Analysis and Modeling of Wire-actuated Wrist with a Universal Joint

By

Zhangshi Liu

Thesis

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Mechanical Engineering

August, 2016

Nashville, Tennessee

Approved: Date:

Prof. Nabil Simaan, Ph.D.

Prof. Karl Zelik, Ph.D.

Prof. Thomas J. Withrow, Ph.D.

ACKNOWLEDGEMENTS

This work is based on the research of design and synthesis of wire-actuated universal joint wrists for surgical application by Saleem Abdul Hamid and Dr. Nabil Simaan, I am grateful for my advisor, my colleagues, friends and my parents in the process of finishing my thesis work.

I would first like to thank my advisor Dr. Nabil Simaan for his consistent guidance and continuous support of my graduate study and research. His patience, immense knowledge and diligence always encouraged and enlightened me in the course of pursuing my master degree. His passion and dedication to work will always be the exemplar for me in my future career. He not only trained me as professional engineer but also taught me how to be an independent thinker.

Second, I would like to thank my colleagues and friends, Long Wang, Haoran Yu, Nima Sarli, Dr. Jason Pile, Dr. Rajarshi Roy, Giuseppe Del Giudice, Rashid Yasin, Saleem Abdul Hamid and Dr. Andrea Bajo. They always gave me a solid hand when I got stuck in difficulties. Without their attentive and sincere help, this work cannot be finished.

Finally, I want to thank my parents Shijun Run and Kongzhi Liu, as well as my dear Grannies. Your deep love is always my greatest strength. Thank you for being in my life.

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Motivation

Robots have had a profound influence on our society. In industry, for example, robots are broadly used in areas of production, inspection and quality control. In public services, robots can be applied to exploration, rescue, surveillance, medicine and health care. In the medical area, robots are used to improve the safety and consistency of surgical procedures, as well as the ability to minimize traumatic and disfiguring incisions to access target organs [19]. During surgery, the role of a surgeon of course has some irreplaceable abilities such as more initiative and flexibility. However, robots also have advantages over surgeons, which usually cannot be gained by training due to human's physical limitation. To make a comparison, Taylor and Joskowicz have listed strengths and limitations of medical robots and humans in Table 1.1 [1]. From the table, we can see that most of the strengths and limitations are complementary. That is to say, the robot can be used as an assistant in a surgery providing the surgeon with a new set of very versatile tools that extend her or his ability to treat patients [20].

Minimally Invasive Surgery (MIS) is a surgical procedure that uses arthroscopic, laparoscopic or customized devices to conduct remote-control manipulation of instruments with indirect observation through skin, body cavity or small anatomical opening [21]. Though MIS has advantages of reducing surgical trauma, shortening hospital stays, accelerating patient recovery and reducing rate of complications [22], the drawbacks are significant such as poor instrument control and ergonomics caused by rigid instrumentation and its associated fulcrum effect [23]. That is, the rigid laparoscopic surgical tools are limited to 4 Degrees of Freedom (DOF) in MIS. However, since most surgical tasks, such as suturing, knot tying, tissue separation, retraction, ablation along a path, etc, require more

Table 1.1: Comparison of medical robots and human [1]

	Strengths	Limitations
Humans	Excellent judgment Excellent hand-eye coordination Excellent dexterity(at natural ”human” scale) Versatile and able to improvise Easily trained Able to integrate and act on multiple information sources	Prone to fatigue and inattention Tremor limits fine motion Limited manipulation ability and dexterity outside natural scale Cannot see through tissue Bulky end-effectors(hands) Affected by radiation and infection Hard to keep sterile Limited geometric accuracy
Robots	Excellent geometric accuracy Untiring and stable Immune to ionizing radiation Can be operated at many different scales of motion and payload Able to integrate multiple sources of numerical and sensor data	Poor judgment Hard to adapt to new situations Limited dexterity Limited ability to integrate and interpret complex information Limited hand-eye coordination and limited haptic sensing

than 5 DOF, how to gain distal dexterity for manipulators is a prerequisite to being able to reap the benefits of MIS.

Miniaturization of the manipulator requires remote actuators and usually uses wires as actuation transmission. Current wire-actuated wrists are predominantly designed with serial architecture because they are relatively easy to design and analyze. However, compared to serial wrists, parallel architecture can offer higher precision, stiffness and payload-to-weight ratio. But wire actuated parallel wrists are difficult to analyze due to singularity within the workspace and due to uni-sense wrench limitations. Moreover, previous works for wire-actuated robots primarily focused on wrench closure workspace that contains a set of poses such that all the wires can work in tension to balance any external wrench [24]. And also, these projects have limited consideration of the effect of wire stiffness on the reduction of wrench-feasible workspace. In this work, we will create kinematic and static models for wire-actuated wrists, and investigate both the effect of wire stiffness on wrench closure workspace and the use of actuation redundancy for enlarging the workspace.

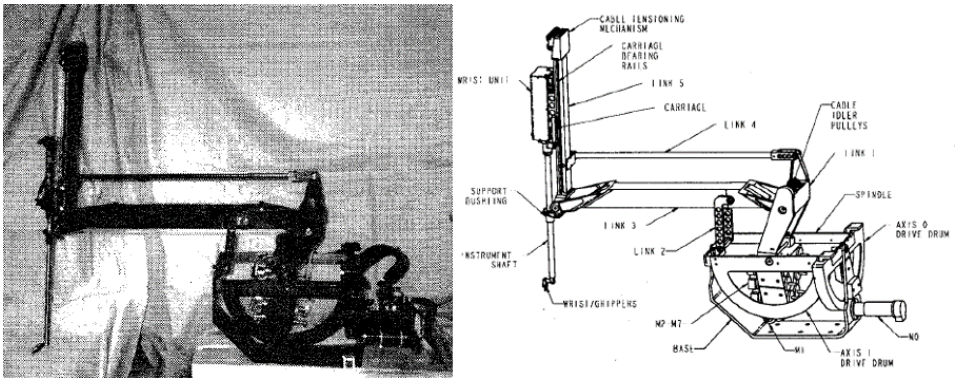

Figure 1.1: Structure of Black Falcon [2]

1.2 Robotic Platforms for Surgical Assistance

We have discussed that robots can act as surgical assistants for MIS, and the problem of how to gain more dexterity has attracted many scholars, surgeons and companies to develop various robot platforms, such as the Black Falcon robot system by Madhani et al. [2], Insertable Robotic Effectors Platform (IREP) by Simaan et al. [3], Robotic Surgical Platform by Lee, et al. [5], and *daVinci*® Surgical System from Intuitive Surgical, Inc [16], etc.

The Black Falcon, shown in Figure 1.1, is an 8-DOF cable driven tele-operator slave robot platform for MIS, which consists of two main subsystems. One is the base unit containing all of the actuators and the other one is the wrist unit which has a mechanical attachment, an instrument shaft and an end-effector [2].

Insertable Robotic Effectors Platform (IREP) is designed for solving problems such as instrument miniaturization, dexterity and collision avoidance between surgical tools operating in confined spaces for MIS, Single Port Access Surgery (SPAS) and Natural Orifice Transluminal Endoscopic Surgery (NOTES) [4][3]. This platform consists of parallelogram mechanism, continuum snake-like arms, wire-actuated wrist, camera module and passive flexible components, as shown in Figure 1.2.

