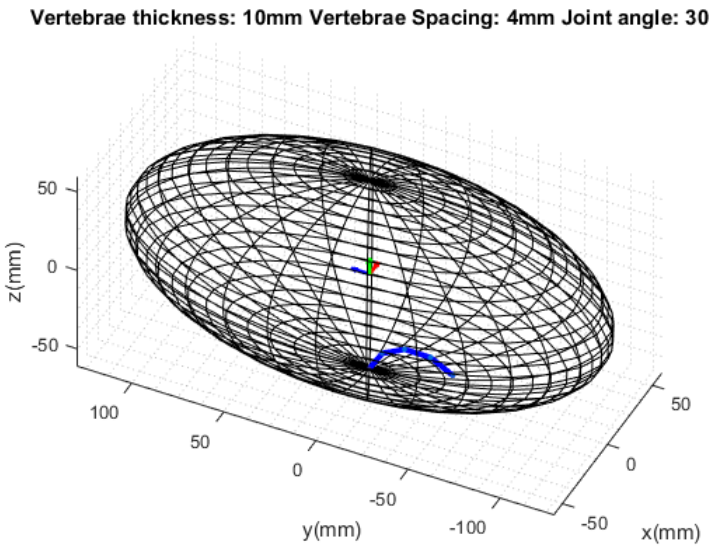


Figure 3.5: Robot with Vertebrae Joints at 150^0

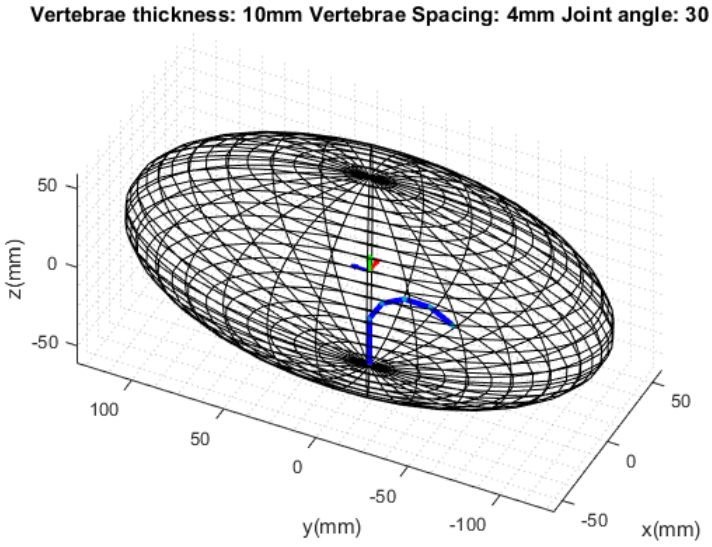


Figure 3.6: Robot with Vertebrae Joints at 150^0 and Prismatic Joint at $D = 30mm$

3.3.2 Inverse Kinematics

Likewise, the inverse kinematics analysis was verified by choosing and for inputting desired points on the bladder surface into MATLAB before running the inverse kinematics program. The iteration produces a solution for the robot end effector pose while displaying the desired orientation and the solution error. To verify the inverse kinematic algorithm, first, an arbitrary set of desired end effector joint parameters is defined as $[D, \theta_{r1}, \theta_p, \theta_y, \theta_{r2}, \theta] = [3, 15, 11, 10, 30, 100]^T$ and used to compute the robot forward kinematics. The output result was a visual plot of the robot pose (see figure 3.7) and the robot transformation matrix

$$ {}^{B}_{6}T = \begin{bmatrix} 0.7353 & 0.0090 & 0.6777 & 23.7255 \\ -0.6413 & 0.3328 & 0.6914 & -35.2230 \\ -0.2193 & -0.9430 & 0.2505 & -26.9702 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.49} $$

The rotation and position part of the transformation matrix, ${}^{B}_{6}T$, is extracted and input to the inverse kinematics program as the desired orientation matrix and desired position vector. The inverse kinematics program was executed and the output results are the joint parameters for reaching the desired pose $[D, \theta_{r1}, \theta_p, \theta_y, \theta_{r2}, \theta]^T = [1.1655, 1.3837, -1.6501, 10.9852, 37.3906, 68.9460]^T$, the solution error, $E_m = [0.0144\, mm, 2.6672^0]^T$, a visual plot of the robot pose (see figure 3.8) and the robot transformation matrix when a solution is achieved.

$$ {}^{B}_{6}T = \begin{bmatrix} 0.5995 & 0.2254 & 0.7680 & 23.7524 \\ -0.7716 & -0.0922 & 0.6294 & -35.1945 \\ 0.2127 & -0.9699 & 0.1186 & -27.0613 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.50} $$

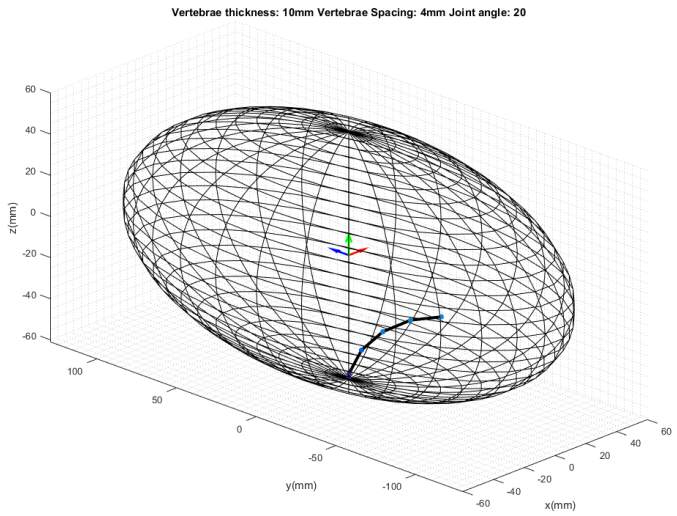


Figure 3.7: Forward Kinematics Computed with $[D, \theta_{r1}, \theta_p, \theta_y, \theta_{r2}, \theta] = [3, 15, 11, 10, 30, 100]^T$,

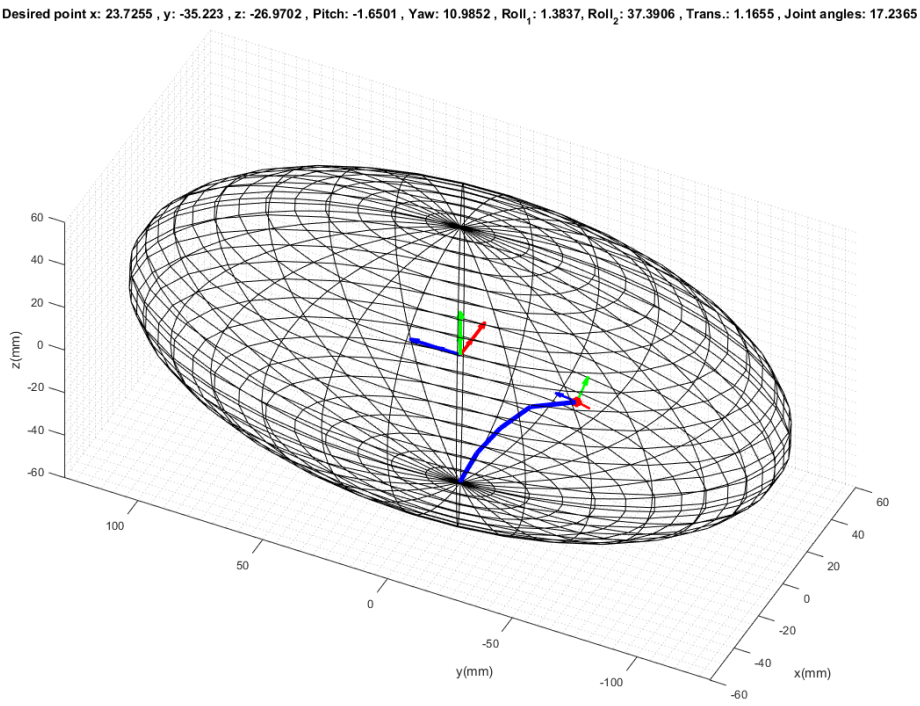


Figure 3.8: Inverse Kinematics Computed with Transformation Matrix from the Forward Kinematics Program

Comparing matrix equations 3.49 and 3.50 showed that there are differences in the elements of the orientation and position part. This is confirmed by the solution error, Em, computed by the inverse kinematics program. Even when the plot showed, visually, a similar robot pose, it is expected that the desired pose of the robot will differ from the computed robot pose from the inverse kinematics solution as the close-loop inverse differential kinematic method used in this research is an iterative method where approximate nth solutions are obtained from the (n-1)th approximation. Furthermore, more desired arbitrary end effector positions are chosen to compute the robot pose (as shown in figures 3.9 to 3.13) from the inverse kinematics. The solutions are obtained and checked to confirm if the goal positions

and orientations are reached, and at what solution error were the results obtained.

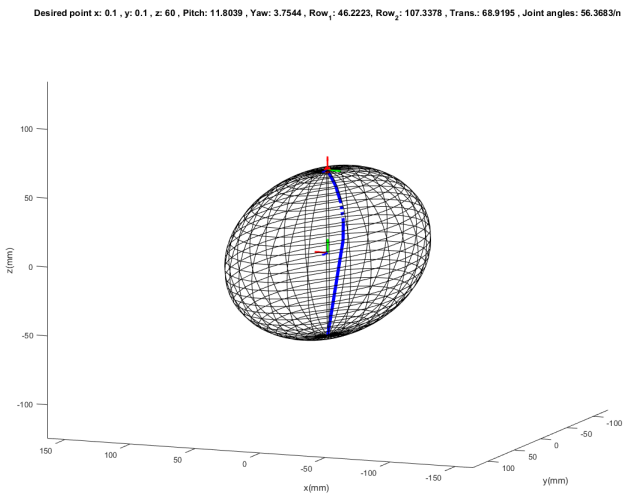


Figure 3.9: Robot Pose at Desired Position $(0.1, 0.1, 60)$ mm with Solution Error, $Em = [0.0146\,mm, 8.2970^0]^T$

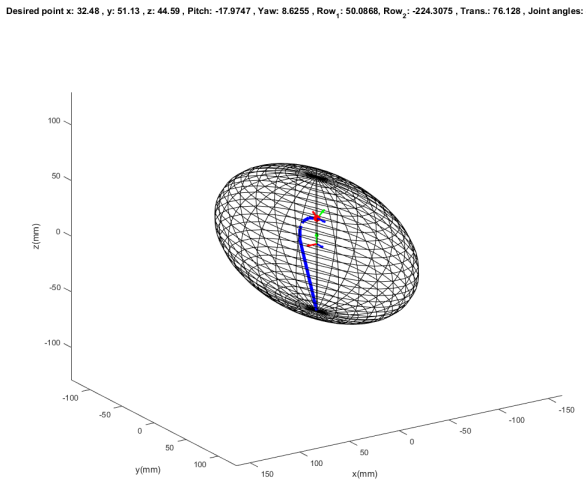


Figure 3.10: Robot Pose at Desired Position $(32.48, 51.13, 44.59)$ mm with Solution Error, $Em = [0.0102\,mm, 3.1627^0]^T$

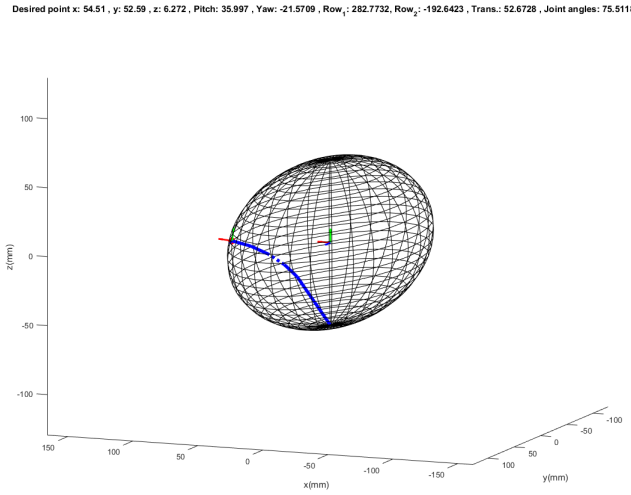


Figure 3.11: Robot Pose at Desired Position $(54.51, 52.59, 6.272)$ mm with Solution Error, $Em = [4.5842\,mm, 3.1149^0]^T$

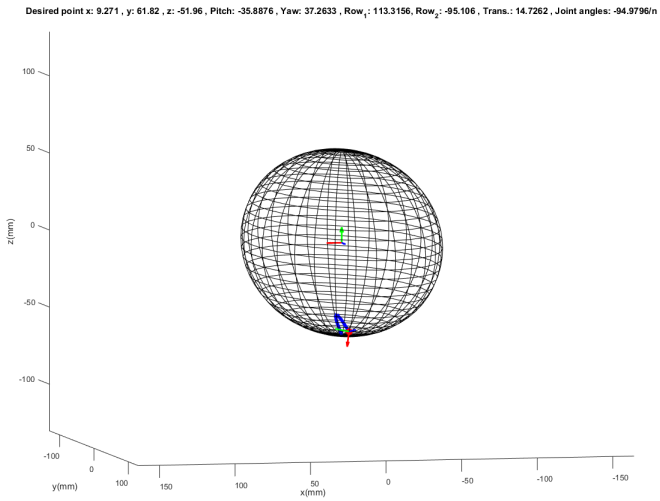


Figure 3.12: Robot Pose at Desired Position $(9.2710, 61.82, -51.96)$ mm with Solution Error, $Em = [0.2814\,mm, 9.9026^0]^T$

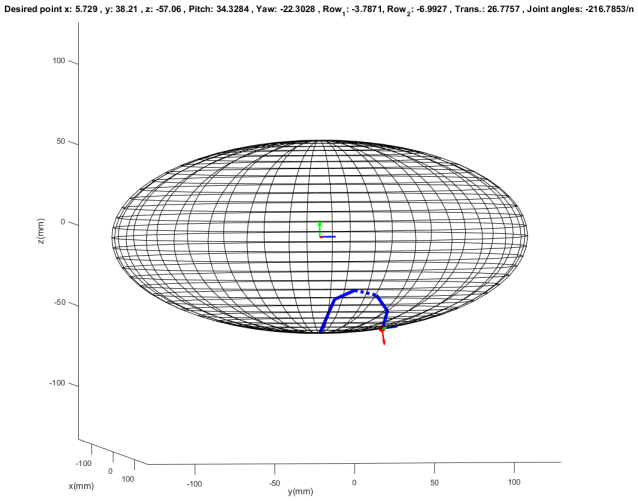


Figure 3.13: Robot Pose at Desired Position $(5.729, 38.21, -57.06)$ mm with Solution Error, $Em = [0.8424\,mm, 10.4616^0]^T$

Figures 3.9 to 3.13 results show that the inverse kinematics model could provide possible solutions for which desired poses around the inner bladder surface are reached.

3.4 Results

The results and plots from the forward kinematics confirm the frames were assigned properly and the MDH parameters adequately describe the proposed robot. Also, the proposed inverse kinematics proves to be a valid procedure for obtaining the desired pose of the robot. However, the achievements of these desired poses are not without error in attaining the goal position and orientation. This error depends on 'how far' the desired pose is from the initial pose. That is the magnitude of the misalignment of the initial and desired orientation, and the deviation of the initial and desired position. Therefore, when the desired pose was selected at the base of the bladder, there was a noticeable larger error compared to desired poses at the top half of the bladder model as seen in the solution errors of figures 3.9 to 3.13. Error optimization employed reduced the solution error of the kinematic model. This error could be minimized further with more capable optimization algorithms. It should also be added that without the use of an error damped Jacobian, for some desired pose, the solution pose obtained had a large deviation compared to the desired pose required (see figures 3.14 and 3.15). Even when the error damped Jacobian was implemented, the robot actuates outside the desired workspace (the bladder) before accessing the desired location with desired orientation when joint limits are not imposed on the joints as shown in figure 3.16 and 3.17.

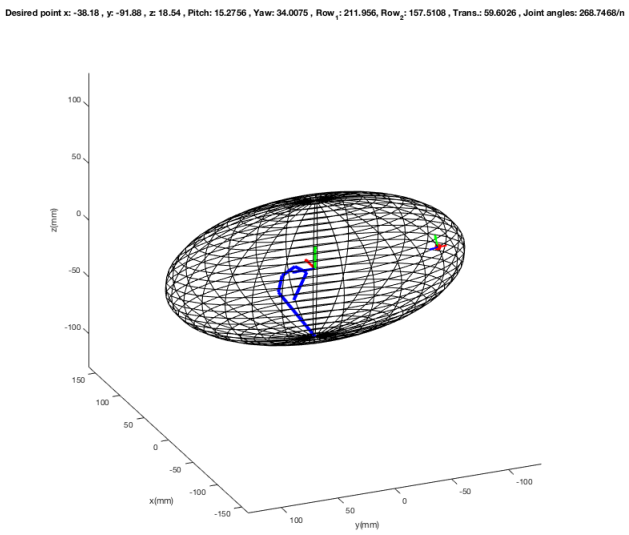


Figure 3.14: Robot Pose at Desired Position $(-38.18, -91.88, 18.54)$ mm without using an Error Damped Jacobian

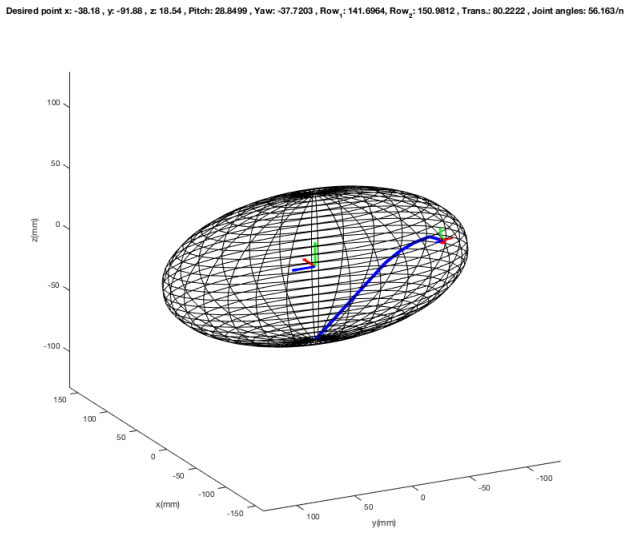


Figure 3.15: Robot Pose at Desired Position $(-38.18, -91.88, 18.54)$ mm using an Error Damped Jacobian

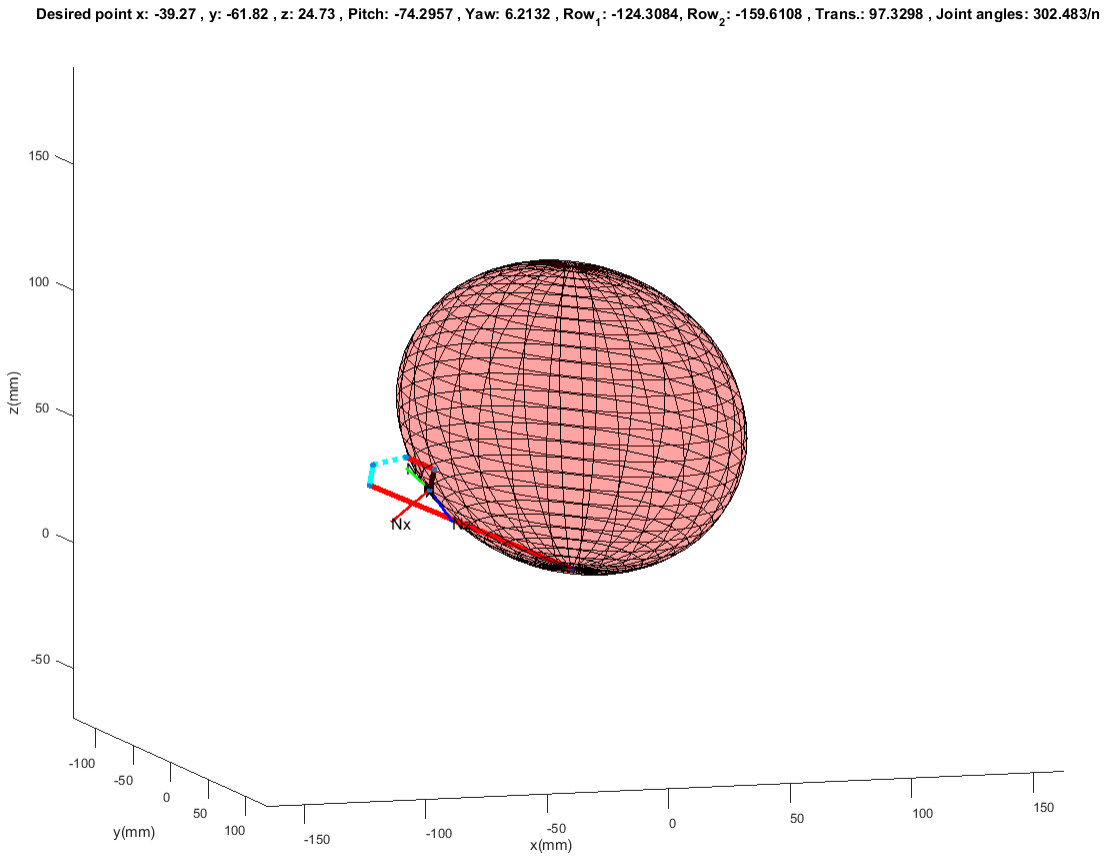


Figure 3.16: Robot Pose at Desired Position $(5.729, 38.21, -57.06)$ mm without Joint Limits

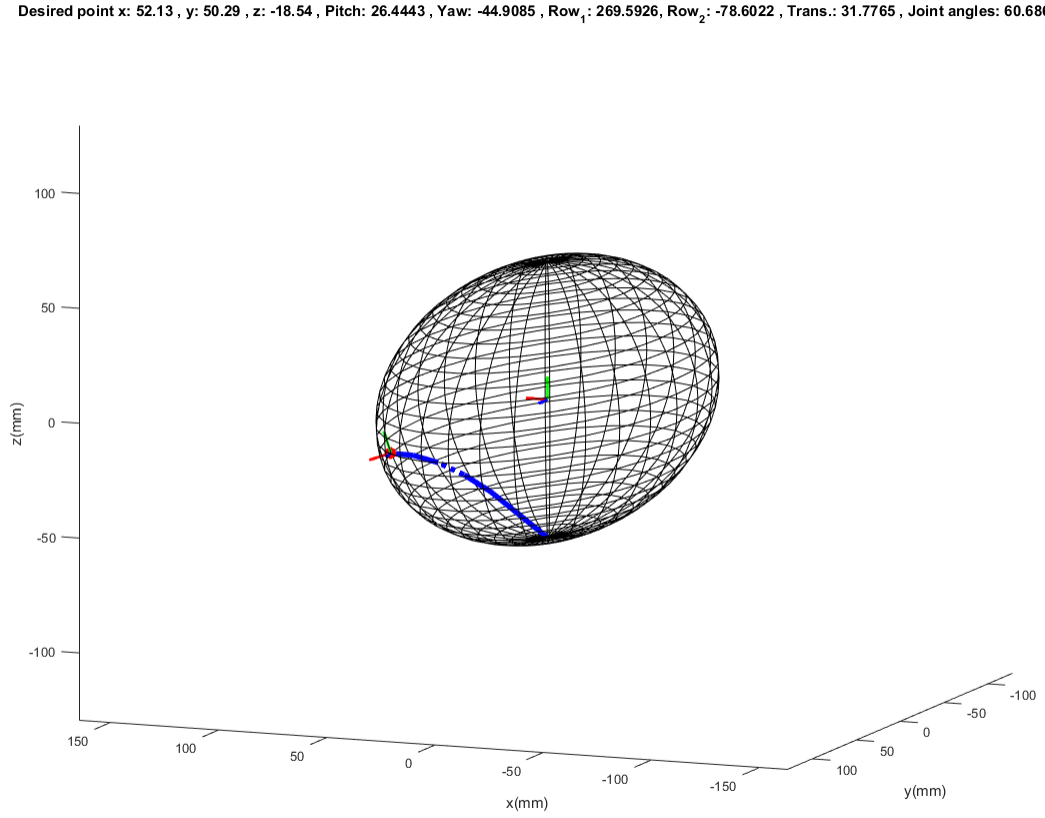


Figure 3.17: Robot Pose at Desired Position $(-39.27, -61.82, 24.73)$ mm with Joint Limits

3.5 Chapter Summary

Chapter 4 describes the kinematics of the proposed concept robot. Using the MDH parameter approach, the proposed robot is assigned joint attached frames and then, the MDH parameters are defined for each joint. Subsequently, the individual joint transformation matrices are obtained and the transformation that describes the entire robot, the robot transformation matrix is derived. This is the forward kinematic analysis for which a MATLAB program and the resulting outputs are provided.

Furthermore, a mathematical model is developed to find the inverse kinematics relationship of the joint parameters. The inverse kinematics developed using the analytical Jacobian in two

subparts; the first, a velocity Jacobian and second, the orientation Jacobian. The analytical Jacobian provides the relationship between the change in joint parameters and the desired position and orientation. This relationship is key to finding the joint configuration required for a desired position and orientation. The desired position and orientation is obtained from the surface of the modeled ellipsoid using a frame generated from the orthogonal basis of the normal to the ellipsoid. Finally, to minimize the computational error, the model solution error comprising of the position and orientation goal error is accounted for in the analytical Jacobian and used in a closed loop iteration program written in MATLAB and the output result provided. The results showed a convergence of the current end-effector pose to the desired pose with varied degree of minimized error especially for small initial difference between the robot end effector pose and the desired pose. The close-loop inverse differential kinematics approach with the an error damped Jacobian, produce cases were the achieved solutions have relatively large solution errors, E_m. These large solution errors could be better addressed using available solution optimization techniques.

CHAPTER 4

ROBOT MECHANICS

Using a quasi-static modeling approach, the forces acting on the robot to produce a change in joint configurations could be determined. Therefore, a static force analysis was implemented to obtain the tendon load force required to change the pose of the continuum end of the robot, the corresponding tip force, the shear stress and bearing area stress at the pin joint when a force is applied on the sensor tip of continuum end.

4.1 Tendon Loading and Tip Force Analysis

The robot is a manipulator segmented into three parts; the flexible-continuum part, the rigid link tube and the fixed link hyper-spherical joint at the base of the robot. The continuum part of the robot is assumed to be a cantilever beam undergoing a deflection due to a tip force, F_T, acting perpendicular to its edge. This tip force is a result of the reaction force of the tendon when a pull is exacted on the distal (or free) end of the continuum part.

The continuum end of the robot is tendon-driven and requires a tendon force to obtain a bending angle. Suppose the angle measured is that formed between frame $\{6\}$ at the end-effector and frame $\{Pr\}$ at the base of the continuum segment. Basically, pulling the actuation tendon creates a bend. Figure 4.1 shows the free-body diagrams of the continuum end of the robot with the acting forces when a tendon load, T, is applied and a bending angle, θ is achieved.

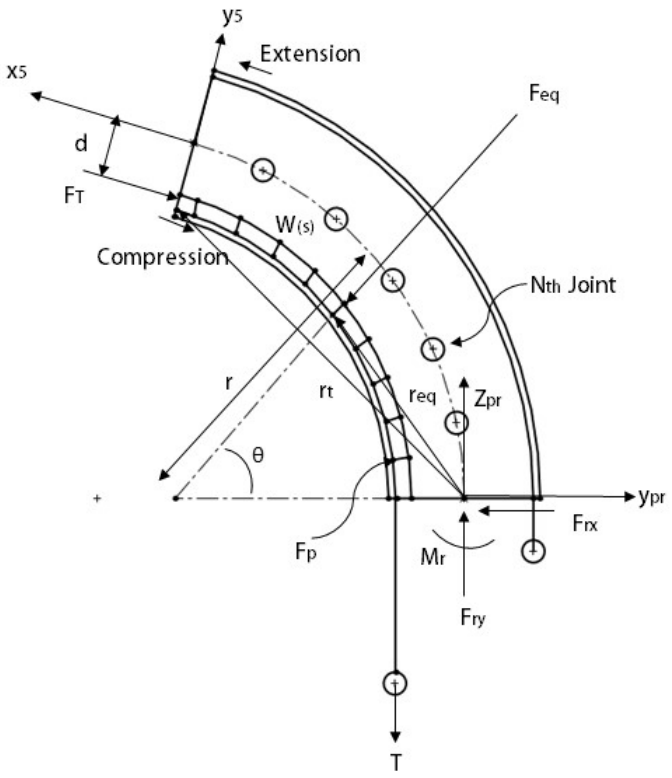


Figure 4.1: Free-body Diagram of the Continuum End as a Beam

According to Prasai et al, to analyse the relationships between the tendon force, T, the bend angle, θ, and the other forces, the following modeling assumptions were made; [75]

1. This robot segment is a continuum and there is a constant curvature deformation when a tendon force is applied,
2. The arc of the tube shows a similar characteristic shape as an arc of a circle,
3. All curves have the same center of radius,
4. There is negligible frictional and gravitational force,

The bend angle, θ is the angle from the frame $\{Pr\}$ to a point where the tip force, F_T, is acting. It is determined as

$$\theta = \frac{S}{r} \tag{4.1}$$

Where S and r are the arc length and radius of the center curve respectively,

The equation for the curvature of the center curve is

$$k = \frac{\partial \theta}{\partial s} \tag{4.2}$$

Using assumption 4, the curvature of any curve running parallel to the center curve is determined by

$$k = \frac{1}{r + x} \tag{4.3}$$

Therefore, the curvature, K_c, of the curve experiencing compression (or the inner arc) is

$$K_c = \frac{1}{r - d} \tag{4.4}$$

The point force, F_p, acting along the compressed arc of the tube is

$$\partial F_p = T \partial \theta \tag{4.5}$$

Therefore, the transverse force, $w(s)$, acting on the inner arc of the bent tube due to the tendon is given as

$$w(s) = \frac{\partial F_p}{\partial s} = T K_c \tag{4.6}$$

$w(s)$ expressed in frame $\{Pr\}$ is

$$w(s) = {}^{Pr}_{6}R \begin{bmatrix} -Tk & 0 & 0 \end{bmatrix}^T \tag{4.7}$$

Where ${}^{Pr}_{6}R = \begin{bmatrix} {}^{Pr}_{6}X & {}^{Pr}_{6}Y & {}^{Pr}_{6}Z \end{bmatrix}$ is the rotation matrix of frame $\{6\}$ with respect to frame $\{Pr\}$. While the negative sign accounts for the direction.

The total force, F_{eq}, acting on the compressed arc of the tube by the tendon is

$$F_{eq} = \int_0^{\phi} w(s) \partial s \tag{4.8}$$

and the tip force, F_T, in vector form is

$$F_T = {}^{Pr}_{6}R \begin{bmatrix} -T & 0 & 0 \end{bmatrix}^T \tag{4.9}$$

At the tube's bent equilibrium state, all forces should sum up to zero.

$$\sum F = F_T + F_r + F_{eq} = 0 \tag{4.10}$$

Therefore, the reaction force is

$$F_r = \begin{bmatrix} 0 & T & 0 \end{bmatrix}^T \tag{4.11}$$

Also, the reaction moment, M_r, is found to be

$$M_r = -Td \tag{4.12}$$

Using the Euler-Bernoulli beam theory, a relationship between curvature and the moment can be estimated by

$$M = kEI \tag{4.13}$$

where E is Young's modulus and I is moment of inertia.

Using equations 4.2 , 4.12 and 4.13, a relationship between the tension force and bending angle is obtained as

$$T = -\frac{EI\theta}{d \times s} \tag{4.14}$$

Where d and s are the diameter of the pin and the length of the flexible-continuum respectively.

4.2 Stress Analysis on the Pin Joint

Assuming there is no resistance to rotation at the pin joint, no friction at the pin, the forces and stresses can be obtained by using the principles of equilibrium. Taking into consideration the size of the structure components and the effect of applied forces to the material of these components, it is important to study the effective characteristics of the combination of these components and their materials during design to avoid failures like bending or breaking. Since breaking could result from pulling on or pushing with the structure components of the robot on the bladder, an allowable shear stress on the pin due to a parallel force when the continuum end of the robot is actuated needs to be established.

4.2.1 Allowable Shear Stress on Pin due to Applied Force, F

When the sensor tip probes the tissue of the bladder with a force, F, a shear stress occurs on the pin. The shear stress on the pin is dependent on the pin diameter, the thickness on the pin joint mount and can be describe as [76, 77]

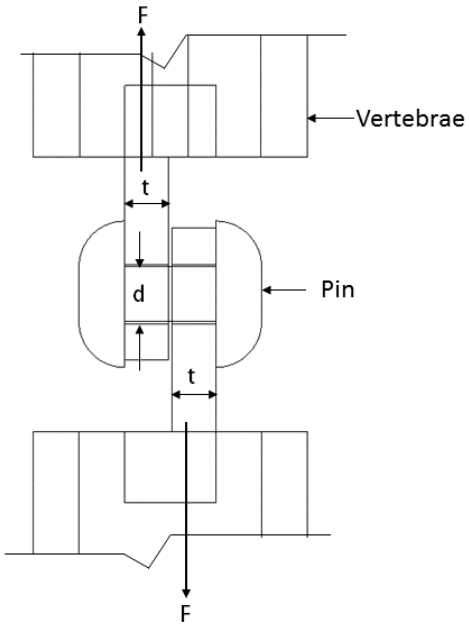


Figure 4.2: Shear Stress Model of the Pin Joint of the Vertebrae

Shear stress average, SS_A, is described as

$$SS_A = \frac{Applied force}{Area} \equiv \frac{4F}{\pi d^2} \tag{4.15}$$

Where d is diameter of the pin and F is applied force.

4.2.2 Bearing Area Stress on Pin

The contact force, F, between the sensor tip and the bladder also creates a contact pressure between the pin and the bearing area of the vertebrae as a result of compression force. Knowing the stress characteristics of the vertebrae will provide an insight to choosing what material it should be made from to avoid failure.[77, 78]

Therefore, the bearing area stress, B_t, is described as

$$B_t = \frac{F}{td} \tag{4.16}$$

Where t is thickness of the pin mount of each vertebrae.

4.2.3 Factor of Safety

Factor of safety, $F.S.$ is

$$F.S. = \frac{UltimateStress}{AllowableStress} \tag{4.17}$$

With a required factor of safety for medical device of ≥ 3, the ultimate stress before failure of the pin should be

$$UltimateStress = F.S. * AllowableStress \tag{4.18}$$

4.3 Results

The results to the tendon loading, tip force and stress analysis are obtained using equations 4.9 to 4.18 as shown in the following sections.

4.3.1 Tendon Loading and Tip Force Analysis

Assuming a planar motion of the continuum end of the robot and initializing the constant parameters, it is possible to determine the tendon and tip force at different bending angle from 0^0 to 90^0 as shown in figure 4.3. The figure suggest that a tendon load of about 3 N is required to achieve a maximum bending angle of 90^0 and a corresponding tip force of about 8 N.

The parameters initialized are

1. Young Modulus for silicone tube, $E_s = 1.707\ N/mm^2$.

2. Internal Radius of Tube, $r_2 = 1.5875\ mm$.
3. Outer Radius of Tube, $r_1 = 3.175\ mm$.
4. Continuum Robot Length, $s = 60\ mm$.
5. Distance of Tendon to Centroid, $l = 1\ mm$.