The Robotic Surgical Platform is developed by Jusuk Lee's robotics group at the Sam-

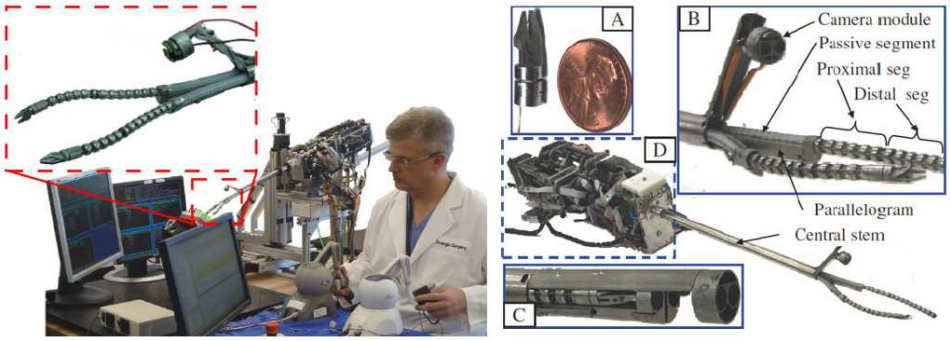

Figure 1.2: Insertable Robotic Effectors Platform (IREP) [3][4]

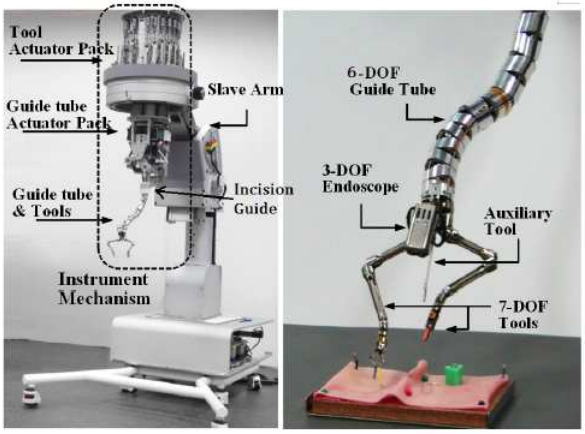

Figure 1.3: Robotic Surgical Platform [5]

sung Advanced Institute of Technology (SAIT). From Figure 1.3 we can see that it comprises of a snake-like 6-DOF guide tube, two 7-DOF tools, a 5-DOF slave arm and a 3-DOF stereo camera, capable of reaching various surgical sites inside the abdominal cavity from a single incision on the body [5].

The *daVinci*® surgical system is one of the most famous surgical assistant platforms in the world, which is composed of a surgical arm cart that is a manipulator unit consisting of several instrument arms, a master console where the surgeon handles telemanipulators and optical controls using three-dimensional vision, and a conventional monitor cart, as shown in Figure 1.4. This surgical system is good at multi-port minimally invasive operations in dealing with delicate and vulnerable anatomical structures [6].

Figure 1.4: *daVinci*® surgical system [6]

1.3 Wrist Classification for Minimally Invasive Surgery

One of the most important components of these surgical platforms is the robot manipulator which is used for procedures such as cutting tissue and suturing trauma. Since in MIS the target is supposed to be reached through a single incision on the body, the motions of rigid laparoscopic surgical tools manipulating tissue are constrained to a pivot, which has only 4 DOF [16], namely two tilting angles about the pivot point and translation and rotation about the longitudinal axis of the tools. However, most surgical tasks such as suturing, knot tying, tissue separation, retraction, ablation along a path, require more than 5 DOF. Thus, the main concerns of dexterity improvement focus on how to restore the degrees of freedom and provide distal dexterity for those operations. Manipulation with good distal dexterity can shorten execution time, reduce surgical trauma and blood loss, and proper mechanism design of the wrist can remove the limitations within surgical environment.

The current surgical wrists of manipulators have various mechanisms with regard to different purposes or under different conditions. However, no matter how discrepant the wrists seem from each other, they all have certain degrees of freedom and consist of three kinds of joints: Roll, Pitch and Yaw. Based on this, wrists can be classified by their DOF, such as 2-DOF, 3-DOF and 4-DOF wrists where the DOF of gripper/forceps is not taken into consideration. Moreover, a sub-classification can be built according to different com-

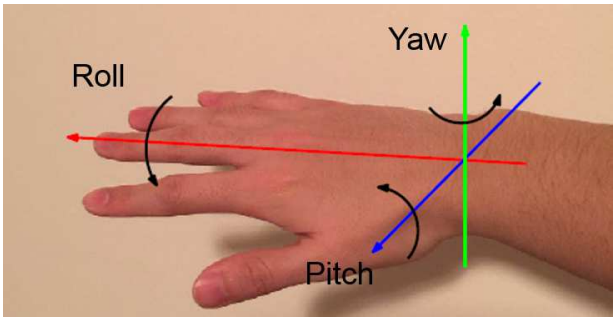

Figure 1.5: Joint type of Roll, Pitch and Yaw

binations of joint types: roll (R), pitch (P) or yaw (Y) (Figure 1.5 shows the axis for each joint type using a human hand). For example, the 2-DOF wrists may have sub-categories of RP wrist (rotations about Roll axis and Pitch axis), RY wrist (rotations about Roll axis and Yaw axis) and PY wrist (rotations about Pitch axis and Yaw axis); For 3-DOF, we basically have RYP wrist (rotations about Roll axis first, then Yaw axis and finally Pitch axis) and PYR wrist(rotations about Pitch axis first, then Yaw axis and finally Roll axis). There is also one example of 4-DOF wrist which can realize rotations of RPPY (rotations about Roll, Pitch, Pitch and Yaw axis sequentially). The classification is listed in Table 1.2.

One example of a 2-DOF wrist is a mechanism for dexterous end effector placement during Minimally Invasive Surgery (MIS) [7] designed by Minor and Mukherjee, et al., as shown in Figure 1.6. This Dexterous Articulated Linkage for Surgical Applications (DALSA), which is a geared serial link mechanism, provides motions of articulation and end effector rotation about the articulated axis. It provides RY and RP rotations via 180 degrees bi-directional tip articulation and unlimited rotation about the articulated longitudinal axis. Articulation is divided among several links, to provide encircling capability and improved reachability. Disadvantages of DALSA include that since it is a two DOF mechanism, it is incapable of placing sutures with arbitrary orientation at surgical sites without rotating the port; another is due to gear backlash which inevitably degrades tip placement, accuracy and repeatability.