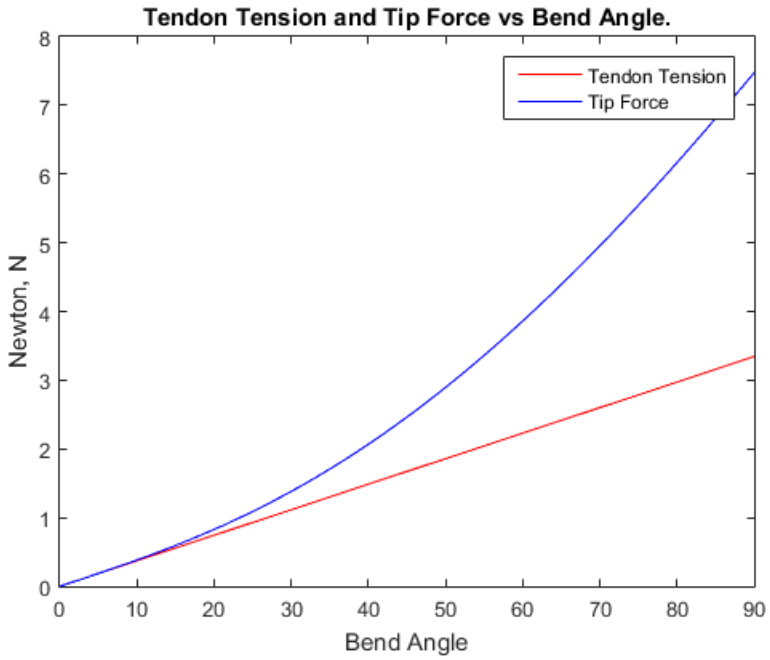


Figure 4.3: Tendon Tension and Tip Force verse Bend Angle

4.3.2 Stresses

The recommended tissue probing force is 0.1 N and a factor of safety, $F.S. = 3$ is choosen to determine the ultimate shear, bearing area and torsional stresses on the pin joint. The diameter, d, distance from the center of the pin joint to the tendon, l, and thickness, t, of the pin joint mount are initialized to 6×10^{-3} mm, 10×10^{-3} mm and 5×10^{-3} mm respectively. Clinically approved materials for surgery, along with their mechanical properties are presented in table 4.2. It can be deduce from table 4.2 that the clinically approved materials listed could be used in the design of the structural components of the robot.

The recommended tissue probing force is 0.1 N and a factor of safety, $F.S. = 3$ is choosen to determine the ultimate shear, bearing area and torsional stresses on the pin joint.

Table 4.1: Table Showing the Stresses for Various FDA Clinically Approved Design Materials

	Stresses (Nmm^{-2})	Ultimate Stresses ($Stress * 3$) (Nmm^{-2})	Clinically Approved Design Material for Surgical Instruments				
			Stainless Steel	Titanium	Niobium	Tantalum	Copper
Max. Shear Stress	0.317	0.951	✓	✓	✓	✓	✓
Max. Bearing Area Stress	0.313	0.939	✓	✓	✓	✓	✓
Appropriate design Material			✓	✓	✓	✓	✓

4.4 Chapter Summary

Chapter 4 details the mechanics of the flexible-continuum part of the concept robot using static analysis. The required tendon loading for a corresponding tip force was provided assuming constant-curvature of the continuum end. Also, the maximum shear stress and bearing area stress experience at individual joint when a load is exerted at the sensor tip is investigated assuming a safety of factor greater or equal to 3 as recommended by the FDA. Subsequently, it is found that the ideal design material for the structure of the robot could either be Stainless steel, Titanium, Niobium, Tantalum or Copper.

CHAPTER 5

PROTOTYPING AND ROBOT WORKING PRINCIPLE

5.1 Prototyping

The flexible-continuum module is prototyped to validate the constant-curvature bending as a result of the single tendon actuation and silicone tube permitting uniform joint angle distribution at the vertebrae rotational joints. Building a physical prototype of the conceptualize robotic system flexible-continuum module required using rapid prototyping tools. Primarily, a 3D printer. The vertebrae of the flexible-continuum module and a section of rigid tube alone are 3D printed and assembled. Firstly, a MakerBot Replicator 2 3D printer is used to 3D print the dummy sensor, the vertebrae and a section of the rigid at a scale of $1 : 3.47$ in PLA material. Two vertebrae were attached at the rotary joints by a 1.86 mm diameter screw, and also to the dummy sensor. Afterwards, a nylon wire is routed from the dummy sensor to the rigid tube through the vertebrae. Subsequently, the assembly is inserted into a silicone tube with an inner and outer diameter of 10.51 mm and 12.67 mm respectively (see figures 5.1 to 5.4).

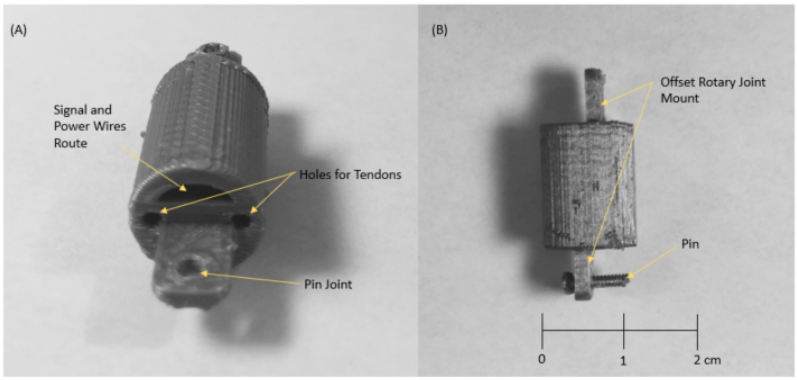


Figure 5.1: Image Showing Different Views of the 3D Printed Vertebra at Scale 1 : 3.47. (A) Front View, (B) Side View

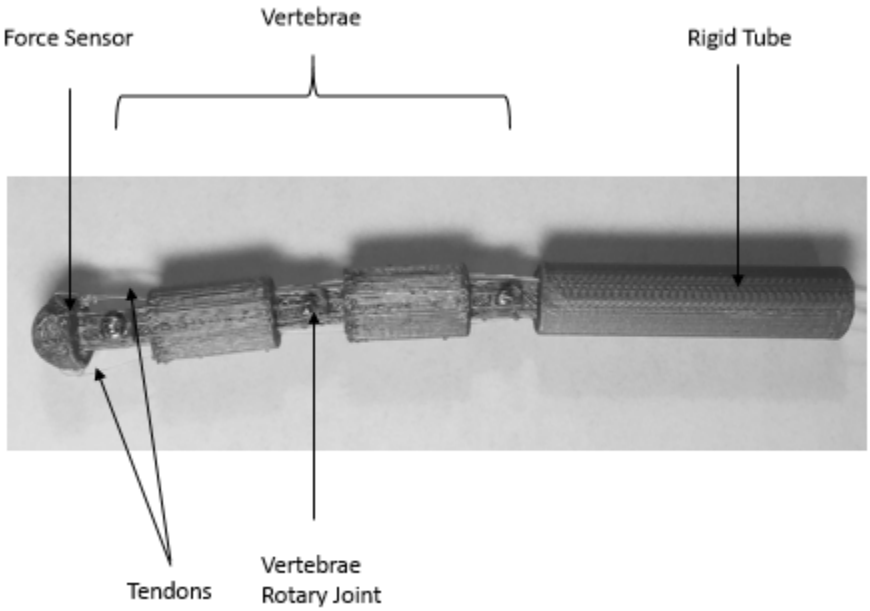


Figure 5.2: Image Showing 3D Printed and Assembled Flexible-Continuum Part without a Silicone Tube at Scale 1 : 3.47

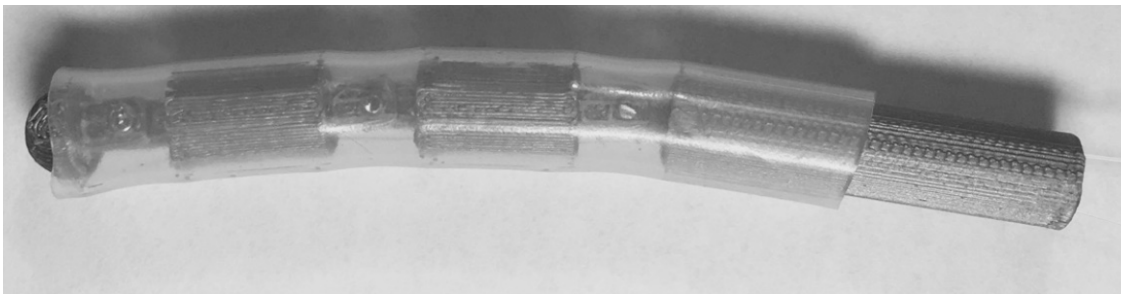


Figure 5.3: Image Showing 3D Printed and Assembled Flexible-Continuum Part with a Silicone Tube at Scale 1 : 3.47

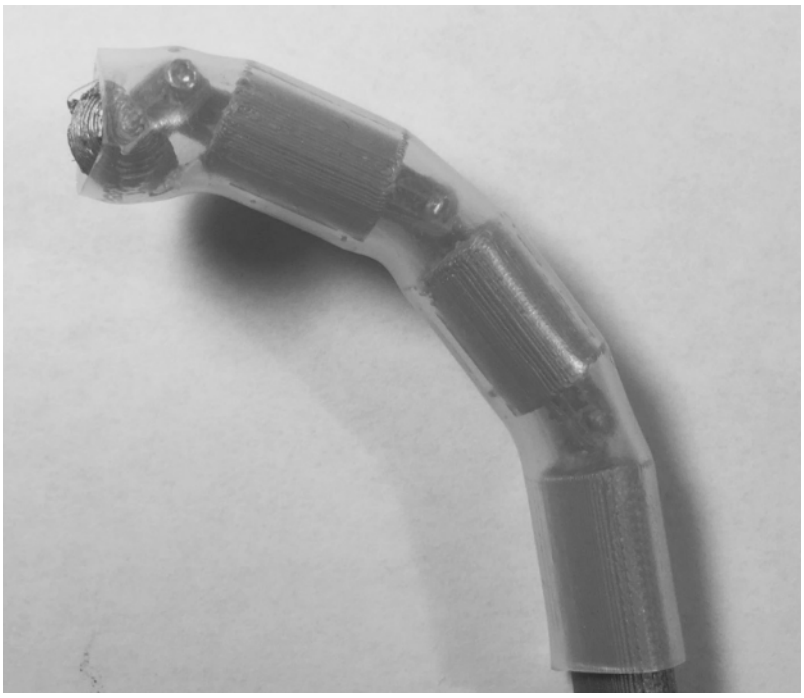


Figure 5.4: Image Showing 3D Printed and Assembled Flexible-Continuum Part with Actuation Load on One Tendon at Scale 1 : 3.47

Figure 5.4 shows the assembled flexible-continuum module with a single tendon actuation and distributed bending angle over the length of the flexible-continuum module confirming the design intent. Also, from the observing the 3D printed components individually and

when assembled, it was concluded that the offset rotary joint mount design was beneficial to the concentricity of the vertebrae in the silicone tube and hence, the proper routing of the tendons through the entire length of the flexible-continuum module. Proper routing of the tendons through the vertebrae allows for a uniformly distributed transverse force on the compression side of the bending flexible-continuum module as described by equation 4.6. Moving forward, the vertebrae and a section of the rigid tube were 3D printed (see figure 5.5) at a scale of $1:1$ using a polyjet printing process where additive layering of photopolymer material is continuously cured by UV light until the 3D design is developed. This printing method provided an extreme detail finish for a minimum supported and unsupported 3D design wall thickness of 0.3 mm and 0.6 mm respectively at a printing accuracy of $\pm 0.1 - 0.2$ mm for every 100 mm.

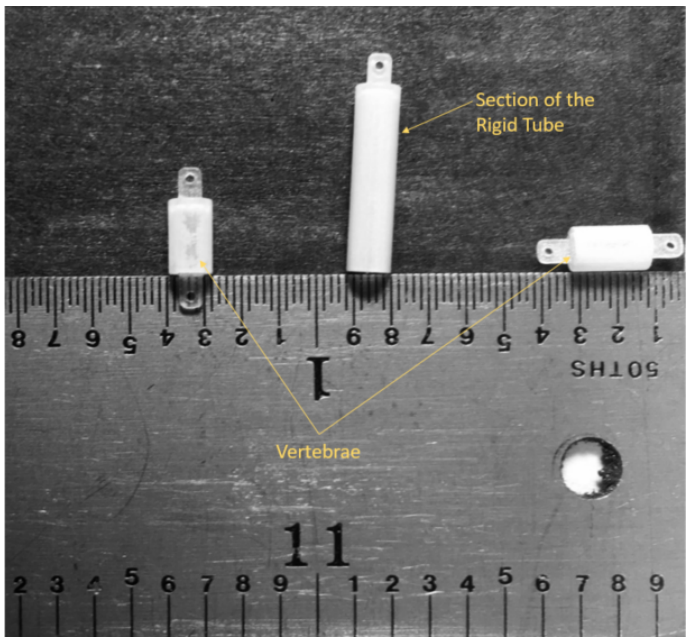


Figure 5.5: Image Showing 3D Printed Vertebrae and Rigid Tube Parts at Scale $1:1$

5.2 Robot Working Principle

The intended working principle of the conceptualized robot follows eight stages;

1. The patient is allowed to assume a dorsal lithotomy position on a surgical table.
2. The flexible-continuum module and rigid tube of the robot is manually inserted into the urethra such that the joint between the rigid tube and the flexible-continuum module is just inside the bladder.
3. An ultrasound scan of the bladder is made and a 3D reconstruction is obtained in real time.
4. A desired location is selected from the 3D reconstructed bladder model. A desired position and orientation frame is automatically generated from the desired location.
5. The end-effector of the robot autonomously moves to acquire the desired pose and applies a normal for of 0.1N.
6. The tissue deformation is detected and the bio-mechanical information of the tissue location is extracted.

5.3 Chapter Summary

Chapter 5 provides an overview of fabrication of various components of the concept robot using 3D printing rapid prototyping equipment. The flexible-continuum part was first 3D printed at a scale of $1 : 3.47$ for ease of assembling and validation of the tendon driven actuation. Actuating a single tendon produced an over bending angle distributed at the individual vertebrae rotary joints as expected. Furthermore, the components of the flexible-continuum part were then 3D printed at scale $1 : 1$ and visually inspected for design feature flaws that may arise at the mm scale. Lastly, an overall working principle of the robot during a transurethral procedure to measure the bladder tissue elasticity information was described.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.0.1 Conclusions

The purpose of this research was to investigate a minimally invasive flexible robotic device to evaluate the bladder wall tissue elasticity properties by palpation using a force sensor in order to diagnose bladder elasticity dysfunction. To accomplish this, the geometric information of the bladder including position and contact orientation of the interested location are of importance. It was also effective for this device to be inexpensive and small enough in size for a transurethral navigation and positioning. To accomplish these, a literature survey of existing solutions was performed to understand the design constraints, required actuation and modeling principles to control the robot. Thereafter, a design of the components and assembly of the robotic system was undertaken and discussed in chapter 2. The robotic system involved a flexible-continuum module coupled to a rigid tube and an actuation base. The design constraints lead to a concept robot with ten joints in which six joints are active and the other four are passive. The joints are also extrinsically actuated using tendons. Subsequently, a detailed analysis of the robot forward and inverse kinematics was performed to understand the motion capabilities and workspace of the robot as well as provide a means for controlling the robot pose. During this analysis, joint motions were constrained as discussed in section (2.2). Furthermore, an analysis into the loading force on the tendon required to actuate the flexible-continuum part from a bending angle range of 0^0 to $\pm 90^0$ was obtained. The ultimate bearing, torsional and shear stresses on the pins and pin joints were analyzed. It was found that the ideal structural design material could either be Stainless steel, Titanium, Niobium, Tantalum or Copper.

This design can overcome the challenges of transurethral navigation, position and stiffness limitations associated with urological surgery. Especially, bladder tissue dysfunction diagnosis through palpation. In conclusion, this research demonstrated that a combination of rigid links and a flexible silicone continuum module can be successfully used to provide a dexterious robot for estimating the bladder elasticity by extending the workspace, positioning and orientation of a contact force probe. Research surveyed demonstrated that the elasticity of the bladder wall varies with the volumetric pressure and region of interest. In response, there is a need for location specific elasticity information. The minimally invasive robotic system described here may advance us toward the prospect of making it technically feasible to obtain differences in the bladder elasticity at different regions of the bladder. Such system with the potential of obtaining biomechanical tissue information can also be used to assess pharmacological interventions in bladder conditions.

6.0.2 Future Work

Future development and integration of a force sensor at the end effector will be required for force sensing. The integration of a force sensor, its signal and power wires should also take into account the curvatures of the flexible-continuum part of the robot during bending. The flexible-continuum part and the rigid tube are designed to be mounted on an actuation base which controls five extrinsic joints. This research omitted the development of this base. However, it is an important feature of the robotic system and would be required in future work. Also, of importance is the structure material used in fabrication. In this research, the flexible-continuum part and the rigid tube is fabricated in nylon instead of the ideal material arrived at from the stress analysis. Future work will require using any of the clinical design material as fabrication material to experimentally confirm the analysis presented in this document. Also there is a need to develop a method to capture the robot motion during operation. A possible method may be visual servoing to capture the end-effector and joint

location as a 3D imaging technique is already used in tandem with the robot to obtain and reconstruct the geometry of the bladder. Furthermore, the method used to obtain the inverse kinematics solution of the robot given a desired pose is an iterative method which is prone to iterative error and therefore, an error in the desired pose obtained. The inverse kinematics optimization technique used here could be improved for solutions with less errors. In future, an experimentation on a physical prototype (using Titanium as the design material) will be required to confirm the analytical analysis.

APPENDIX A

A.1 Forward kinematics Matlab Program

Contents


```
clc, clear all;
```

FORWARD KINEMATICS PROGRAM.

```
AUTHOR: SAMSON ADEJOKUN.
1001104374.
```

COMPUTE FORWARD KINEMATICS.