Kim, et al. [8], designed a 2 DOF PY stiffness-adjustable snake-like mechanism, as

Table 1.2: Classification of Surgical Wrists

	2-DOF		3-DOF		4-DOF
	RY	PY	RYP	PYR	RPPY
Minor, et al., Articulated Manipulator for MIS [7]	✓				
Kim, et al., Variable neutral-line manipulator [8]		✓			
Breedveld, et al., Endo-Periscope [9]		✓			
Seow, et al., Articulated manipulator [10]		✓			
Harada, et al., Micro manipulator [11]		✓			
Merlet, et al., Parallel Micro Manipulator [12]		✓			
Nakamura, et al., Multi-DOF Forceps Manipulator [13]		✓			
Awtar, et al., End-effector for $FlexDex^{TM}$ [14]			✓		
Takahashi, et al., Link driven type multiple d.o.f. forceps [15]			✓		
Guthart, et al., Endoscopic $EndoWrist^{TM}$ Instrument [16]			✓		
Tadano, et al., A forceps with force sensing using pneumatic servo system [17]			✓		
Simaan, et al., Multi-backbone bending snake-like units [18][25]				✓	
Lee, et al., 3-DOF wrist for SAIT single port access surgical robot [5]				✓	
Madhani, et al., A 4-DOF wrist for Black Falcon [2]					✓

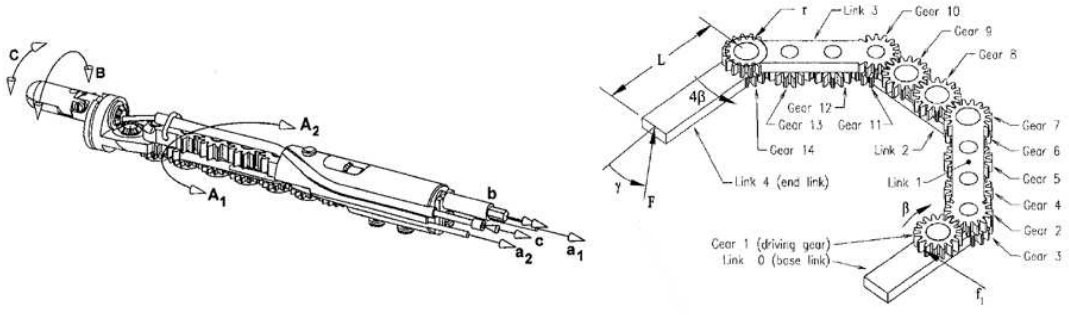

Figure 1.6: Structure of 2 DOF geared rolling wrist DALSA [7]

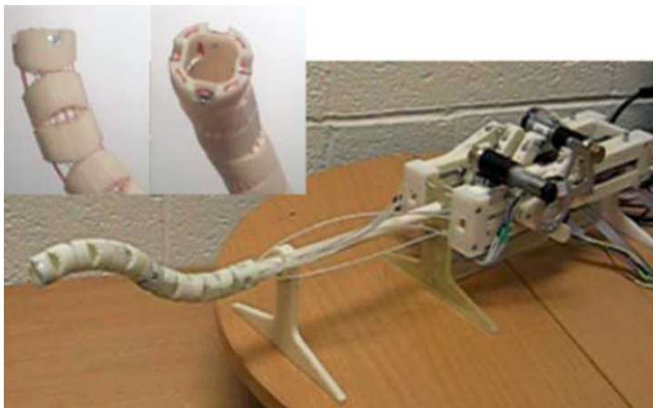

Figure 1.7: 2 DOF stiffness-adjustable snake-like mechanism [8]

shown in Figure 1.7. Each link has two cylindrical contact surfaces oriented orthogonally to each other. There are two wire pairs of which each is in control of Pitch or Yaw motion, and the motion of the two pairs affects each other. Moreover, its simple, thin and hollow structure is suitable for surgical application such as MIS or Natural Orifice Translumenal Endoscopic Surgery (NOTES). The stiffness of this mechanism can be continuously adjusted by varying the wire tension. Experiments show that when wire tension varying from $20N$ to $59.9N$, the stiffness will change from $0.242N/mm$ to $0.529N/mm$, and the max bending angle can reach 90^o.

Another example of 2-DOF PY wrist is the Endo-Periscope designed by Breedveld, et al. [9] in Figure 1.8. This device is to provide visual feedback during laparoscopic surgery by attaching a camera on its tip. The wrist is a spring that combines high torsion stiffness with a low bending stiffness. Four cables are guided through the ring springs. When in straight position, the two ring springs are completely compressed. When the handgrip is bent, part A of the cable becomes longer. However, part B of the cable cannot be shortened since the ring spring in the tip is completely compressed. Instead, part C of the cable becomes shorter. Thus the spring in the handgrip becomes shorter and as a result, the three other cables are released. Then the tip will bend until it reaches the same angles as the handgrip. This wrist can enable the camera sitting on its top to rotate about Pitch or Yaw axis over 180 degrees [9].

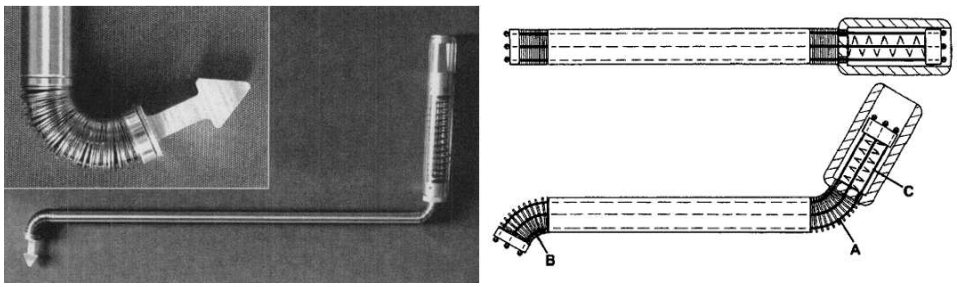

Figure 1.8: 2 DOF wire actuated bending joint [9]

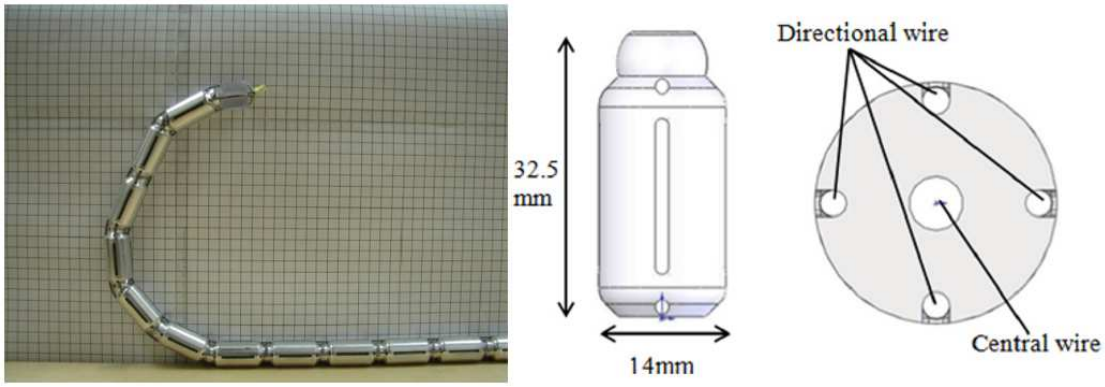

Figure 1.9: 2 DOF wire actuated revolved sliding joint [10]

Figure 1.9 shows a 2-DOF PY cable driven robot arm which consists of 18 revolved sliding joints arranged serially [10]. It is an articulated manipulator with multiple instruments for natural orifice endoscopic transluminal endoscopic surgery (NOTES), which aims to reduce infection risk, improve surgical workflow and encourage solo surgery by providing surgeons with all the required instruments. The sliding joints mainly refer to the robot connecting arm. Each of the joint has dome-shaped top and matching concave bottom of the linkage piece enabling it to rotate relatively to its neighboring piece. From Figure 1.9, the wrist's PY rotations are controlled by four directional wires passing through the linkage pieces. Each opposing pair of wires works antagonistically to provide two rotational DOF in yaw and pitch. Experiments show that the manipulator can provide at least 100 degrees left angular displacement and 107 degrees right angular displacement [10].