DECLARE PARAMETERS. view([-64,40]) axis equal grid minor Symbolically declare all other parameters.

```
syms L1 L2 th R R1 R2 P TH T h B A B0 thp a al d thr Y Pr thy thr1 thr2 L3
% syms R P TH T B A B0 Y Pr d


th = 9.1412;
th = pi*th/180;
th = th/4;
T = 10;

h = 4;
L1 = T+(h/2);L2=T+h;
% L3 is length of sensor.
L3 = L1;
% Spherical joint at base parameters.
thp = -83.9945;
thp = pi*thp/180;
thy = -0.1109;
thy = pi*thy/180;
a = 30;
thr1 = 71.3459;
thr1 = pi*thr1/180;
thr2 = -34.6491;
thr2 = pi*thr2/180;
```

INPUT MDH PARAMETERS.

```
          FRAME         JOINT           MDH PARAMETER.
          C    N    #    Type    Var    a    alpha       d    theta
```

```
M_T = [      B    R1  1      R       thr1 0     0          -60     thr1;
             R1   P   2      R       thp  0    (pi/2)      0     thp;
             P    A   3      R        0   0     0          0     (pi/2);
             A    Y   4      R       thy  0    (pi/2)      0     ((pi/2)+thy);
             Y    R2  5      R       thr2 0    (pi/2)      0     thr2;
             R2   Pr  6      P       a    0     0          a     0;
             Pr   1   7      R       th   0    -(pi/2)     0     -(pi/2)+th;
             1    2   8      R       th   L2    0          0     th;
             2    3   9      R       th   L2    0          0     th;
             3    4   10     R       th   L2    0          0     th;
             4    5   11     R       0    L3    0          0     0];

% disp('MDH TABLE');
% disp('  C    N    #  Typ  Var a  alpha  d  theta');
% disp(M_T);

ti = size(M_T,1);
CTN = cell(ti,1);
N = 1;
```

COMPUTE HOMOGENEOUS TRANSFORMS.

```
for N = 1:size(M_T,1)

    % Find the Homogenenoeus Transforms of each frame.
    CTN{N}    = [  cos(sym(M_T(N,9)))                      -sin((sym(M_T(N,9))))
       0                       sym((M_T(N,6)));
                 (sin(sym(M_T(N,9))))*cos(M_T(N,7))    (cos((sym(M_T(N,9))))*cos(sym(M_T(N,7)))
))     -sin(sym(M_T(N,7)))  (-sin(sym(M_T(N,7))))*sym(M_T(N,8));
                 (sin(sym(M_T(N,9))))*sin(M_T(N,7))    (cos((sym(M_T(N,9))))*sin(sym(M_T(N,7)))
))     cos(sym(M_T(N,7)))   (cos(sym((M_T(N,7))))*sym(M_T(N,8)));
                  sym(0)                                      sym(0)
       sym(0)                  sym(1)];

    % Display the homogeneous Transforms of each frame
    X = ['HT from ', char(M_T(N,1)),' to ' ,char(M_T(N,2)),':'];
%     disp(X);
%     disp(double(CTN{N}));
%     disp(CTN{N});
    N =N + 1;
end
```

COMPUTE FORWARD KINEMATICS HOMOGENEOUS TRANSFORM w.r.t GLOBAL FRAME B.

Mutiply all HT of each frame to get transformation matrix of each frame to base frame.

```
   CT{1} = CTN{1};
   CT{2} = CTN{1}*CTN{2};
   CT{3} = CT{2}*CTN{3};
   CT{4} = CT{3}*CTN{4};
   CT{5} = CT{4}*CTN{5};
   CT{6} = CT{5}*CTN{6};
   CT{7} = CT{6}*CTN{7};
```

```
   CT{8} = CT{7}*CTN{8};
   CT{9} = CT{8}*CTN{9};
   CT{10} = CT{9}*CTN{10};
   CT{11} = CT{10}*CTN{11};


   for i = 1:numel(CT)
   X = ['HT from B0 to ' ,num2str(i),':'];
%    disp(X)
%    disp(double(CT{i}));
%    disp(CT{i});
   i = i+1;
   end
```

LINKS AND JOINT PLOTS.

```
   Rec = [0,0;0,T];
   xRec = Rec(1,:);
   yRec = Rec(2,:);
   NRec = [xRec;yRec;zeros(1,2)];

   for i = 1:numel(CT)
       Rot{i} = double(CT{i}(1:3,1:3));
       Trans{i} = double(CT{i}(1:3,4));
%        Rot{i} = CT{i}(1:3,1:3);
%        Trans{i} = CT{i}(1:3,4);


       % Calculate Joint Positions.
       px{i} = Trans{i}(1,1);
       py{i} = Trans{i}(2,1);
       pz{i} = Trans{i}(3,1);
%        hold on

       % Plot Joint Spheres.
%         [xs,ys,zs] = sphere(30);
%         mesh(xs+px{i},ys+py{i},zs+pz{i})
%         axis equal

        i =i+1;
   end

   for i = 1:numel(CT)-1
%        plot3([px{numel(CT)-1},px{numel(CT)}],[py{numel(CT)-1},py{numel(CT)}],[pz{numel(CT)-
1},pz{numel(CT)}],'k','LineWidth',3)
%        hold all
%        plot3([px{i},px{i+1}],[py{i},py{i+1}],[pz{i},pz{i+1}],'k','LineWidth',3)

       i = i+1;
   end
```

ELLIPSOID PLOT.

```
% ae = 60;
```

```
% be = 130;
% ce = 60;
% x0e = 0;
% y0e = 0;
% z0e = 0;
% center = [0 0 0];
% [xe,ye,ze] = ellipsoid(0,0,60,ae,be,ce,30);
% S = surf(xe,ye,ze,'FaceColor','red');
% rotate(S,[1 0 0],0,[-60,0,0])
% axis equal
% alpha(.2)
%
%    Y = ['Vertebrae thickness: ' , num2str(T),'mm Vertebrae Spacing: ' , num2str(h),'mm Join
t angle: ' , num2str(th*180/pi),];
%    title(Y)
%    xlabel('x(mm)')
%    ylabel('y(mm)')
%    zlabel('z(mm)')
```

Published with MATLAB® R2016a

A.2 Results from Forward kinematics Matlab Program

The resulting transformations of frames 1 through 11 using equation (3.3) are given as:

Transformation matrix of frame $\{R_1\}$ w.r.t $\{B\}$,

$$ {}^{B}_{R_1}T = \begin{bmatrix} \cos\theta_{R_1} & sin_{R_1} & 0 & 0 \\ \sin\theta_{R_1} & \cos_{R_1} & 0 & 0 \\ 0 & 0 & 1 & -60 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.1} $$

Transformation matrix of frame $\{P\}$ w.r.t $\{R_1\}$,

$$ {}^{R_1}_{P}T = \begin{bmatrix} \cos\theta_P & sin\theta_P & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\theta_P & \cos\theta & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.2} $$

Transformation matrix of frame $\{A\}$ w.r.t $\{P\}$,

$$ {}^{P}_{A}T = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.3} $$

Transformation matrix of frame $\{Y\}$ w.r.t $\{A\}$,

$$
{}^{A}_{Y}T = \begin{bmatrix} \cos(\theta_y + \frac{\pi}{2}) & sin(\theta_y + \frac{\pi}{2}) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_y + \frac{\pi}{2}) & \cos(\theta_y + \frac{\pi}{2}) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.4}
$$

Transformation matrix of frame $\{R_2\}$ w.r.t $\{Y\}$,

$$
{}^{Y}_{R_2}T = \begin{bmatrix} \cos(\theta_{R_2}) & sin(\theta_{R_2}) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin(\theta_{R_2}) & \cos(\theta_{R_2}) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.5}
$$

Transformation matrix of frame $\{Pr\}$ w.r.t $\{R_2\}$,

$$
{}^{R_2}_{Pr}T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & D \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.6}
$$

Transformation matrix of frame $\{1\}$ w.r.t $\{Pr\}$,

$$
{}^{Pr}_{1}T = \begin{bmatrix} \cos(\frac{\theta_y}{5} + \frac{\pi}{2}) & sin(\frac{\theta_y}{5} + \frac{\pi}{2}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ sin(\frac{\theta_y}{5} + \frac{\pi}{2}) & cos(\frac{\theta_y}{5} + \frac{\pi}{2}) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.7}
$$

Transformation matrix of frame $\{1\}$ w.r.t $\{2\}$,

$$
{}^{1}_{2}T = \begin{bmatrix} \cos(\frac{\theta}{5}) & sin(\frac{\theta}{5}) & 0 & 14 \\ \sin(\frac{\theta}{5}) & \cos(\frac{\theta}{5}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.8}
$$

Transformation matrix of frame {2} w.r.t {3},

$$
{}^{2}_{3}T = \begin{bmatrix} \cos(\frac{\theta}{5}) & sin(\frac{\theta}{5}) & 0 & 14 \\ \sin(\frac{\theta}{5}) & \cos(\frac{\theta}{5}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.9}
$$

Transformation matrix of frame {4} w.r.t {3},

$$
{}^{3}_{4}T = \begin{bmatrix} \cos(\frac{\theta}{5}) & sin(\frac{\theta}{5}) & 0 & 14 \\ \sin(\frac{\theta}{5}) & \cos(\frac{\theta}{5}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.10}
$$

Transformation matrix of frame {5} w.r.t {4},

$$
{}^{4}_{5}T = \begin{bmatrix} \cos(\frac{\theta}{5}) & sin(\frac{\theta}{5}) & 0 & 14 \\ \sin(\frac{\theta}{5}) & \cos(\frac{\theta}{5}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.11}
$$

Using Eqn. (3.4), the elements of the robot transformation matrix is computed as,

$$
\begin{aligned}
R_{11} = {} & 5\cos\left(\frac{\theta}{5}\right) sin(\theta_{R_1})\sin(\theta_Y) - 20\cos^3\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 16\cos^5\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) - 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& + \sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) + \sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& + 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) - 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& + \sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(t\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y)
\end{aligned}
$$

$$
\begin{aligned}
R_{12} = {} & 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) - \sin\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) \\
& - 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) + 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) + \sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& - 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_P)cos(\theta_{R_1})\sin(\theta_{R_2}) + 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) + 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& + 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& - 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& + 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P)
\end{aligned}
$$

$$
R_{13} = \cos(\theta_P)\cos(\theta_{R_1})\cos(\theta_{R_2}) - \cos(\theta_Y)\sin(\theta_{R_1})\sin(\theta_{R_2}) - \cos(\theta_{R_1})\sin(\theta_P)\sin(\theta_{R_2})\sin(\theta_Y)
$$

$$
\begin{aligned}
R_{21} = {} & 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) - 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) \\
& - 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) - \sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& - 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) + \sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) - 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) \\
& + \sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& - 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y)
\end{aligned}
$$

$$
\begin{aligned}
R_{22} = &\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) - 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
&+ 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) + \sin\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) \\
&+ 20\cos^3\left(\frac{\theta}{5}\right))\cos(\theta_{R_1}\cos(\theta_{R_2})\cos(\theta_Y) - 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
&- 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) + 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
&- 12\cos^5\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) \\
&+ 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
&- 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
&+ 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
&- 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) \\
&+ 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1})
\end{aligned}
$$

$$
R_{23} = \cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_{R_1}) + \cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_{R_2}) - \sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2})\sin(\theta_Y)
$$

$$
\begin{aligned}
R_{31} = &5\cos\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) + \sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) \\
&- 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) + 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) \\
&- 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) + 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) \\
&- \sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) + 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) \\
&- 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y)
\end{aligned}
$$

$$
\begin{aligned}
R_{32} = {} & 5\cos^3\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) - \sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) - 20\cos\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) \\
& + 16\cos^5\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) - 5\cos\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) \\
& + 20\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) - 16\cos^5\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) \\
& + 12\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) - 16\cos^4\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y)
\end{aligned}
$$

$$
R_{33} = \cos(\theta_{R_2})\sin(\theta_P) + \cos(\theta_P)\sin(\theta_{R_2})\sin(\theta_Y)
$$

$$
\begin{aligned}
P_x = {} & D\sin(\theta_{R_1})\sin(\theta_Y) - 28\cos^2\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) - 84\cos\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 56\cos^3\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) + 112\cos^4\left(\frac{\theta}{5}\right)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 28\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) - a\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& + 84\cos^2\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) - 56\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) \\
& - 112\cos^4\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_Y)\sin(\theta_P) - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\cos(\theta_Y)\sin(\theta_{R_1}) \\
& - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y) \\
& + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_Y)
\end{aligned}
$$

$$
\begin{aligned}
P_y = {} & 28\cos\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) - D\cos(\theta_{R_1})\sin(\theta_Y) + 84\cos^2\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) \\
& - 56\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) - 112\cos^4\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\sin(\theta_Y) \\
& + 28\cos\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) - a\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) \\
& + 84\cos^2\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) - 56\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) \\
& - 112\cos^4\left(\frac{\theta}{5}\right)\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R_1}) + 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& - 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_1})\cos(\theta_{R_2})\cos(\theta_Y) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\sin(\theta_{R_1})\sin(\theta_{R_2}) \\
& - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y) \\
& + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_{R_2})\sin(\theta_P)\sin(\theta_{R_1})\sin(\theta_Y)
\end{aligned}
$$

$$
\begin{aligned}
P_z = {} & D\cos(\theta_P)\cos(\theta_Y) - 28\cos\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) - 84\cos^2\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) \\
& + 56\cos^3\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) + 112\cos^4\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_Y) \\
& + 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) + 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) \\
& - 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\sin(\theta_P)\sin(\theta_{R_2}) + 28\cos\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) \\
& - 56\cos^2\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y) \\
& - 112\cos^3\left(\frac{\theta}{5}\right)\sin\left(\frac{\theta}{5}\right)\cos(\theta_P)\cos(\theta_{R_2})\sin(\theta_Y)
\end{aligned}
$$

A.3 Matlab Program for Inverse Kinematics given a Desired Position and Orientation

Contents


```
clear all
clc
```

INVERSE KINEMATICS PROGRAM.

```
AUTHOR: SAMSON ADEJOKUN.
1001104374.
```

```
% axis equal
% view([-64,40])
% hold on
% tic
```

Ellipsoid parameters.

```
syms xs ys zs
x0e = 0;
y0e = 0;
z0e = 0;
ae = 60;
be = 130;
ce = 60;
```

Robot Parameters.

```
T = 10;
h = 4;
L1 = T+(h/2);L2=T+h;
L3 = L1;
```

End-effector Desired Position.

```
Dx = 5.729;
Dy = 38.21;
Dz = -57.06;

D = [Dx;Dy;Dz];
```

End-effector Initial Position.

```
x = 0;
y = 0;
z = L3-18;
P = [x;y;z];
```

Initial Joint Variables.

```
a = 0;
thp = 0;
thy = 0;
thr1 = 0;
th = 0;
thr2 = 0;
```

Initial Change in Joint Variables.

```
Da = 0;
Dthp = 0;
Dthy = 0;
Dthr1 = 0;
Dth = 0;
Dthr2 = 0;
```

Plot Ellipsoid.

[xe,ye,ze] = ellipsoid(0,0,0,ae,be,ce,30); S = surf(xe,ye,ze,'FaceColor','none','FaceAlpha',0,'FaceLighting','none','EdgeColor','k');

Data Cursor Points.

```
dx = Dx;
dy = Dy;
dz = Dz;
```

Angular Position of End-Effector.

```
BT5 = [   cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(
thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - co
s(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi
```

```
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(
thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*s
in(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*s
in(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)
*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(th
r1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(
thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2
)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(th
r2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin
(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin
(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*c
os(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(
thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(si
n(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))
))), cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)
*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr
1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(
cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)
*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(th
r2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(th
p)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(
thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy
 + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(t
hr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))))
 - sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*s
in(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)
*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)
*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(th
y + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy +
 pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr
1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))))),
cos(thp)*cos(thr1)*cos(thr2) - sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/
2)*sin(thp));
   - cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*
sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy
 + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(
cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)
*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(th
r2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr
1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(t
hr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))
 - sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*si
n(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
 pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(co
s(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(
cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2
)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)
*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*
sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)
```

```
*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr
1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))),
sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(t
hr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi
/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(t
hr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos
(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))
) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*si
n(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin
(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*co
s(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*
sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))) - co
s(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr
1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(t
hr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))
- sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(
thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(t
hp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(
thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*si
n(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))), sin(t
hr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*si
n(thr1);
```

```
                                                                cos(th/4)*(cos(th/4)*
(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(
th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + co
s(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - sin(th/4)*(
cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(t
h/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos
(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))) - sin(th/4)*(
cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/
2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(
thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + s
in(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2
)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(t
hr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))),
```

```
                                                                - cos(th/4)*(cos(th/4)*(cos(
th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4
- pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp
)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + sin(th/4)*(cos(t
h/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 -
 pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)
```

```
*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))) - sin(th/4)*(cos(t
h/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
 sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2)
 + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - sin(th
/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) +
cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2)
+ cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))),
                                          cos(thr2)*sin(thp) - cos(thp)*cos(thy + pi/2)*sin(
thr2)];


Oc = rotm2eul(BT5)';
```

Normal Vector Frame and Desired Orientation.

```
N = 2*[(dx-x0e)/ae^2; (dy-y0e)/be^2; (dz-z0e)/ce^2];

% Find Normal Matrix.
Nx = N(:).'/norm(N);
Nyz=null(Nx).';
N_Mat=[Nx; Nyz];
N_Int=[ 0,  0, 1; 0, -1, 0;1,  0, 0];

N_Plot = N_Mat;
% quiver3(dx,dy,dz,N_Plot(1,1),N_Plot(1,2),N_Plot(1,3),'LineWidth',2, 'Color','r','MaxHeadSiz
e',2,'AutoScaleFactor',10);  % Plot Frame on Point.
% quiver3(dx,dy,dz,N_Plot(2,1),N_Plot(2,2),N_Plot(2,3),'LineWidth',2, 'Color','b','MaxHeadSiz
e',2,'AutoScaleFactor',10);
% quiver3(dx,dy,dz,N_Plot(3,1),N_Plot(3,2),N_Plot(3,3),'LineWidth',2, 'Color','g','MaxHeadSiz
e',2,'AutoScaleFactor',10);

%BT5des = inv(N_Int)*N_Mat;
BT5des = N_Mat;
```

Base Frame.

```
N_Base = [1 0 0;0 1 0;0 0 1];
% quiver3(0,0,0,N_Base(1,1),N_Base(1,2),N_Base(1,3),'LineWidth',2, 'Color','r','MaxHeadSize',
2,'AutoScaleFactor',10);  % Plot Frame on Point.
% quiver3(0,0,0,N_Base(2,1),N_Base(2,2),N_Base(2,3),'LineWidth',2, 'Color','b','MaxHeadSize',
2,'AutoScaleFactor',10);
% quiver3(0,0,0,N_Base(3,1),N_Base(3,2),N_Base(3,3),'LineWidth',2, 'Color','g','MaxHeadSize',
2,'AutoScaleFactor',10);


Od = rotm2eul(BT5des)';


[P1,P2,P3,P4,P6] = deal(1);
P5 = 0;
I = eye(3,3);
Im = eye(6,6);

PG = (norm(D-P))^2;
```

```
OG = ((dot(Od(1,1),Oc(1,1))-1)^2)+((dot(Od(2,1),Oc(2,1))-1)^2)+((dot(Od(3,1),Oc(3,1))-1)^2);
E = PG + OG;

while (((Od(1,1)-Oc(1,1))^2)+((Od(2,1)-Oc(2,1))^2)+((Od(3,1)-Oc(3,1))^2)>1e-2) && (((D(1,1)-x
)^2)+((D(2,1)-y)^2)+((D(3,1)-z)^2)>1e-2)

Dp = (D-P);
PG = (norm(D-P))^2;
OG = ((dot(Od(1,1),Oc(1,1))-1)^2)+((dot(Od(2,1),Oc(2,1))-1)^2)+((dot(Od(3,1),Oc(3,1))-1)^2);
Em = PG + OG;
   Do(1,1) = abs(Oc(1,1)-Od(1,1));
   Do(2,1) = abs(Oc(2,1)-Od(2,1));
   Do(3,1) = abs(Oc(3,1)-Od(3,1));


BT5 = [   cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(
thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - co
s(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(
thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*s
in(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*s
in(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)
*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(th
r1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(
thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2
)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(th
r2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin
(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin
(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*c
os(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(
thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(si
n(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))
))), cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)
*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr
1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(
cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)
*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(th
r2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(th
p)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(
thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy
 + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(t
hr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))))
 - sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*s
in(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)
*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)
*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(th
y + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy +
 pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr
1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))))],
cos(thp)*cos(thr1)*cos(thr2) - sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/
2)*sin(thp));
```

```
   - cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*
sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy
 + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(
cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)
*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(th
r2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr
1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(t
hr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))
 - sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*si
n(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
 pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(co
s(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(
cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2
)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)
*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*
sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)
*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr
1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))),
sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(t
hr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi
/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(t
hr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos
(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))
) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*si
n(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin
(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*co
s(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*
sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))) - co
s(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr
1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(t
hr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))
- sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(
thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(t
hp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(
thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*si
n(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))), sin(t
hr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*si
n(thr1);
```

```
                                                                cos(th/4)*(cos(th/4)*
(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(
th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + co
s(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - sin(th/4)*(
cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(t
h/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos
(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))) - sin(th/4)*(
cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/
```

```
2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(
thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + s
in(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2
)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(t
hr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))),
                                                                                    - cos(th/4)*(cos(th/4)*(cos(
th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4
 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp
)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + sin(th/4)*(cos(t
h/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 -
 pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)
*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))) - sin(th/4)*(cos(t
h/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
 sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2)
 + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - sin(th
/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) +
cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2)
+ cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))),
                                                cos(thr2)*sin(thp) - cos(thp)*cos(thy + pi/2)*sin(
thr2)];


Jv = [ 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy
 + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)
*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + 14*sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(
thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*si
n(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + 14*sin
(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - a*(cos(thr1)
*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + 14*cos(th/4)*(cos(th/4)*(sin(th/4 -
pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(co
s(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)
*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*si
n(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin
(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) + 14*sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2
)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(th
r2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin
(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(th
y + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp
)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) + 14*cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*si
n(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)) + L3*(cos
(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1
)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)
*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1
)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(th
r1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) +
 sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(t
hy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(th
p)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(t
hy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin
(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) + sin(t
```

```
h/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*
sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*s
in(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*
cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1
)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - s
in(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy
 + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)
*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy
 + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(t
hy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))))), 14*sin(
th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy
 + pi/2)) + cos(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)) - 14*cos(th/4)*(cos(th/4 - pi
/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) - sin(th/4
 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)) - 14*cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(
thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*c
os(thp)*cos(thr1)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2
) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(t
hy + pi/2))) + 14*sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(
thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/
2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2
)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2))) - L3*(cos(th/4)*(c
os(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(
thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)) - sin(th/4)*(si
n(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2))
 + cos(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi
/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) + cos(th/4
 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thp)
*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*cos(th
r1)*sin(thy + pi/2)))) - sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thp
)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*cos(t
hr1)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*
cos(thr1)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)))
 + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*c
os(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2)) - sin(th/4)*
(sin(th/4 - pi/2)*(cos(thr1)*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2
)) + cos(th/4 - pi/2)*cos(thp)*cos(thr1)*sin(thy + pi/2))))) - 14*cos(th/4 - pi/2)*(cos(thr1)
*sin(thp)*sin(thr2) + cos(thp)*cos(thr1)*cos(thr2)*cos(thy + pi/2)) + 14*sin(th/4 - pi/2)*cos
(thp)*cos(thr1)*sin(thy + pi/2) - a*cos(thp)*cos(thr1)*sin(thy + pi/2), a*(sin(thr1)*sin(thy
 + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - 14*sin(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2
) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - 14*cos(th/4)*(sin(th/4 - pi/2)*(sin(thr1)*sin(thy +
 pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*si
n(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) - 14*sin(th/4)*(cos(th/4 - pi/2)*(sin(thr1)*si
n(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thy + p
i/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) - L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(
sin(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(th/4
 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) + sin(th/
4)*(cos(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + sin(t
h/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))) + si
n(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*s
in(thp)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy
 + pi/2))) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/
2)*sin(thp)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin
(thy + pi/2))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/
2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(th
r1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thr1)*sin(thy +
 pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*si
n(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin
```

```
(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*cos(thr2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) + sin(th/4)*(cos(th/4 - pi/2)*
(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*cos(thr2)
*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))))) - 14*cos(th/4)*(cos(th/
4)*(sin(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(t
h/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))) + sin
(th/4)*(cos(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + s
in(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2))))
- 14*sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
pi/2)*sin(thp)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*
sin(thy + pi/2))) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(th
y + pi/2)*sin(thp)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(t
hp)*sin(thy + pi/2)))) + 14*cos(th/4 - pi/2)*cos(thr2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)
*sin(thp)*sin(thy + pi/2)), 14*sin(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*si
n(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)) + cos(th/
4 - pi/2)*sin(th/4)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp
)) - cos(thp)*cos(thr1)*cos(thr2))) - 14*cos(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(si
n(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2))
 - sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/
2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2))) - 14*cos(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin
(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)) - L3*(cos(
th/4)*(cos(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1
)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)) - sin(th/4)*sin(th/4 - pi/2)*(sin
(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*
cos(thr2))) - sin(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - c
os(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)) + cos(th/4 - pi/2)*sin(th/
4)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos
(thr1)*cos(thr2)))) - sin(th/4)*(cos(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*
sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)) + cos(t
h/4 - pi/2)*sin(th/4)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(t
hp)) - cos(thp)*cos(thr1)*cos(thr2))) + sin(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(sin
(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2))
 - sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2
)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2))))) - 14*cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(si
n(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2))
 + 14*sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
pi/2)*sin(thp)) - cos(thp)*cos(thr1)*cos(thr2)), - cos(thy + pi/2)*sin(thr1) - cos(thr1)*sin(
thp)*sin(thy + pi/2), (7*cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*s
in(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)
*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/2 + 14*cos(th/4)*((cos(th/4 - pi
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(
cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr
1)*sin(thr2)))/4) - (7*sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin
(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*c
os(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/2 - 14*sin(th/4)*((sin(th/4 - pi/2
)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(co
s(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)
*sin(thr2)))/4) + (7*cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy
 + pi/2)))/2 + (7*cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr
1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)
*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr
2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(
thr2)))))/2 - (7*sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1
)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(th
r1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*
(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2
```

```
)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(t
hr2)))))/2 + 14*cos(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1
)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(th
r1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(th/4 - pi
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(
cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr
1)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(t
hp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos
(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(cos(thr
2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(
thr2)))/4)) - 14*sin(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr
1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((sin(th/4 - p
i/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*
(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(th
r1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(
thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*co
s(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 - pi/2)*(c
os(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(th
r2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin
(thr2)))/4)) + L3*(cos(th/4)*((cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(th
r1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy +
pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin
(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - p
i/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*c
os(thr1)*sin(thr2)))))/4 - (sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1)
 + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/
2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th
/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2
)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(
thr1)*sin(thr2)))))/4 + cos(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) +
cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2)
- cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(t
h/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 -
 pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)
*cos(thr1)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr
1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - p
i/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*
(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(th
r1)*sin(thr2)))/4)) - sin(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + co
s(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) -
cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((sin(th/
4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - p
i/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*c
os(thr1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)
*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr
1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 - pi/
2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(c
os(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1
)*sin(thr2)))/4))) - sin(th/4)*((cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(
thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy
+ pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(c
os(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 -
 pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)
*cos(thr1)*sin(thr2)))))/4 + (sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr
```

```
1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + p
i/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(
th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi
/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*co
s(thr1)*sin(thr2)))))/4 + cos(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1)
 + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2
) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((sin
(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4
 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(th
p)*cos(thr1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(t
hr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos
(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 -
 pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2
)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(
thr1)*sin(thr2)))/4)) + sin(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) +
cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2)
- cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(t
h/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 - (sin(th/4 -
 pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)
*cos(thr1)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr
1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - p
i/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*
(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(th
r1)*sin(thr2)))/4))) + (cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*si
n(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(th
y + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*
(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4
 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(th
p)*cos(thr1)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1)
 + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/
2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th
/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2
)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(
thr1)*sin(thr2))))))/4 - (sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*
sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(
thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4
)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th
/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(
thp)*cos(thr1)*sin(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr
1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + p
i/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(
th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi
/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*co
s(thr1)*sin(thr2))))))/4) - (7*sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))/2;
 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/
2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(
thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(
thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy
+ pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*cos(th/4)
*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)
) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(th
p)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1)
 + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/
2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + 14*sin(th/4 - pi
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*sin(th/4)*(cos(th/4
```

)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th
/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(
thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr
1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(t
hr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - a*(cos(thy + pi/2)*sin(thr
1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi
/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(
thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*s
in(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(
thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + p
i/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(co
s(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + p
i/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*si
n(thp)) + cos(thp)*cos(thr1)*sin(thr2))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2
)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(th
r2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin
(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(th
y + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/
2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(
thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(si
n(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))
) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/
2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(
thp)) + cos(thp)*cos(thr1)*sin(thr2))))))) + 14*cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy
 + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)), 14*sin(th/4)*
(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1
)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)) - 14*cos(th/4)*(cos(th/4 - pi/2)*(s
in(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) - sin(th/4 - pi/2
)*cos(thp)*sin(thr1)*sin(thy + pi/2)) - L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)
*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) - sin(th/4 - p
i/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*si
n(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)
sin(thy + pi/2))) - sin(th/4)(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + c
os(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy +
pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(t
hy + pi/2)*sin(thr1)) - sin(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)))) - sin(th/4)*(c
os(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(
thy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)) + sin(th/4)*(co
s(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1))
- sin(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi
/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) - sin(th/4
- pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)
*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(th
r1)*sin(thy + pi/2))))) - 14*cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(t
hr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) - sin(th/4 - pi/2)*cos(thp)*sin(thr1)*si
n(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr
2)*cos(thy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2))) + 14*si
n(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(t
hy + pi/2)*sin(thr1)) + cos(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2)) + sin(th/4)*(cos
(th/4 - pi/2)*(sin(thp)*sin(thr1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) -
 sin(th/4 - pi/2)*cos(thp)*sin(thr1)*sin(thy + pi/2))) - 14*cos(th/4 - pi/2)*(sin(thp)*sin(th
r1)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)*sin(thr1)) + 14*sin(th/4 - pi/2)*cos(thp)*
sin(thr1)*sin(thy + pi/2) - a*cos(thp)*sin(thr1)*sin(thy + pi/2), 14*sin(th/4 - pi/2)*(cos(th
r1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - a*(cos(thr1)*sin(thy + pi/2) + co
s(thy + pi/2)*sin(thp)*sin(thr1)) + 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2)

```
 + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2
) - sin(thp)*sin(thr1)*sin(thy + pi/2))) + 14*sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thy
 + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy
 + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2))) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th
/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th/4 - pi/2
)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2))) + sin(th/4)*(co
s(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + sin(th/4 -
pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))) + sin(th/4
)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1
)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/
2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(
thr1)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy +
 pi/2))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + c
os(thy + pi/2)*sin(thp)*sin(thr1)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) -
sin(thp)*sin(thr1)*sin(thy + pi/2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2)
 + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2
) - sin(thp)*sin(thr1)*sin(thy + pi/2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)
*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thr1
)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(t
hr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + sin(th/4 - pi/2)*cos(thr2)*(cos(
thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))))) + 14*cos(th/4)*(cos(th/4)*(si
n(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th/4 -
pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2))) + sin(th/4)
*(cos(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + sin(th/
4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))) + 14*s
in(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*s
in(thr1)) + sin(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(th
y + pi/2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(th
p)*sin(thr1)) - cos(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*si
n(thy + pi/2)))) - 14*cos(th/4 - pi/2)*cos(thr2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(th
r1)*sin(thy + pi/2)), 14*cos(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy
 + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) - sin(th/4)*sin
(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + c
os(thp)*cos(thr2)*sin(thr1))) - 14*sin(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(cos(thr1
)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) + cos
(th/4 - pi/2)*sin(th/4)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(
thr1)) + cos(thp)*cos(thr2)*sin(thr1))) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*cos(th/4 - pi/2
)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(
thr2)*sin(thr1)) - sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy
 + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1))) - sin(th/4)*(cos(th/4)*sin(th/4
 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(th
p)*cos(thr2)*sin(thr1)) + cos(th/4 - pi/2)*sin(th/4)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) +
cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)))) - sin(th/4)*(cos(th/4)*
(cos(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*
sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) + cos(th/4 - pi/2)*sin(th/4)*(sin(thr2)*(cos(thr1)
*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1))) + sin
(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*si
n(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) - sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(co
s(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1))
))) + 14*cos(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*si
n(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) + 14*cos(th/4)*cos(th/4 - pi/2)*(sin(thr2)*(cos(thr1
)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) + cos(thp)*cos(thr2)*sin(thr1)) - 14*
sin(th/4)*sin(th/4 - pi/2)*(sin(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*s
in(thr1)) + cos(thp)*cos(thr2)*sin(thr1)),   cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*s
in(thy + pi/2), 14*sin(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(t
hp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos
(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((sin(th/4 -
```

```
 pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2
)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(
thr1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin
(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 - pi/2)*
(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(
thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*s
in(thr2)))/4)) - (7*cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/2 - 14*cos(th/4)*((cos(th/4 - pi/2)*(
cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(t
hr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*si
n(thr2)))/4) + (7*sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)
*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*
sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/2 + 14*sin(th/4)*((sin(th/4 - pi/2)*(co
s(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(cos(thr
2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(
thr2)))/4) - (7*cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi
/2)))/2 - (7*cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin
(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos
(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(c
os(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)
))))/2 + (7*sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(
thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + p
i/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(
thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(co
s(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))
)))/2 + L3*(cos(th/4)*(sin(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - s
in(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) +
 cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((sin(th
/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 -
pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*
sin(thr1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)
*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(th
y + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 - pi
/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(
cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr
1)*sin(thr2)))/4)) - (cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin
(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + c
os(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - p
i/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos
(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*
sin(thr2)))))/4 + (sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(th
p)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(
thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2
)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(th
r2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin
(thr2)))))/4 - cos(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*
sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy
 + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(th/4 - pi/
2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(c
os(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1
)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - pi/2)*(cos
(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(cos(thr2
```