Another example of 2 DOF PY wrist is the micro manipulator for intrauterine fetal surgery under Open Magnetic Resonance Imaging (MRI) conditions by Harada, et al. [11].

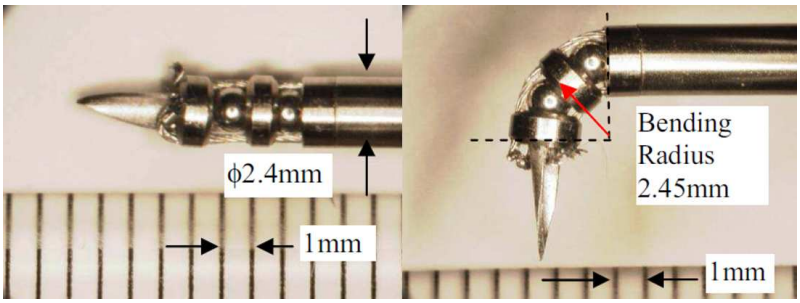

Figure 1.10: 2 DOF wire actuated ball sliding joint [11]

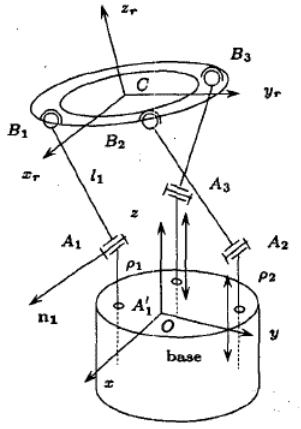

Figure 1.11: Micro parallel wrist [12]

The wrist consists of two ball joints and is driven by four wires to bend through 90 degrees in Pitch and Yaw directions, as shown in Figure 1.10. The diameter of the balls is 2.4mm and the bending radius is 2.45mm. This kind of joint is easy to control, but at the same time, it is easily susceptible to spin. Moreover, there is an inner hole through all ball joints left for future surgical application.

Merlet et al. designed a PY parallel wrist for micro-macro robot in minimally invasive surgery [12], where the ”macro” part, referring to the classical tool of endoscope, has a large workspace with poor accuracy while the ”micro” part, namely the wrist, has small workspace with high accuracy. The wrist shown in Figure 1.11, which is put at the end of the endoscope, has 3-DOF: 2 rotation DOF around Pitch and Yaw axis and one translation along z axis. If we do not take translation into account, the wrist can be regarded as a 2-DOF PY wrist.

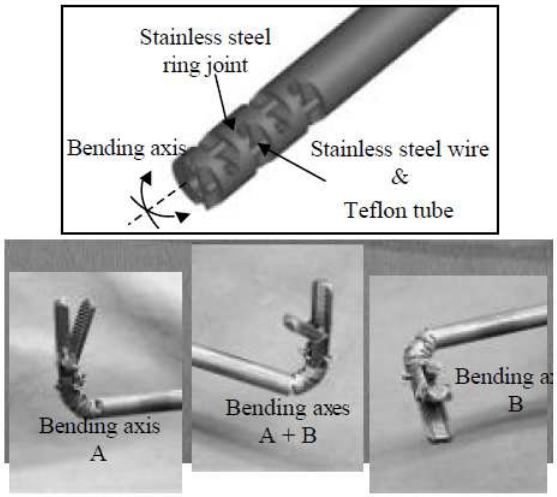

Figure 1.12: 2-DOF wire-driven bending mechanism [13]

Figure 1.12 shows a 2-DOF bending mechanism designed by Nakamura, et al., for laparoscopic surgery [13]. The upper figure is the structure of the wrist and the lower ones show the example of forceps' motions. The wrist, which is driven by four stainless steel wires, provides 2 additional DOF of Pitch and Yaw bending compared with previous forceps manipulator. The ranges of bending motion are from 0 to 90 degrees.

Awtar, et al. presented a 3-DOF RYP end effector for a MIS tool [14] in Figure 1.13. The MIS tool is designed to be attached to the surgeon's forearm, forming an extension for hand and arm. The end effector's wrist is a wire actuated two-hinge output joint in which the two rotational axes lie in a common axial plane. This wrist is designed to not only provide a tight workspace but also eliminate output joint motion coupling in order to meet the objective of one-to-one motion mapping between the input and output. The mechanism's outer ring is pivoted with respect to the tool shaft about a yaw axis and an inner ring is pivoted with respect to the outer ring about a pitch axis. The two ends of the yaw transmission cable are attached at two diametrically opposite points on the outer ring along the pitch axis while the two ends of the pitch transmission cable are attached at two diametrically opposite points on the inner ring that line up along the yaw axis.

Figure 1.14 is also an example of 3-DOF RYP wrist by Takahashi, et al. It is a forceps

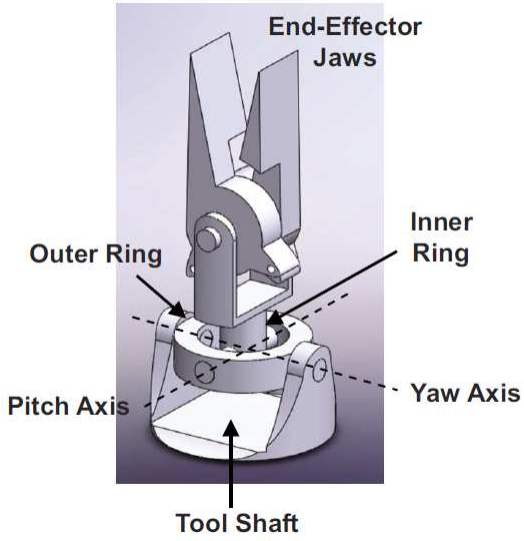

Figure 1.13: 2-DOF wire actuated universal joint [14]

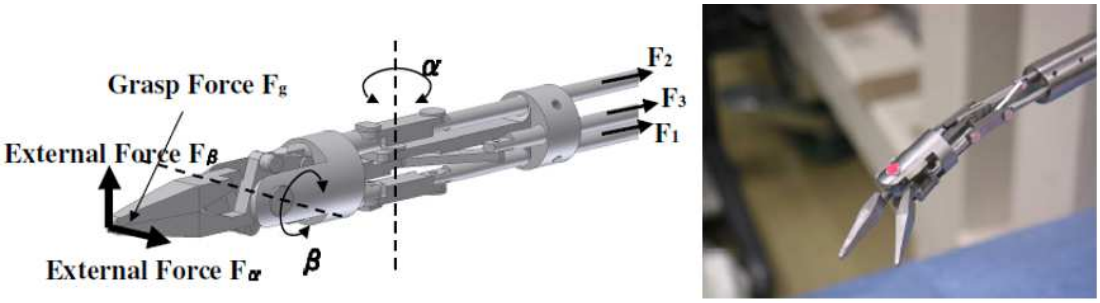

Figure 1.14: 4-DOF forceps manipulator [15]

manipulator that has a roll joint, two bending joints(Yaw and Pitch joints) and a holder. The driving part has 4 DOF in total such as linear motions in 3 DOF and one rotation. The linear motions are converted to bending motions in 2 DOF and the grasping motion at the end effector by the link mechanism. Each joint in the manipulator is actuated by a pneumatic cylinder, which generates torque using a rack and pinion [15].