)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4))) - (cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))))/4 + sin(th/4)*(sin(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4)) + (cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))/4 + (sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))/4 + cos(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + sin(th/4)*((cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4))) + (sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))))/4) - 14*cos(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 + cos(th/4)*((cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 - (sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp

```
)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(t
hy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))/4 - sin(th/4)*((sin(th/4 - p
i/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)))/4 + (cos(th/4 - pi/2)*
(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(th
r1)*sin(thr2)))/4)) + (7*sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + p
i/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))/2;
```

```
0,
```

```
14*cos(th/4 - pi/2
)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) + 14*cos(th/4)*(cos(th/4)*(cos(th
/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*sin(t
hp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy +
pi/2)*sin(thp)) - cos(th/4 - pi/2)*sin(thp)*sin(thy + pi/2))) - 14*sin(th/4)*(cos(th/4)*(sin(
th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*sin
(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy
+ pi/2)*sin(thp)) + sin(th/4 - pi/2)*sin(thp)*sin(thy + pi/2))) + 14*cos(th/4)*(cos(th/4 - pi
/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*sin(thp)*sin
```

```
(thy + pi/2)) - 14*sin(th/4)*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2
)*sin(thp)) - cos(th/4 - pi/2)*sin(thp)*sin(thy + pi/2)) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4
)*(cos(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi
/2)*sin(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*c
os(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*sin(thp)*sin(thy + pi/2))) - sin(th/4)*(cos(th/4)
*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/
2)*sin(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*co
s(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*sin(thp)*sin(thy + pi/2)))) - sin(th/4)*(cos(th/4)
*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) - cos
(th/4 - pi/2)*sin(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thp)*sin(thr2) - c
os(thr2)*cos(thy + pi/2)*sin(thp)) + sin(th/4 - pi/2)*sin(thp)*sin(thy + pi/2))) + sin(th/4)*
(cos(th/4)*(cos(th/4 - pi/2)*(cos(thp)*sin(thr2) - cos(thr2)*cos(thy + pi/2)*sin(thp)) + sin(
th/4 - pi/2)*sin(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thp)*sin(thr2) - co
s(thr2)*cos(thy + pi/2)*sin(thp)) - cos(th/4 - pi/2)*sin(thp)*sin(thy + pi/2))))) + 14*sin(th
/4 - pi/2)*sin(thp)*sin(thy + pi/2) - a*sin(thp)*sin(thy + pi/2),
```

```
    a*cos(thp)*cos(thy + pi/2) - 14*sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*cos(thp)*cos(thy +
 pi/2) - sin(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*c
os(thp)*cos(thy + pi/2) + cos(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2))) - 14*cos(th/4
)*(sin(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) + cos(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy +
pi/2)) - 14*sin(th/4)*(cos(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) - sin(th/4 - pi/2)*cos(thp)*
cos(thr2)*sin(thy + pi/2)) - L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*cos(thp)*c
os(thy + pi/2) + cos(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 -
 pi/2)*cos(thp)*cos(thy + pi/2) - sin(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2))) + sin
(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) - sin(th/4 - pi/2)*cos(thp)*cos(
thr2)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) + cos(th/4 - pi
/2)*cos(thp)*cos(thr2)*sin(thy + pi/2)))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)
*cos(thp)*cos(thy + pi/2) - sin(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2)) - sin(th/4)*
(sin(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) + cos(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi
/2))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) + cos(th/4 - pi/2)*co
s(thp)*cos(thr2)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) - si
n(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2))))) - 14*sin(th/4 - pi/2)*cos(thp)*cos(thy
+ pi/2) - 14*cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*cos(thp)*cos(thy + pi/2) + cos(th/4 - pi/
2)*cos(thp)*cos(thr2)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*cos(thp)*cos(thy + pi/2)
 - sin(th/4 - pi/2)*cos(thp)*cos(thr2)*sin(thy + pi/2))) - 14*cos(th/4 - pi/2)*cos(thp)*cos(t
hr2)*sin(thy + pi/2),
```

```
                          14*cos(th/4 - pi/2)*(cos(thr2)*sin(thp) - cos(thp)*cos(thy + pi/2
)*sin(thr2)) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(cos(thr2)*sin(thp) - cos
```

```
(thp)*cos(thy + pi/2)*sin(thr2)) - sin(th/4)*sin(th/4 - pi/2)*(cos(thr2)*sin(thp) - cos(thp)*
cos(thy + pi/2)*sin(thr2))) - sin(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(cos(thr2)*sin(thp) - cos
(thp)*cos(thy + pi/2)*sin(thr2)) + cos(th/4 - pi/2)*sin(th/4)*(cos(thr2)*sin(thp) - cos(thp)*
cos(thy + pi/2)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(cos(thr2)*si
n(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2)) + cos(th/4 - pi/2)*sin(th/4)*(cos(thr2)*sin(thp)
 - cos(thp)*cos(thy + pi/2)*sin(thr2))) + sin(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(cos(thr2)*si
n(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2)) - sin(th/4)*sin(th/4 - pi/2)*(cos(thr2)*sin(thp)
 - cos(thp)*cos(thy + pi/2)*sin(thr2))))) + 14*cos(th/4)*(cos(th/4)*cos(th/4 - pi/2)*(cos(thr
2)*sin(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2)) - sin(th/4)*sin(th/4 - pi/2)*(cos(thr2)*sin
(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2))) - 14*sin(th/4)*(cos(th/4)*sin(th/4 - pi/2)*(cos(
thr2)*sin(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2)) + cos(th/4 - pi/2)*sin(th/4)*(cos(thr2)*
sin(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2))) + 14*cos(th/4)*cos(th/4 - pi/2)*(cos(thr2)*si
n(thp) - cos(thp)*cos(thy + pi/2)*sin(thr2)) - 14*sin(th/4)*sin(th/4 - pi/2)*(cos(thr2)*sin(t
hp) - cos(thp)*cos(thy + pi/2)*sin(thr2)),                                          cos(thp)*s
in(thy + pi/2),
```

```
                                                    - (7*sin(th/4 - pi/2)*(sin(thp)*sin(thr2
) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/2 - (7*cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(t
hp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/
2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
 sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))))/2 - (7*sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(
sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy
 + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/
2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))))/2 - L3*(cos(th/4)*((cos(th/4)*(cos(th/4)*(
sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)
*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(t
hr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))))/4 + (sin(th/4)*(cos(th/4
)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi
/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*co
s(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))))/4 + cos(th/4)*((cos(t
h/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 -
 pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/4)*((sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(
thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4) + (sin(th
/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 -
pi/2)*cos(thp)*sin(thy + pi/2)))/4 + sin(th/4)*((cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(t
hp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4)) + sin(th/
4)*((cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/4)*((cos(th/4 - pi/2)*(sin(thp)*sin(th
r2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4)
 - (sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + c
os(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 - sin(th/4)*((sin(th/4 - pi/2)*(sin(thp)*sin(thr
2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4))
) + sin(th/4)*((cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr
2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2
)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(t
hy + pi/2))))/4 - (sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(
thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - p
i/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*si
n(thy + pi/2))))/4 + cos(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*c
os(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/4)*((cos(t
h/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*
cos(thp)*sin(thy + pi/2))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*co
s(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 - sin(th/4)*((sin(th
/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*c
os(thp)*sin(thy + pi/2))/4)) - sin(th/4)*((cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) +
cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/
4)*((sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/
4 - pi/2)*cos(thp)*sin(thy + pi/2))/4) + (sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + c
os(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + sin(th/4
)*((cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4
 - pi/2)*cos(thp)*sin(thy + pi/2))/4))) + (cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*
(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy
 + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi
/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(
sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy
 + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/
2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))))/4 + (sin(th/4)*(cos(th/4)*(cos(th/4)*(cos
(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*co
s(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2
```

```
)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - sin(th/4)*(cos(th/4)*(sin(
th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos
(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)
*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))))/4) - (7*cos(th/4)*(sin(th/4
 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp
)*sin(thy + pi/2)))/2 - 14*cos(th/4)*((sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(th
r2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4) - 14*cos(th/4)*((cos
(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4
 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/4)*((sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + co
s(thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4) + (sin(
th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4
 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + sin(th/4)*((cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos
(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4)) - (7*si
n(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/
4 - pi/2)*cos(thp)*sin(thy + pi/2)))/2 - 14*sin(th/4)*((cos(th/4 - pi/2)*(sin(thp)*sin(thr2)
 + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/4) - 1
4*sin(th/4)*((cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy +
pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 + cos(th/4)*((cos(th/4 - pi/2)*(sin(th
p)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 - (sin(th/4 - pi/2)*cos(thp)*sin(thy +
pi/2))/4) - (sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + p
i/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))/4 - sin(th/4)*((sin(th/4 - pi/2)*(sin(thp
)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)))/4 + (cos(th/4 - pi/2)*cos(thp)*sin(thy + p
i/2))/4)) - (7*cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))/2];

Z0 = [0;0;1];
Z1 = [0;0;1];
Z2 = [sin(thr1);-cos(thr1);0];
Z3 = [cos(thp)*cos(thr1);cos(thp)*sin(thr1);sin(thp)];
Z4 = [ - cos(thy + pi/2)*sin(thr1) - cos(thr1)*sin(thp)*sin(thy + pi/2); cos(thr1)*cos(thy +
pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2); cos(thp)*sin(thy + pi/2)];
Z5 = [ - cos(thy + pi/2)*sin(thr1) - cos(thr1)*sin(thp)*sin(thy + pi/2); cos(thr1)*cos(thy +
pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2); cos(thp)*sin(thy + pi/2)];


Jw = [P1*Z0 P2*Z1 P3*Z2 P4*Z3 P5*Z4 P6*Z5];

% Aalpha = [0 -sin(Oc(1,1)) cos(Oc(1,1))*sin(Oc(2,1));0 cos(Oc(1,1)) sin(Oc(1,1))*sin(Oc(2,1)
); 1 0 cos(Oc(2,1))];
Aalpha = [cos(Oc(1,1))*sin(Oc(2,1)) -sin(Oc(1,1)) 0; sin(Oc(1,1))*sin(Oc(2,1)) cos(Oc(1,1)) 0
; cos(Oc(2,1)) 0 1];
Aalphain = inv(Aalpha);

M(1:3,1:3) = I;
M(4:6,4:6) = Aalphain;
JB = [Jv;Jw];
JA = M*JB;
% JAc = H*(JA*H);

e = [Dp;Do];
  E = (1/2)*(transpose(e))*e;
%   LinkLengths = [0,0,0,0,a,L2];
% wi = (10^-3)*(LinkLengths.^2);
% wIm = diag(wi);
%   iJ = transpose(JAc)*inv(JAc*transpose(JAc) + E*Im);
    iJ = transpose(JA)*inv(JA*transpose(JA) + E*Im);

%   p = Im - iJ*Jv;
```

```
  Dq = iJ*[Dp;Do];

  Dthr1 = Dq(1,1);
  Dthp = Dq(2,1);
  Dthy = Dq(3,1);
  Dthr2 = Dq(4,1);
  Da = Dq(5,1);
  Dth = Dq(6,1);

   a = a + Da;
   thr1 = thr1 + Dthr1;
   thp = thp + Dthp;
   thy = thy + Dthy;
   thr2 = thr2 + Dthr2;
   th = th + Dth;

      if (thr1 > pi*360/180 || thr1 < -pi*360/180)
           thr1 = mod(thr1,pi*360/180);
      else
      end
      if (thp > pi*45/180 || thp < -pi*45/180)
           thp = mod(thp,pi*45/180);
      else
      end
      if (thy > pi*45/180 || thy < -pi*45/180)
              thy = mod(thy,pi*45/180);
      else
      end
      if (thr2 > pi*360/180 || thr2 < -pi*360/180)
            thr2 = mod(thr2,pi*360/180);
      else
      end
     if (a > 130 || a < -130)
       a = mod(a,130);
     else
     end
    if (th > pi*360/180 || th < -pi*360/180)
        th = mod(th,pi*360/180);
    else
    end

x = 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy +
pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*s
in(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*s
in(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(t
hy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*cos(th
/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi
/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin
(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(th
r1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy +
pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + 14*sin(th/4 -
 pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*sin(th/4)*(cos(t
h/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin
(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + c
os(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(
thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - co
s(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - a*(cos(thy + pi/2)*sin(
```

```
thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 -
 pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(c
os(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1
)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*s
in(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy
+ pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*
(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2
)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(t
hr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy
+ pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)
*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - p
i/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos
(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*
sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin
(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(c
os(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*
(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr
2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy +
pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*s
in(thp)) + cos(thp)*cos(thr1)*sin(thr2))))))) + 14*cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(
thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2));
y = a*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*sin(th/4)*(cos(th
/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2
)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(
thr1)*sin(thr2))) - 14*sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(t
hy + pi/2)) - 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*
sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*s
in(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - 14*cos(th/4)*(cos(th/4)*(sin(th/4 - pi/
2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(t
hr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*si
n(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(t
hy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(th
p)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(
cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)
*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(th
r2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy +
 pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*s
in(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(t
hy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)) - L3*(cos(th
/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*s
in(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*si
n(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*c
os(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)
*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) + si
n(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy
+ pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*
sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy
+ pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(th
y + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))) + sin(th/4
)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin
(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(
thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos
(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*s
in(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - sin(
th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy +
pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*si
```

```
n(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy +
pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy
+ pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))));
z = 14*cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + 14*cos(th
/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2)
+ cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) - 14*sin(
th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2))
+ cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2
) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) + L3*(c
os(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos
(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin
(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + p
i/2))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(
thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(
thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi
/2)))) - sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos
(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 -
pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*s
in(thy + pi/2))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(
thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - p
i/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*si
n(thy + pi/2))))) + 14*cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*c
os(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*sin(th/4)*(sin(th/4 - pi/2)
*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(th
y + pi/2)) - 14*sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2) + a*cos(thp)*sin(thy + pi/2) - 60;

Oc = rotm2eul(BT5)';
   P = [x;y;z];

%   disp([a;thr1*(180/pi);thp*(180/pi);thy*(180/pi);thr2*(180/pi);th*(180/pi)])

end

    b011 = 0;
    b012 = 0;
    b013 = -60;

    b021 = 0;
    b022 = 0;
    b023 = -60;

    b031 = 0;
    b032 = 0;
    b033 = -60;

    b041 = 0;
    b042 = 0;
    b043 = -60;

    b051 = 0;
    b052 = 0;
    b053 = -60;

    b061 = -a*(cos(pi/2 + thy)*sin(thr1) + cos(thr1)*sin(thp)*sin(pi/2 + thy));
    b062 = a*(cos(thr1)*cos(pi/2 + thy) - sin(thp)*sin(thr1)*sin(pi/2 + thy));
    b063 = a*cos(thp)*sin(pi/2 + thy) - 60;
```

```
    b071 = -a*(cos(pi/2 + thy)*sin(thr1) + cos(thr1)*sin(thp)*sin(pi/2 + thy));
    b072 = a*(cos(thr1)*cos(pi/2 + thy) - sin(thp)*sin(thr1)*sin(pi/2 + thy));
    b073 = a*cos(thp)*sin(pi/2 + thy) - 60;

    b081 = 14*sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2
)) - a*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*cos(th/5 - pi/2)
*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(t
hr1)*sin(thr2));
    b082 = a*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*sin(th/5 -
 pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*cos(th/5 - pi/2)
*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(t
hr1)*sin(thr2));
    b083 = 14*cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - 14
*sin(th/5 - pi/2)*cos(thp)*sin(thy + pi/2) + a*cos(thp)*sin(thy + pi/2) - 60;

    b091 = 14*cos(th/5)*(sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin
(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*sin(th/5)*(cos(th/5 - pi/2)*(cos(thy +
pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/5 - pi/2)*(cos(thr2)*(sin(thr1
)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14
*sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - a*(cos(t
hy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*cos(th/5 - pi/2)*(cos(thr2)*(
sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2
));
    b092 = a*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*sin(th/5)*
(cos(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/5
 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(th
p)*sin(thr1)*sin(thr2))) - 14*sin(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1
)*sin(thy + pi/2)) - 14*cos(th/5)*(sin(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin
(thr1)*sin(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - 14*cos(th/5 - pi/2)*(cos(thr2)*(
cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2
));
    b093 = 14*cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + 14
*cos(th/5)*(cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(
th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*sin(th/5)*(sin(th/5 - pi/2)*(sin(thp)*sin(thr2) +
 cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*sin(th
/5 - pi/2)*cos(thp)*sin(thy + pi/2) + a*cos(thp)*sin(thy + pi/2) - 60;

    b0101 = 14*cos(th/5)*(sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*si
n(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
 pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*sin(th/5)*(cos(th/5 - pi/2)*(cos(thy +
 pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/5 - pi/2)*(cos(thr2)*(sin(thr
1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 1
4*cos(th/5)*(cos(th/5)*(sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(
thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + p
i/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/5)*(cos(th/5 - pi/2)*(cos(thy + pi/2
)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*si
n(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + 14*si
n(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*sin(th/5
)*(cos(th/5)*(cos(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2
)) - sin(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(t
hp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/5)*(sin(th/5 - pi/2)*(cos(thy + pi/2)*sin(thr1
) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi
/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - a*(cos(thy + pi
/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*cos(th/5 - pi/2)*(cos(thr2)*(sin(thr
```

```
1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2));
    b0102 = a*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*sin(th/5)
*(cos(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/
5 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(t
hp)*sin(thr1)*sin(thr2))) - 14*sin(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - 14*cos(th/5)*(sin(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*si
n(thr1)*sin(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
 pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - 14*cos(th/5)*(cos(th/5)*(sin(th
/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/5 - pi/2
)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(
thr1)*sin(thr2))) + sin(th/5)*(cos(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/5 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*sin(th/5)*(cos(th/5)*(cos(th/5 -
 pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/5 - pi/2)*(c
os(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1
)*sin(thr2))) - sin(th/5)*(sin(th/5 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*s
in(thy + pi/2)) + cos(th/5 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*si
n(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*cos(th/5 - pi/2)*(cos(thr2)*(cos(thr
1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2));
    b0103 = 14*cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + 1
4*cos(th/5)*(cos(th/5)*(cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + p
i/2)) - sin(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/5)*(sin(th/5 - pi/2)*(sin(thp)*si
n(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/5 - pi/2)*cos(thp)*sin(thy + pi/2))) -
 14*sin(th/5)*(cos(th/5)*(sin(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy +
 pi/2)) + cos(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/5)*(cos(th/5 - pi/2)*(sin(thp)*
sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)))
 + 14*cos(th/5)*(cos(th/5 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) -
 sin(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*sin(th/5)*(sin(th/5 - pi/2)*(sin(thp)*sin(th
r2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/5 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*s
in(th/5 - pi/2)*cos(thp)*sin(thy + pi/2) + a*cos(thp)*sin(thy + pi/2) - 60;

    b0111 = 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*si
n(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
 pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 14*sin(th/4)*(cos(th/4 - pi/2)*(cos(thy +
 pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr
1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + 1
4*cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(
thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + p
i/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2
)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*si
n(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + 14*si
n(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + 14*sin(th/4
)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2
)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(t
hp)) + cos(thp)*cos(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1
) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi
/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - a*(cos(thy + pi
/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(si
n(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 -
 pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*
cos(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*si
n(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*
cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) + sin(th/4)*(cos(th/4)*(cos(th/4
- pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(
cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr
1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*
sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy
```

```
 + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))) + sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(
th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) - sin(th/4 - pi
/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*co
s(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(
thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*co
s(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2)))) - sin(th/4)*(cos(th/4)*(sin(th/4 -
pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(co
s(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)
*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thy + pi/2)*sin(thr1) + cos(thr1)*sin(thp)*si
n(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(sin(thr1)*sin(thy + pi/2) - cos(thr1)*cos(thy +
 pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2))))))) + 14*cos(th/4 - pi/2)*(cos(thr2)*(sin(th
r1)*sin(thy + pi/2) - cos(thr1)*cos(thy + pi/2)*sin(thp)) + cos(thp)*cos(thr1)*sin(thr2));
    b0112 = a*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - 14*sin(th/4)
*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/
4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(t
hp)*sin(thr1)*sin(thr2))) - 14*sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - 14*cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*si
n(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
 pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - 14*cos(th/4)*(cos(th/4)*(sin(th
/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2
)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(
thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr
1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2
)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*sin(th/4)*(cos(th/4)*(cos(th/4 -
 pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(c
os(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1
)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*s
in(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*si
n(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))) - 14*cos(th/4 - pi/2)*(cos(thr2)*(cos(thr
1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)) - L3
*(cos(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin
(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy +
pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos
(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(c
os(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)
))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*
sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*s
in(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*
cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1
)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))))) +
sin(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(t
hr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi
/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) - sin(th/4)*(sin(th/4 - pi/2)*(cos(t
hr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos
(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2)))
) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(cos(thr1)*cos(thy + pi/2) - sin(thp)*sin(thr1)*si
n(thy + pi/2)) + cos(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*sin(thy + pi/2) + cos(thy + pi/2)*sin
(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))) + sin(th/4)*(cos(th/4 - pi/2)*(cos(thr1)*co
s(thy + pi/2) - sin(thp)*sin(thr1)*sin(thy + pi/2)) - sin(th/4 - pi/2)*(cos(thr2)*(cos(thr1)*
sin(thy + pi/2) + cos(thy + pi/2)*sin(thp)*sin(thr1)) - cos(thp)*sin(thr1)*sin(thr2))))));
    b0113 = 14*cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + 1
4*cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + p
i/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi/2)*(sin(thp)*si
n(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2))) -
 14*sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy +
 pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/2)*(sin(thp)*
sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)))
```

```
 + L3*(cos(th/4)*(cos(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(t
hr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(th/4 - pi
/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin
(thy + pi/2))) - sin(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(th
r2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos(th/4 - pi/
2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(
thy + pi/2)))) - sin(th/4)*(cos(th/4)*(cos(th/4)*(sin(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(
thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) + sin(th/4)*(cos
(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*co
s(thp)*sin(thy + pi/2))) + sin(th/4)*(cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(t
hp)*cos(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - sin(th/4)*(sin(
th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos
(thp)*sin(thy + pi/2))))) + 14*cos(th/4)*(cos(th/4 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos
(thr2)*cos(thy + pi/2)) - sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2)) - 14*sin(th/4)*(sin(th/4
 - pi/2)*(sin(thp)*sin(thr2) + cos(thp)*cos(thr2)*cos(thy + pi/2)) + cos(th/4 - pi/2)*cos(thp
)*sin(thy + pi/2)) - 14*sin(th/4 - pi/2)*cos(thp)*sin(thy + pi/2) + a*cos(thp)*sin(thy + pi/2
) - 60;


%   plot3([b011 b021],...
%         [b012 b022],...
%         [b013 b023],'b','linewidth',4)
%
%
%     plot3([b021 b031],...
%         [b022 b032],...
%         [b023 b033],'b','linewidth',4)
%
%
%     plot3([b031 b041],...
%         [b032 b042],...
%         [b033 b043],'b','linewidth',4)
%
%
%     plot3([b041 b051],...
%         [b042 b052],...
%         [b043 b053],'b','linewidth',4)
%
%
%     plot3([b051 b061],...
%         [b052 b062],...
%         [b053 b063],'b','linewidth',4)
%
%
%     plot3([b061 b071],...
%         [b062 b072],...
%         [b063 b073],'k','linewidth',4)
%
%         plot3([b071 b081],...
%         [b072 b082],...
%         [b073 b083],'b','linewidth',4)
%
%         plot3([b081 b091],...
%         [b082 b092],...
%         [b083 b093],'b','linewidth',4)
%
%         plot3([b091 b0101],...
```

```
%         [b092 b0102],...
%         [b093 b0103],'b','linewidth',4)
%
%          plot3([b0101 b0111],...
%         [b0102 b0112],...
%         [b0103 b0113],'b','linewidth',4)
%
%     plot3(Dx,Dy,Dz,'r*','linewidth',6)
%
% % Frame on End Effector.
% EOc = eul2rotm(Oc(:).');
% quiver3(dx,dy,dz,EOc(1,1),EOc(1,2),EOc(1,3),'LineWidth',2, 'Color','r','MaxHeadSize',2,'Aut
oScaleFactor',10);
% quiver3(dx,dy,dz,EOc(2,1),EOc(2,2),EOc(2,3),'LineWidth',2, 'Color','b','MaxHeadSize',2,'Aut
oScaleFactor',10);
% quiver3(dx,dy,dz,EOc(3,1),EOc(3,2),EOc(3,3),'LineWidth',2, 'Color','g','MaxHeadSize',2,'Aut
oScaleFactor',10);
%
%
%
% % Plot Frame on Point.
% quiver3(0,0,0,1,0,0,'LineWidth',3, 'Color','r','MaxHeadSize',0.5,'AutoScaleFactor',20);
% quiver3(0,0,0,0,1,0,'LineWidth',3, 'Color','b','MaxHeadSize',0.5,'AutoScaleFactor',20);
% quiver3(0,0,0,0,0,1,'LineWidth',3, 'Color','g','MaxHeadSize',0.5,'AutoScaleFactor',20);
%
%
% title(['Desired point x: ',num2str(Dx),' , y: ',num2str(Dy),' , z: ',num2str(Dz),' , Pitch:
 ',num2str(thp*180/pi),' , Yaw: ',num2str(thy*180/pi),' , Row_1: ',num2str(thr1*180/pi),', Ro
w_2: ',num2str(thr2*180/pi),' , Trans.: ',num2str(a),' , Joint angles: ',num2str(th*180/pi),'
/n'])
% xlabel('x(mm)')
% ylabel('y(mm)')
% zlabel('z(mm)')
% toc
```

Published with MATLAB® R2016a

A.4 Results from Inverse kinematics Matlab Program

Using Eqns. 3.5 and 3.8, the elements of the robot's velocity jacobian, J_V are computed as,

$$\begin{aligned}
\frac{\delta P_x}{\delta \theta_{R1}} = {} & D\cos(\theta_{R1})\sin(\theta_Y) - 28\cos(\tfrac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y) - 84\cos^2(\tfrac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y) \\
& + 56\cos^3(\tfrac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y) + 112\cos^4(\tfrac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y) \\
& - 28\cos(\tfrac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}) + D\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}) \\
& - 84\cos^2(\tfrac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}) + 56\cos^3(\tfrac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}) \\
& + 112\cos^4(\tfrac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}) - 28\cos(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y) \\
& + 28\cos(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}) \\
& + 56\cos^2(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y) \\
& + 112\cos^3(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y) \\
& - 56\cos^2(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}) \\
& - 112\cos^3(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}) \\
& + 28\cos(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y) \\
& - 56\cos^2(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y) \\
& - 112\cos^3(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y)
\end{aligned}$$

$$\begin{aligned}
\frac{\delta P_x}{\delta \theta_P} = {} & -\cos(\theta_{R1})(D\cos(\theta_P)\cos(\theta_Y) - 28\cos(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) - 84\cos^2(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) \\
& + 56\cos^3(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) + 112\cos^4(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) \\
& + 56\cos^2(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) + 112\cos^3(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) \\
& - 28\cos(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) + 28\cos(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y) \\
& - 56\cos^2(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y) \\
& - 112\cos^3(\tfrac{\theta}{5})\sin(\tfrac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))
\end{aligned}$$

$$\frac{\delta P_x}{\delta \theta_Y} = D\cos(\theta_Y)\sin(\theta_{R1}) - 28\cos(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_{R1}) - 84\cos^2(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_{R1})$$
$$+ 56\cos^3(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_{R1}) + 112\cos^4(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_{R1})$$
$$- 28\cos(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_P)\sin(\theta_Y) + D\cos(\theta_{R1})\sin(\theta_P)\sin(\theta_Y)$$
$$- 84\cos^2(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_P)\sin(\theta_Y) + 56\cos^3(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_P)\sin(\theta_Y) +$$

$$\frac{\delta P_x}{\delta \theta_{R2}} = 112\cos^4(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_P)\sin(\theta_Y) + 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_{R1})\sin(\theta_Y)$$
$$- 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_{R1})\sin(\theta_Y)$$
$$- 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_{R1})\sin(\theta_Y)$$
$$- 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_P)$$
$$+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_P)$$
$$+ 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_P)$$

$$\frac{\delta P_x}{\delta a} = -28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})(\cos(\theta_Y)\sin(\theta_{R1})\sin(\theta_{R2}) - \cos(\theta_P)\cos(\theta_{R1})\cos(\theta_{R2})$$
$$+ \cos(\theta_{R1})\sin(\theta_P)\sin(\theta_{R2})\sin(\theta_Y))(2\cos(\frac{\theta}{5}) + 4\cos(\frac{\theta}{5})^2 - 1)$$

$$\frac{\delta P_x}{\delta \theta} = \sin(\theta_{R1})\sin(\theta_Y) - \cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P)$$

$$\frac{\delta P_y}{\delta \theta_{R1}} = \frac{(28\sin(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y))}{5} + \frac{(28\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}))}{5}$$
$$+ \frac{(28\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}))}{5} - \frac{(168\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y))}{5}$$
$$- \frac{(448\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y))}{5} - \frac{(112\cos(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}))}{5}$$
$$- \frac{(112\cos(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}))}{5} - \frac{(28\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P))}{5}$$
$$- \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}))}{5} + \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}))}{5}$$
$$+ \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}))}{5} - \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}))}{5}$$
$$+ \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}))}{5} + \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}))}{5}$$
$$+ \frac{(168\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y))}{5} + \frac{(28\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y))}{5}$$
$$- \frac{(112\cos(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y))}{5}$$
$$- \frac{(168\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P))}{5}$$
$$- \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y))}{5}$$
$$+ \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y))}{5}$$
$$+ \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y))}{5}$$
$$+ \frac{(168\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P))}{5}$$
$$+ \frac{(448\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P))}{5}$$

$$\begin{aligned}\frac{\delta P_y}{\delta \theta_P} &= D\sin(\theta_{R1})\sin(\theta_Y) - 28\cos(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y) - 84\cos^2(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y) \\ &+ 56\cos^3(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y) + 112\cos^4(\frac{\theta}{5})\sin(\theta_{R1})\sin(\theta_Y) \\ &+ 28\cos(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P) - D\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P) \\ &+ 84\cos^2(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P) - 56\cos^3(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P) \\ &- 112\cos^4(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_P) - 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}) \\ &- 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}) \\ &+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}) \\ &+ 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R1})\sin(\theta_{R2}) \\ &+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}) \\ &+ 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)\sin(\theta_{R1}) \\ &- 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y) \\ &+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y) \\ &+ 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y)\end{aligned}$$

$$\begin{aligned}\frac{\delta P_y}{\delta \theta_Y} = -\sin(\theta_{R1})(&D\cos(\theta_P)\cos(\theta_Y) - 28\cos(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) - 84\cos^2(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) \\ &+ 56\cos^3(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) + 112\cos^4(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y) \\ &+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) + 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) \\ &- 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}) + 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y) \\ &- 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y) \\ &- 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))\end{aligned}$$

$$\begin{aligned}\frac{\delta P_y}{\delta \theta_{R2}} = 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})&(\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_{R1}) + \cos(\theta_{R1})\cos(\theta_Y)\sin(\theta_{R2}) \\ &- \sin(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2})\sin(\theta_Y))(2\cos(\frac{\theta}{5}) + 4\cos^2(\frac{\theta}{5}) - 1)\end{aligned}$$

$$\frac{\delta P_y}{\delta a} = -\cos(\theta_{R1})\sin(\theta_Y) - \cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1})$$

$$\begin{aligned}
\frac{\delta P_y}{\delta \theta} = {} & \frac{(28\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}))}{5} - \frac{(28\sin(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y))}{5} \\
& - \frac{(28\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y))}{5} + \frac{(112\cos(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y))}{5} \\
& - \frac{(112\cos(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}))}{5} - \frac{(28\sin(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}))}{5} \\
& + \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y))}{5} - \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y))}{5} \\
& - \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_{R1})\cos(\theta_{R2})\cos(\theta_Y))}{5} - \frac{(168\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y))}{5} \\
& - \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}))}{5} + \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}))}{5} \\
& + \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R1})\sin(\theta_{R2}))}{5} + \frac{(28\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y))}{5} \\
& + \frac{(168\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y))}{5} + \frac{(448\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R1})\sin(\theta_Y))}{5} \\
& - \frac{(112\cos(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y))}{5} \\
& - \frac{(168\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}))}{5} \\
& - \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y))}{5} \\
& + \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y))}{5} \\
& + \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_{R1})\sin(\theta_Y))}{5} \\
& + \frac{(168\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}))}{5} \\
& + \frac{(448\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)\sin(\theta_{R1}))}{5}
\end{aligned}$$

$$\frac{\delta P_z}{\delta \theta_{R1}} = 0$$

$$\frac{\delta P_z}{\delta \theta_P} = 28\cos(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P) - D\cos(\theta_Y)\sin(\theta_P)$$
$$+ 84\cos^2(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P) - 56\cos^3(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P)$$
$$- 112\cos^4(\frac{\theta}{5})\cos(\theta_Y)\sin(\theta_P) - 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R2})$$
$$+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R2}) + 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\sin(\theta_{R2})$$
$$- 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y) + 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y)$$
$$+ 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\sin(\theta_P)\sin(\theta_Y)$$

$$\frac{\delta P_z}{\delta \theta_Y} = -\cos(\theta_P)(a\sin(\theta_Y) - 28\cos(\frac{\theta}{5})\sin(\theta_Y) - 84\cos^2(\frac{\theta}{5})\sin(\theta_Y) + 56\cos^3(\frac{\theta}{5})\sin(\theta_Y)$$
$$+ 112\cos^4(\frac{\theta}{5})\sin(\theta_Y) - 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y)$$
$$+ 56\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y) + 112\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_{R2})\cos(\theta_Y))$$

$$\frac{\delta P_z}{\delta \theta_{R2}} = 28\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})(\cos(\theta_{R2})\sin(\theta_P) + \cos(\theta_P)\sin(\theta_{R2})\sin(\theta_Y))(2\cos(\frac{\theta}{5}) + 4\cos(\frac{\theta}{5})^2 - 1)$$

$$\frac{\delta P_z}{\delta a} = \cos(\theta_P)\cos(\theta_Y)$$

$$\frac{\delta P_z}{\delta \theta} = \frac{(28\sin(\theta_P)\sin(\theta_{R2}))}{5} + \frac{(28\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y))}{5}$$
$$- \frac{(112\cos(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}))}{5} - \frac{(392\cos^2(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}))}{5}$$
$$+ \frac{(168\cos^3(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}))/5 + (448\cos^4(\frac{\theta}{5})\sin(\theta_P)\sin(\theta_{R2}))}{5}$$
$$- \frac{(28\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))}{5} + \frac{(112\cos(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))}{5}$$
$$+ \frac{(168\cos(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y))}{5} + \frac{(392\cos^2(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))}{5}$$
$$- \frac{(168\cos^3(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))}{5} - \frac{(448\cos^4(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_{R2})\sin(\theta_Y))}{5}$$
$$- \frac{(168\cos^2(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y))}{5} - \frac{(448\cos^3(\frac{\theta}{5})\sin(\frac{\theta}{5})\cos(\theta_P)\cos(\theta_Y))}{5}$$

APPENDIX B

B.1 Matlab Program to Compute Tendon Loading and Tip Force

Contents


```
clc;
clear all;
```

TENDON LOADING AND TIP FORCE PROGRAM.

AUTHOR: SAMSON ADEJOKUN. 1001104374

Initialize Parameters.

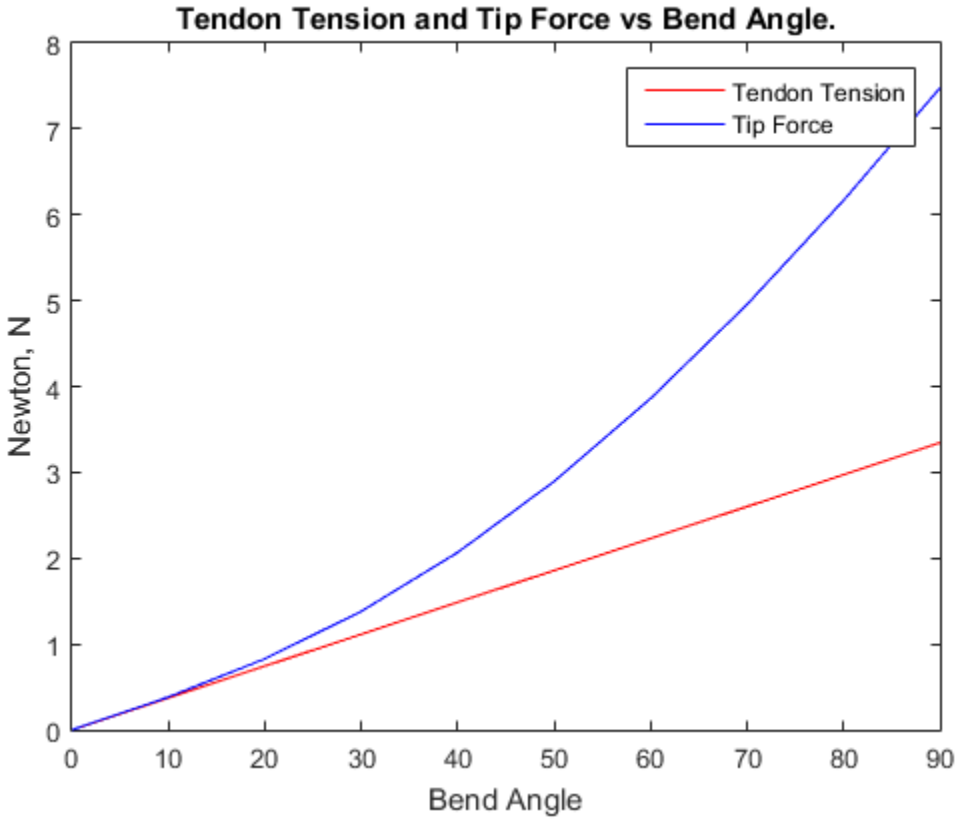
```
E = 1.707;
r1 = 3.175;
r2 = 1.5875;
d = -1;
x = -1;
th = linspace(0,pi/2,10);
s = 60;
l = s;
```

Commute Tendon Loading and Tip Force.

```
I = (pi/4)*((r1^4)-(r2^4));
T = -(E*I.*th)./(d.*s);
Ft = sqrt(((T.*sin(th)).^2) + ((T-T.*(cos(th)-1)).^2));
```

Plot Tendon Loading and Tip Force.

```
plot(th*180/pi,T,'r');
hold on
plot(th*180/pi,Ft,'b');
ylabel('Newton, N')
xlabel('Bend Angle')
title ('Tendon Tension and Tip Force vs Bend Angle.')
legend ('Tendon Tension','Tip Force')
```



Display Angle, Tension Force and Tip Force.

```
Output = ['Angle' 'Tension Force' 'Tip Force'];
disp(Output);
disp([linspace(0,90,10)' T' Ft'])
```

```
AngleTension ForceTip Force
         0         0         0
   10.0000    0.3715    0.3827
   20.0000    0.7431    0.8279
   30.0000    1.1146    1.3813
   40.0000    1.4861    2.0677
   50.0000    1.8577    2.8951
   60.0000    2.2292    3.8611
   70.0000    2.6007    4.9563
   80.0000    2.9723    6.1673
   90.0000    3.3438    7.4769
```

Published with MATLAB® R2015a

Bibliography

[1] Oliver W Hakenberg et al. “Bladder wall thickness in normal adults and men with mild lower urinary tract symptoms and benign prostatic enlargement”. In: *Neurourology and urodynamics* 19.5 (2000), pp. 585–593.