The steerable grasper EndoWrist^{TM} of da Vinci system from Intuitive Surgical is a 3-DOF RYP mechanism, which was used to restore the degrees of freedom lost in laparoscopy by being placed inside the patient and controlled naturally [16]. The wrist itself can be rotated along the roll axis and it also has a pair of perpendicular joints (pitch and yaw), where the pitch joint is a belt-actuated and the yaw rotates about hinged pulley [26].

Figure 1.15: Endoscopic EndoWrist^{TM} Instruments [16]

It can achieve 90 degrees of articulation, and can execute a broad range of surgical procedures by selecting specialized tip design. Figure 1.15 shows the EndoWrist^{TM} with tips of forceps and needle driver.

Another example of 3-DOF wrist is a RYP manipulator for teleoperated laparoscopic surgery in Figure 1.16 designed by Tadano, et al., which use pneumatic cylinders as actuators in order to provide a force display to surgeons without a force sensor because the cylinders can estimate the external force from the driving force and the impedance. The manipulator has one rolling joint and two bending joints (Yaw and Pitch joints), each of which is actuated by a pneumatic cylinder. The diameter is 10mm and the feature of the forceps is that one bending joint and the gripper are realized at the same point, making the tip compact [17].

Simaan, et al. designed a 3-DOF YPR snake-like manipulators [18][25], which consists of a base disk, an end disk, several spacer disks, one primary backbone and three secondary backbones which are made of flexible super-elastic hollow tubes, as shown in Figure 1.17. The primary backbone is fixed to both the base/end disks and other spacer disks while the secondary backbones are only attached to the end disk and they can slide and bend through holes in the base and spacer disks. The Pitch and Yaw rotations can be manipulated by

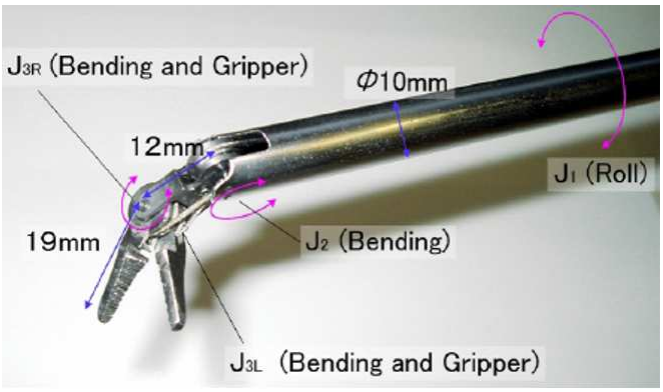

Figure 1.16: Manipulator for laparoscopic surgery [17]

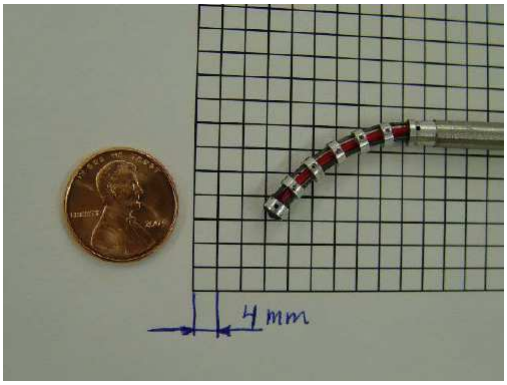

Figure 1.17: 3-DOF YPR multi-backbone snake-like manipulator [18]

actively changing the lengths of two out of the three secondary backbones. Moreover, a detachable milli-parallel unit sitting on the end disk provides the Roll rotation and can be equipped with various tools at the same time, which is driven by super-elastic wires passing through the secondary backbones.

SAIT single port access surgical robot has a 3-DOF PYR wrist, as shown in Figure 1.18. The wrist, part of the tool arm, sits at the end of the guide tube for surgical tasks such as suturing and grasping. Each of the wrist joints (pitch joint, yaw joint and roll joint) is actuated by a pair of wires that originate from the tool actuator pack [5].

Slisbury et al. developed a 4-DOF RPPY wrist for the Black Falcon robot system [2]. Black Falcon is a 8-DOF cable-driven teleoperator slave for MIS, which has a 3-DOF base positioner, a 4-DOF detachable wrist and a 1-DOF gripper. The wrist has Roll-Pitch-Pitch-

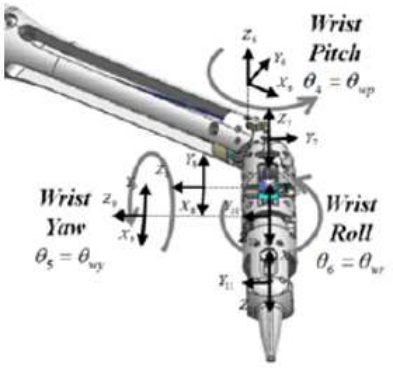

Figure 1.18: 3-DOF PYR wrist of SAIT single port access surgical robot [5]

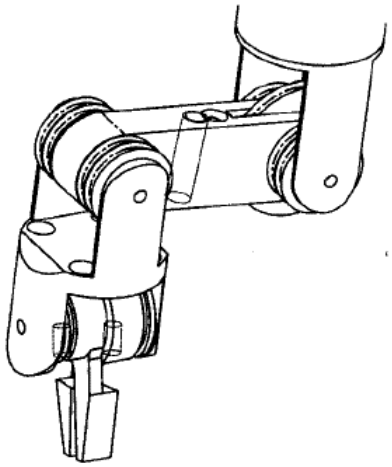

Figure 1.19: 4 DOF wrist for Black Falcon [2]

Yaw joints that the first Roll rotation is about the instrument shaft as shown in Figure 1.19. This wire-driven 4-DOF wrist allows positional redundancy but also has limitations that the 4-DOF structure may occupy too much space while it has essentially the same singularities as a 3-DOF RPY wrist.

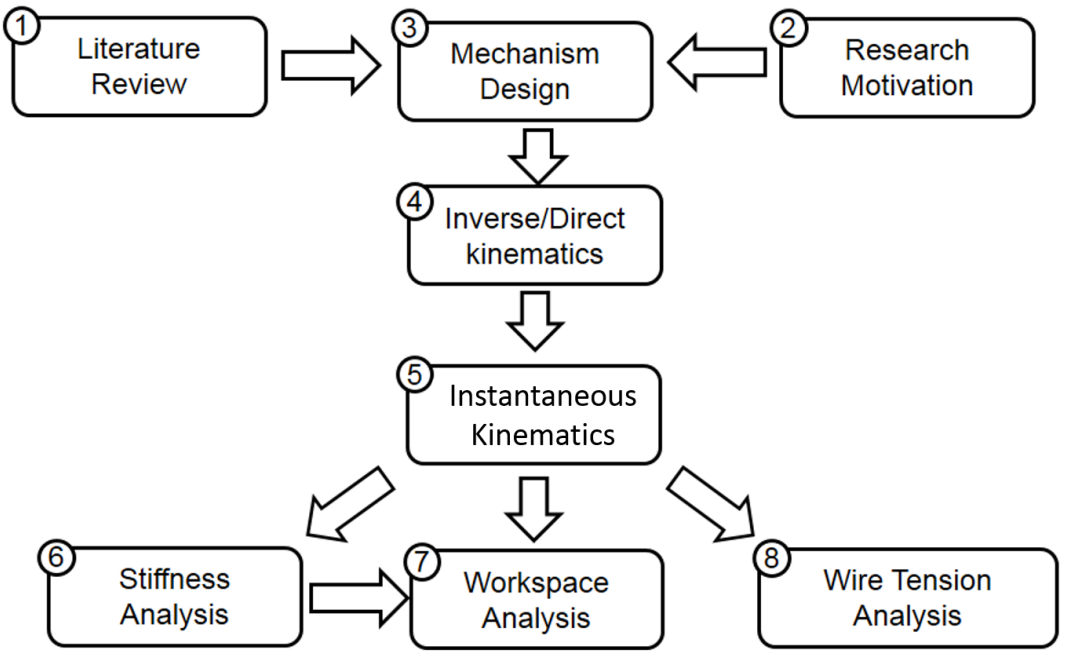

Figure 1.20: Organization of the work

1.4 Organization and Contribution

In this project, we will model, analyze and design a hybrid serial-parallel wire-actuated surgical wrist using universal joint for MIS. A review of research motivation begins the design process and gives an idea about mechanical design. Here we should notice that in the analysis process of this project, we use a scaled-up model where the friction is negligible. Moreover, we will present the inverse/direct kinematic model, and give validations using MatLab. Further, we calculate Jacobian matrix and use it as foundation to conduct stiffness analysis, define workspace assuming joints have infinite/finite stiffness and provide a method to optimize wire tensions. In the last section, we will fabricate a prototype of the wrist and conduct experiments to test the kinematics model and stiffness model. The procedures are shown in Figure 1.20.

Chapter 2

Kinematics and Statics Modeling

2.1 Nomenclature and Mechanism Analysis

The wire actuated wrist to be designed is a 3-DOF mechanism, which consists of a 2-DOF universal joint and 1 rotational joint. Thus it is regarded as a serial-parallel hybrid system, where the universal joint is a parallel mechanism actuated by either 3 wires or 4 wires, and the third rotational joint is serially connected to it. The universal joint has three parts: bottom hook, cross and top hook, and the 3^{rd} DOF will either lie under the bottom hook (case 1) or sit on the top hook (case 2). In this project, we will analyze both the 3-wire and 4-wire mechanisms for each case.

In order to better illustrate the structures, let's define the frames first. In the following nomenclature, we will name 5 coordinate systems: world coordinate system (*WCS,* or frame $\{0\}$) whose axes are $\hat{\mathbf{x}}_{wcs}, \hat{\mathbf{y}}_{wcs}$ and $\hat{\mathbf{z}}_{wcs}$ and origin is O_{wcs}, coordinate system $\{i\}$ (or frame $\{i\}$, $i = 1,2,3,4$) with origins O_i and axes $\hat{\mathbf{x}}_i, \hat{\mathbf{y}}_i$ and $\hat{\mathbf{z}}_i$. The relationships between each frame can be expressed using rotation matrix ${}^{i}\boldsymbol{R}_{j}$ or homogeneous transformation matrix ${}^{i}\boldsymbol{T}_{j}$. Here i and j are notations of frame $\{i\}$ and frame $\{j\}$. Moreover, we call the hook height of bottom hook h_1 and the hook height of top hook h_2.

In case 1, namely the 3^{rd} DOF lying under the bottom hook, the whole universal joint rotates together with it. As shown in Figure 2.1 and Figure 2.2, we first define frame $\{0\}$ (*WCS*), whose axes $\hat{\mathbf{x}}_{wcs}, \hat{\mathbf{y}}_{wcs}$ and $\hat{\mathbf{z}}_{wcs}$ are fixed in the space and origin (O_{wcs}) is coincident with the center of bottom plate. Frame $\{1\}$ (CS_1) has its origin translated by h_1 along $\hat{\mathbf{z}}_0$ and is rotated about $\hat{\mathbf{z}}_0$ by angle α_1. Frame $\{2\}$ (CS_2) has the same origin of frame 1 sitting in the center of the cross and rotates with the cross about axis $\hat{\mathbf{x}}_2$ while $\hat{\mathbf{x}}_2$ remains parallel to $\hat{\mathbf{x}}_1$. The last coordinate system for case 1 is frame $\{3\}$ (CS_3) which is fixed on top hook. O_3 is at the center of top plate and frame 3 rotates around $\hat{\mathbf{y}}_2$ while $\hat{\mathbf{y}}_3$ remains parallel to