[2] Nid. *The Urinary Tract & How It Works*. July 2017. URL: https://www.niddk.nih.gov/health-information/urologic-diseases/urinary-tract-how-it-works.

[3] Walter OBrien et al. “Overactive Bladder-Sensory Urgency”. The Urology Group.

[4] E. *Infections of the Urinary System — Clinical Gate*. July 2017. URL: https://clinicalgate.com/infections-of-the-urinary-system/.

[5] Ezekiel H Landau et al. “Loss of elasticity in dysfunctional bladders: urodynamic and histochemical correlation”. In: *The Journal of urology* 152.2 (1994), pp. 702–705.

[6] NIH. *Bladder Control Problems in Women (Urinary Incontinence)*. [Online; Stand 19. Dezember 2012]. 2017. URL: https://www.niddk.nih.gov/health-information/urologic-diseases/bladder-control-problems-women.

[7] Ivan Nenadic et al. “Noninvasive Evaluation of Bladder Wall Mechanical Properties as a Function of Filling Volume: Potential Application in Bladder Compliance Assessment”. In: *PloS one* 11.6 (2016), e0157818.

[8] Chunhui Li et al. “Quantitative elasticity measurement of urinary bladder wall using laser-induced surface acoustic waves”. In: *Biomedical optics express* 5.12 (2014), pp. 4313–4328.

[9] Johns Hopkins Medicine Health Library. *Cystometry*. Johns Hopkins. Aug. 2017. URL: http://www.hopkinsmedicine.org/healthlibrary/test_procedures/urology/cystometry_92,P07718/.

[10] Jerry G Blaivas, Asnat Groutz, Michael Verhaaren, et al. “Does the method of cystometry affect the incidence of involuntary detrusor contractions? A prospective randomized urodynamic study”. In: *Neurourology and urodynamics* 20.2 (2001), pp. 141–145.

[11] van Doorn ES Van Waalwijk, A Remmers, and RA Janknegt. "Conventional and extramural ambulatory urodynamic testing of the lower urinary tract in female volunteers." In: *The Journal of urology* 147.5 (1992), pp. 1319–25.

[12] MB Siroky, I Goldstein, and RJ Krane. "Functional voiding disorders in men." In: *The Journal of urology* 126.2 (1981), pp. 200–204.

[13] RJ Webb, PD Ramsden, and DE Neal. "Ambulatory monitoring and electronic measurement of urinary leakage in the diagnosis of detrusor instability and incontinence". In: *BJU International* 68.2 (1991), pp. 148–152.

[14] Peter K Sand, Randy C Hill, and Donald R Ostergard. "Supine urethroscopic as screening Methbds Detrusor Instability and Standing Cystometry for the Detection of Detrusor Instability." In: *Obstetrics & Gynecology* 70.1 (1987), pp. 57–60.

[15] Adam J Flisser and Jerry G Blaivas. "Role of cystometry in evaluating patients with overactive bladder". In: *Urology* 60.5 (2002), pp. 33–42.

[16] B Clarke. "The role of urodynamic assessment in the diagnosis of lower urinary tract disorders". In: *International Urogynecology Journal* 8.4 (1997), pp. 196–199.

[17] Jonathan Ophir et al. "Elastography: a quantitative method for imaging the elasticity of biological tissues". In: *Ultrasonic imaging* 13.2 (1991), pp. 111–134.

[18] J-L Gennisson et al. "Ultrasound elastography: principles and techniques". In: *Diagnostic and interventional imaging* 94.5 (2013), pp. 487–495.

[19] Selcen Kanyilmaz et al. "Bladder wall thickness and ultrasound estimated bladder weight in healthy adults with portative ultrasound device". In: *Journal of research in medical sciences: the official journal of Isfahan University of Medical Sciences* 18.2 (2013), p. 103.

[20] Su-Lin Lee et al. "From medical images to minimally invasive intervention: Computer assistance for robotic surgery". In: *Computerized Medical Imaging and Graphics* 34.1 (2010), pp. 33–45.

[21] B Jaffray. "Minimally invasive surgery". In: *Archives of disease in childhood* 90.5 (2005), pp. 537–542.

[22] Mark Vierra MD. "Minimally invasive surgery". In: *Annual review of medicine* 46.1 (1995), pp. 147–158.

[23] Deepak Trivedi et al. "Soft robotics: Biological inspiration, state of the art, and future research". In: *Applied Bionics and Biomechanics* 5.3 (2008), pp. 99–117.

[24] Jessica Burgner-Kahrs, D Caleb Rucker, and Howie Choset. "Continuum robots for medical applications: A survey". In: *IEEE Transactions on Robotics* 31.6 (2015), pp. 1261–1280.

[25] Pierre E Dupont et al. "Design and control of concentric-tube robots". In: *IEEE Transactions on Robotics* 26.2 (2010), pp. 209–225.

[26] Robert J Webster III, Joseph M Romano, and Noah J Cowan. "Mechanics of precurved-tube continuum robots". In: *IEEE Transactions on Robotics* 25.1 (2009), pp. 67–78.

[27] Paolo Dario et al. "A novel mechatronic tool for computer-assisted arthroscopy". In: *IEEE transactions on information technology in biomedicine* 4.1 (2000), pp. 15–29.

[28] Michael DM Kutzer et al. "Design of a new cable-driven manipulator with a large open lumen: Preliminary applications in the minimally-invasive removal of osteolysis". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2913–2920.

[29] David B Camarillo et al. "Mechanics modeling of tendon-driven continuum manipulators". In: *IEEE Transactions on Robotics* 24.6 (2008), pp. 1262–1273.

[30] Hyun-Soo Yoon and Byung-Ju Yi. "A 4-DOF flexible continuum robot using a spring backbone". In: *Mechatronics and Automation, 2009. ICMA 2009. International Conference on*. IEEE. 2009, pp. 1249–1254.

[31] Yong-Jae Kim et al. “Design of a tubular snake-like manipulator with stiffening capability by layer jamming”. In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 4251–4256.

[32] Allen Jiang et al. “The granular jamming integrated actuator”. In: *Advanced Robotics and Intelligent Systems (ARIS), 2014 International Conference on*. IEEE. 2014, pp. 12–17.

[33] Yusof Ganji, Farrokh Janabi-Sharifi, et al. “Catheter kinematics for intracardiac navigation”. In: *IEEE Transactions on Biomedical Engineering* 56.3 (2009), pp. 621–632.

[34] William S Rone and Pinhas Ben-Tzvi. “Mechanics modeling of multisegment rod-driven continuum robots”. In: *Journal of Mechanisms and Robotics* 6.4 (2014), p. 041006.

[35] Mohammad Dehghani and S Ali A Moosavian. “Modeling of continuum robots with twisted tendon actuation systems”. In: *Robotics and Mechatronics (ICRoM), 2013 First RSI/ISM International Conference on*. IEEE. 2013, pp. 14–19.

[36] D Caleb Rucker et al. “Equilibrium conformations of concentric-tube continuum robots”. In: *The International journal of robotics research* 29.10 (2010), pp. 1263–1280.

[37] Gregory S Chirikjian and Joel W Burdick. “Kinematically optimal hyper-redundant manipulator configurations”. In: *IEEE transactions on Robotics and Automation* 11.6 (1995), pp. 794–806.

[38] Jinwoo Jung, Ryan S Penning, and Michael R Zinn. “A modeling approach for robotic catheters: Effects of nonlinear internal device friction”. In: *Advanced Robotics* 28.8 (2014), pp. 557–572.

[39] Tianjiang Zheng et al. “Dynamic continuum arm model for use with underwater robotic manipulators inspired by octopus vulgaris”. In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 5289–5294.

[40] Kai Xu and Nabil Simaan. "Analytic formulation for kinematics, statics, and shape restoration of multibackbone continuum robots via elliptic integrals". In: *Journal of Mechanisms and Robotics* 2.1 (2010), p. 011006.

[41] Jessica Burgner et al. "Debulking from within: a robotic steerable cannula for intracerebral hemorrhage evacuation". In: *IEEE Transactions on Biomedical Engineering* 60.9 (2013), pp. 2567–2575.

[42] Hyun-Soo Yoon et al. "Compact design of a dual master-slave system for maxillary sinus surgery". In: *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE. 2013, pp. 5027–5032.

[43] Nabil Simaan et al. "Design and integration of a telerobotic system for minimally invasive surgery of the throat". In: *The International journal of robotics research* 28.9 (2009), pp. 1134–1153.

[44] Samuel B Kesner and Robert D Howe. "Robotic catheter cardiac ablation combining ultrasound guidance and force control". In: *The International Journal of Robotics Research* 33.4 (2014), pp. 631–644.

[45] Roger E Goldman et al. "Design and performance evaluation of a minimally invasive telerobotic platform for transurethral surveillance and intervention". In: *IEEE transactions on biomedical engineering* 60.4 (2013), pp. 918–925.

[46] Richard J Hendrick et al. "Hand-held transendoscopic robotic manipulators: A transurethral laser prostate surgery case study". In: *The International journal of robotics research* 34.13 (2015), pp. 1559–1572.

[47] Da Vinci Surgery. *Da Vinci Surgery - Minimally Invasive Robotic Surgery with the da Vinci Surgical System*. July 2017. URL: http://www.davincisurgery.com/da-vinci-surgery/da-vinci-surgical-system/?gclid=Cj0KCQjwh_bLBRDeARIsAH4ZYENm0dJMxYw0uhpAe6cMaAnTQEALw_wcB.

[48] Clinica Universidad de Navarra. *Da Vinci: robotic surgery.* Last, First. (2017) Retrieved July 30, 2017, from. July 2017.

[49] NASA. *Robotic Surgery.* July 2017. URL: https://spinoff.nasa.gov/spinoff2000/hm1.htm.

[50] Drug Watch. *Da Vinci Surgical System - FDA Warning, Injuries & Complications.* July 2017. URL: https://www.drugwatch.com/davinci-surgery/.

[51] RoboCatz. *Da Vinci Surgical Robot.* July 2017. URL: http://robocatz.com/daVinci-surgical.htm.

[52] Liliana Ioana V and Mariana Pcurar. "UNILATERAL UPPER MOLAR DISTALIZATION IN A SEVERE CASE OF CLASS II MALOCCLUSION. CASE PRESENTATION." In: *Romanian Journal of Oral Rehabilitation* 4.4 (2012), p. 5.

[53] Paolo Dario, Blake Hannaford, and Arianna Menciassi. "Smart surgical tools and augmenting devices". In: *IEEE transactions on robotics and automation* 19.5 (2003), pp. 782–792.

[54] RH Taylor and D Stoiariovici. "gMedical robotics in computerintegrated surgery, h IEEE Trans". In: *Robot. Automat* 19.5 (2003), pp. 922–926.

[55] P Abolmaesumi, SE Salcudean, and WH Zhu. "Visual servoing for robot-assisted diagnostic ultrasound". In: *Engineering in Medicine and Biology Society, 2000. Proceedings of the 22nd Annual International Conference of the IEEE.* Vol. 4. IEEE. 2000, pp. 2532–2535.

[56] Chrystèle Rubod et al. "Biomechanical properties of human pelvic organs". In: *Urology* 79.4 (2012), 968–e17.

[57] Javier Palacio-Torralba et al. "Patient specific modeling of palpation-based prostate cancer diagnosis: effects of pelvic cavity anatomy and intrabladder pressure". In: *International journal for numerical methods in biomedical engineering* 32.1 (2016).

[58] Jelizaveta Konstantinova et al. “Autonomous robotic palpation of soft tissue using the modulation of applied force”. In: *Biomedical Robotics and Biomechatronics (BioRob), 2016 6th IEEE International Conference on*. IEEE. 2016, pp. 323–328.

[59] Cheng-Jen Chuong et al. “Viscoelastic properties measurement of the prolapsed anterior vaginal wall: a patient-directed methodology”. In: *European Journal of Obstetrics & Gynecology and Reproductive Biology* 173 (2014), pp. 106–112.

[60] Katsuaki Yokoyama et al. “Novel contact force sensor incorporated in irrigated radiofrequency ablation catheter predicts lesion size and incidence of steam pop and thrombus”. In: *Circulation: Arrhythmia and Electrophysiology* 1.5 (2008), pp. 354–362.

[61] Shashank Sanjay Kumat. “A Robotic Device to Assist with In-Vivo Measurement of Human Pelvic Organ Tissue Properties”. MA thesis. The University of Texas at Arlington, 2016.

[62] John J Craig. *Introduction to robotics: mechanics and control*. Vol. 3. 63. Pearson Prentice Hall Upper Saddle River, 2005.

[63] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. Vol. 3. 119-120,128-133,140. wiley New York, 2006.

[64] Stefan Schaal. *Jacobian methods for inverse kinematics and planning*. University Lecture.

[65] Alessandro De Luca. “Differential kinematics”. University Lecture.

[66] Michael Milford. *Velocity, Force and the Jacobian*. Youtube. 2017. URL: https://www.youtube.com/watch?v=5DJ10praEZ8&list=LLwbKB6ejFzqAAF_yDy0G7WA&index=23.

[67] Yang Cao. *Analytical Jacobian*. Youtube. 2017. URL: https://www.youtube.com/watch?v=wLoC_el3Sis&list=LLwbKB6ejFzqAAF_yDy0G7WA&index=27.

[68] Richard P Paul and Bruce Shimano. "Kinematic control equations for simple manipulators". In: *Decision and Control including the 17th Symposium on Adaptive Processes, 1978 IEEE Conference on*. IEEE. 1979, pp. 1398–1406.

[69] GORDON E MAYER. "Differential kinematic control equations for simple manipulators". In: *IEEE Transactions on Systems, Man, and Cybernetics* 1.1 (1981).

[70] Bruno Siciliano et al. "Jacobian-based algorithms: A bridge between kinematics and control". In: *Proceedings of the Special Celebratory Symposium In the honor of Professor Bernie Roths 70th Birthday*. 2003, pp. 4–35.

[71] Ashleigh Fay. *Bladder Volume*. July 2017. URL: `http://www.em.emory.edu/ultrasound/ImageWeek/Abdominal/bladder_volume.html`.

[72] Wikipidia. *Theorem of the three geodesics.* July 2017. URL: `https://en.wikipedia.org/wiki/Theorem_of_the_three_geodesics`.

[73] Adria Colomé and Carme Torras. "Closed-loop inverse kinematics for redundant robots: Comparative assessment and two enhancements". In: *IEEE/ASME Transactions on Mechatronics* 20.2 (2015), pp. 944–955.

[74] Stephen K Chan and Peter D Lawrence. "General inverse kinematics with the error damped pseudoinverse". In: *Robotics and Automation, 1988. Proceedings., 1988 IEEE International Conference on*. IEEE. 1988, pp. 834–839.

[75] Abigyat B Prasai et al. "Design and fabrication of a disposable micro end effector for concentric tube robots". In: *Control, Automation, Robotics and Vision (ICARCV), 2016 14th International Conference on*. IEEE. 2016, pp. 1–6.

[76] Amesweb. *Shear Stress (Single and Double Shear)*. URL: `http://www.amesweb.info/StressStrainTransformations/Shear-Stress.aspx`.

[77] Llc Engineers Edge. *Bolt or Pin In Single Shear Equation and Calculator*. July 2017. URL: `http://www.engineersedge.com/material_science/bolt_single_shear_calcs.htm`.

[78] Ecourses. *Mechanics eBook: Shear and Bearing Stress*. July 2017. URL: http://www.ecourses.ou.edu/cgi-bin/eBook.cgi?topic=me&chap_sec=01.2&page=theory.

BIOGRAPHICAL STATEMENT

Samson Adejokun received a bachalor's degree from Ajayi Crowther University (ACU), Nigeria in 2011. He graduate with a First class in Physics with Electronics and best graduating student in his faculty. In 2013, he received a master's degree in Robotics from King's College London. While at King's, Samson researched on developing soft robotic actuators with variable stiffness for minimally invasive surgery and co-authored a paper titled *The Granular Jamming Integrated Actuator*. At UTA, Samson has continued to focus his research interest on the medical application of robots and has worked as a graduate teaching assistant for the Measurement and Instrumentation II lab. Samson has also worked as a robotics research intern at the University of Texas Arlington Research Institute (UTARI) where he was involved in developing soft pneumatic actuators and a robotic rehab glove for applications in rehabilitative and assistive medical care. Samson hopes to begin his doctoral studies at UTA in the fall of 2017 where he will continue to pursue his interest in medical and assistive robotics which may include soft robots, actuator and sensor systems development. Samson currently has over a year of industrial engineering experience and in future plans to pursue a career in robotics research and development.