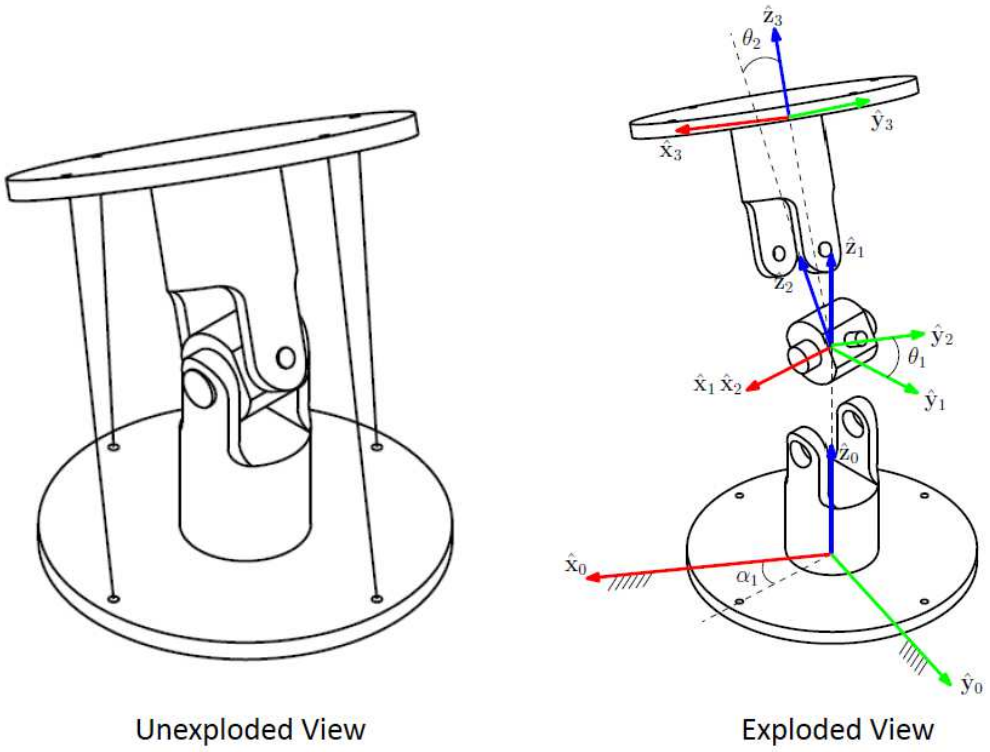

Figure 2.1: Nomenclature for case 1

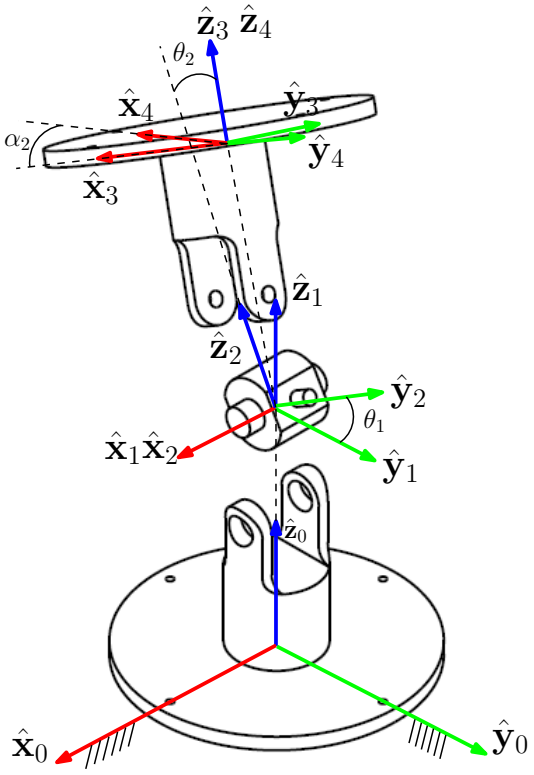

Figure 2.2: Nomenclature for case 2

$\hat{\mathbf{y}}_2$.

In case 2, the third DOF is located on the top hook and frame $\{1\}$ maintains the same orientation as frame $\{0\}$ at all times. Since in case 2 there is no rotation at the base of the universal joint, frame $\{1\}$ is fixed not only on the bottom plate but also in space. Frame $\{2\}$ and $\{3\}$ remain the same with case 1, and Frame $\{4\}$ (CS_4) is added to capture the 3^{rd} DOF α_2 on the top plate, which coincides with frame $\{3\}$ at initial position and rotates about $\hat{\mathbf{z}}_3$. The details of the coordinates can be seen in Figure 2.2. Unlike case 1, in case 2 the rotation of the 3^{rd} DOF will not affect the pose of the universal joint.

We can see that the number of actuation wires does not have influence on locating the coordinates. Moreover, in both case 1 and case 2, no matter where the 3^{rd} DOF is located, it keeps a serial connection to the universal joint. That is to say, it is assumed that actuations for the universal joint and the rotation joint are not coupled. We can compute them separately and then integrate them together in the following analysis.

2.2 Position and Orientation Analysis

In this section, we will analyze inverse and direct kinematics for the hybrid mechanism. For inverse kinematics problem, the input is the end effector's position expressed in world frame, namely the origin position of frame $\{3\}$ (for case 1) or that of frame $\{4\}$ (for case 2), and the outputs are wires' lengths l_1, l_2, l_3 and l_4 as well as the rotation angle α_1 (for case 1) or α_2 (for case 2) of the 3^{rd} DOF. For direct kinematics, the wires' lengths, α_1 and α_2 are given, and we seek to find end effector's position. Since the input for the 3^{rd} DOF can be applied on either the top plate or the bottom plate, we will study these 2 cases for each kinematics analysis. Moreover, in each case, we will analyze both 3-wire and 4-wire joint mechanisms.

2.2.1 Inverse kinematics

2.2.1.1 Inverse kinematics analysis for case 1

In case 1 where the 3^{rd} DOF lies under the bottom plate, frame $\{1\}$ will rotate about $\hat{\mathbf{z}}_0$ with angle α_1. From Figure 2.3 we can see the closed-loop geometry relationships among the vectors. ${}^0\boldsymbol{b}_i$ represents the vector that points from the origin of frame $\{0\}$ to the end of the i^{th} wire on bottom plate. The left superscript indicates the coordinate system in which this vector is expressed and the right subscript refers to the i^{th} wire. Similarly, ${}^0\boldsymbol{a}_i$ refers to the vector pointing from the origin of frame $\{3\}$ to the end of the ith wire on top plate expressed in frame $\{0\}$. Moreover, ${}^0\boldsymbol{t}_1$ and ${}^0\boldsymbol{t}_2$ are the vectors pointing from O_{wcs} to the center of cross O_2 and from O_2 to O_3 respectively. ${}^0\boldsymbol{l}_i$ indicates the ith wire vector connecting the tip of vector ${}^0\boldsymbol{b}_i$ to the tip of vector ${}^0\boldsymbol{a}_i$. Here i can either be $i = 1,2,3$ or $i = 1,2,3,4$ which is based on the number of actuation wires. Thus we can write the following loop closure equation:

$$ {}^0\mathbf{b}_i = {}^0\mathbf{t}_1 + {}^0\mathbf{t}_2 + {}^0\mathbf{a}_i + {}^0\mathbf{l}_i \tag{2.1} $$

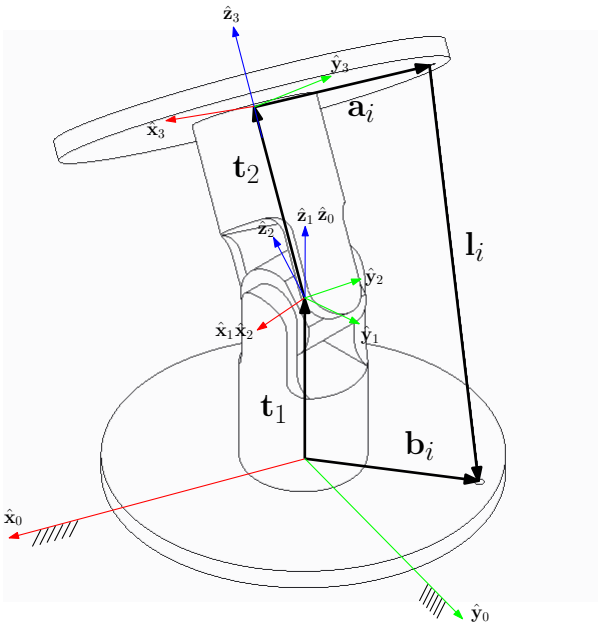

Figure 2.3: Vector loop closure for one actuation wire in case 1

If we use ${}^0\mathbf{EE}_{pos}$ to present end effector's position we can rewrite equation 2.1 as:

$$\begin{aligned} {}^0\mathbf{R}_1\,{}^1\mathbf{b}_i &= \left({}^0\mathbf{R}_1\,{}^1\mathbf{t}_1 + {}^0\mathbf{R}_3\,{}^3\mathbf{t}_2\right) + {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i + {}^0\mathbf{l}_i \\ &= {}^0\mathbf{EE}_{pos} + {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i + {}^0\mathbf{l}_i \end{aligned} \tag{2.2}$$

$$\|{}^0\mathbf{l}_i\| = \|{}^0\mathbf{R}_1\,{}^1\mathbf{b}_i - {}^0\mathbf{EE}_{pos} - {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i\| \tag{2.3}$$

In equation 2.3, ${}^1\mathbf{b}_i$ and ${}^3\mathbf{a}_i$ are fixed vectors, and ${}^0\mathbf{EE}_{pos}$ and ${}^0\mathbf{R}_3$ are known as end effector's position and orientation. ${}^0\mathbf{R}_1$ is function of α_1, which is the only unknown parameter in 2.3. Thus we must find the value of α_1 first and then use equation 2.3 to compute wire lengths. Let us take a look at the structure of the universal joint. If we call the rotation angle of frame $\{2\}$ relative to frame $\{1\}$ θ_1 and call the rotation angle of frame $\{3\}$ relative to frame $\{2\}$ θ_2, we can use θ_1 and θ_2 to represent ${}^0\mathbf{R}_3$. Here we use the

product of exponentials formula:

$$
\begin{aligned}
{}^0\mathbf{R}_3 &= e^{\alpha_1 \hat{\mathbf{z}}_0^\wedge} e^{\theta_1 \hat{\mathbf{x}}_1^\wedge} e^{\theta_2 \hat{\mathbf{y}}_2^\wedge} \\
&= \begin{bmatrix} c\alpha_1 c\theta_2 - s\alpha_1 s\theta_1 s\theta_2 & -s\alpha_1 c\theta_1 & c\alpha_1 s\theta_2 + s\alpha_1 c\theta_2 s\theta_1 \\ s\alpha_1 c\theta_2 + c\alpha_1 s\theta_1 s\theta_2 & c\alpha_1 c\theta_1 & s\alpha_1 s\theta_2 - c\alpha_1 c\theta_2 s\theta_1 \\ -c\theta_1 s\theta_2 & s\theta_1 & c\theta_1 c\theta_2 \end{bmatrix}
\end{aligned} \tag{2.4}
$$

In Equation 2.4, the letters c and s represent short notation for *cos* and *sin*, respectively. The item $e^{\theta_1 \hat{\mathbf{x}}_1^\wedge}$, for example, represents a rotation matrix where the unit vector $\hat{\mathbf{x}}_1$ is the rotation axis and θ_1 is the rotation angle. Here $\hat{\mathbf{x}}_1^\wedge$ means the wedge of $\hat{\mathbf{x}}_1 = [u, v, w]^T$:

$$
{}^0\hat{\mathbf{x}}_1^\wedge = \begin{bmatrix} 0 & -w & v \\ w & 0 & -u \\ -v & u & 0 \end{bmatrix} \tag{2.5}
$$

Moreover, we can present ${}^0\mathbf{R}_3$ by expressing the axes of frame $\{3\}$ in frame $\{0\}$:

$$
\begin{aligned}
{}^0\mathbf{R}_3 &= \begin{bmatrix} {}^0\hat{\mathbf{x}}_3, & {}^0\hat{\mathbf{y}}_3, & {}^0\hat{\mathbf{z}}_3 \end{bmatrix} \\
&= \begin{bmatrix} {}^0x_{3x} & {}^0y_{3x} & {}^0z_{3x} \\ {}^0x_{3y} & {}^0y_{3y} & {}^0z_{3y} \\ {}^0x_{3z} & {}^0y_{3z} & {}^0z_{3z} \end{bmatrix}
\end{aligned} \tag{2.6}
$$

Since ${}^0\mathbf{R}_3$ is given, we can list several equations to solve for θ_1, θ_2 and α by comparing 2.4 and 2.6:

$$
{}^0y_{3z} = \sin\theta_1 \tag{2.7}
$$

$$
\begin{aligned}
{}^0x_{3z} &= -\cos\theta_1 \sin\theta_2 \\
{}^0z_{3z} &= \cos\theta_1 \cos\theta_2
\end{aligned} \tag{2.8}
$$

$$
\begin{aligned}
{}^{0}y_{3x} &= -\sin\alpha_1 \cos\theta_1 \\
{}^{0}y_{3y} &= \cos\alpha_1 \cos\theta_1
\end{aligned}
\tag{2.9}
$$

Equation 2.7 presents two sets of possible solutions for θ_1 [1]:

$$
\theta_1 = \begin{cases}
\text{Atan2}({}^{0}y_{3z}, \sqrt{1 - {}^{0}y_{3z}^2}) + 2k\pi \ (k = 0, \pm 1, \pm 2, \ldots) \\
\text{Atan2}({}^{0}y_{3z}, -\sqrt{1 - {}^{0}y_{3z}^2}) + 2k\pi \ (k = 0, \pm 1, \pm 2, \ldots)
\end{cases}
\tag{2.10}
$$

However, since the joint limit for θ_1 and θ_2 is $[-\frac{\pi}{2}, -\frac{\pi}{2}]$, only 1 value is a valid solution for θ_1. Then substituting θ_1 into equation 2.8 can yield a unique solution for θ_2

$$
\theta_2 = \text{Atan2}\left(\frac{{}^{0}x_{3z}}{-\cos\theta_1}, \frac{{}^{0}z_{3z}}{\cos\theta_1}\right)
\tag{2.11}
$$

Furthermore, we can get values of $\sin\alpha_1$ and $\cos\alpha_1$ in 2.9, and the solutions for α_1 are:

$$
\alpha_1 = \text{Atan2}\left(\frac{{}^{0}y_{3x}}{-\cos\theta_1}, \frac{{}^{0}y_{3y}}{\cos\theta_1}\right) + 2k\pi \ (k = 0, \pm 1, \pm 2, \ldots)
\tag{2.12}
$$

Moreover, because frame 1 rotates about ${}^{0}\hat{\mathbf{z}}_0$, we can write the expression for ${}^{0}\mathbf{R}_1$ directly:

$$
{}^{0}\mathbf{R}_1 = \begin{bmatrix}
\cos\alpha_1 & -\sin\alpha_1 & 0 \\
\sin\alpha_1 & \cos\alpha_1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\tag{2.13}
$$

which contains only one variable α_1. Finally all items in equation 2.3 are obtained so that we can calculate the wire lengths.

[1] We use $\text{Atan2}(\sin(\alpha), \cos(\alpha))$ convention in this thesis.

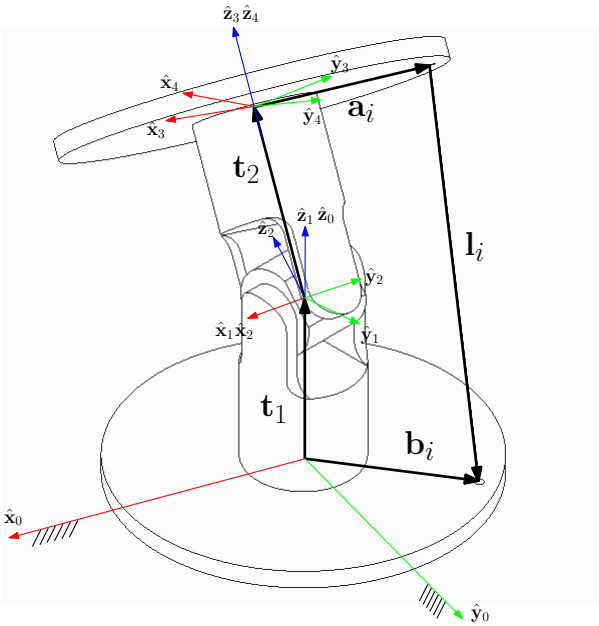

Figure 2.4: Vector loop closure for one actuation wire in case 2

2.2.1.2 Inverse kinematics analysis for case 2

In case 2 where the 3^{rd} DOF sits on the top plate, frame 1 is fixed in frame 0, and frame 4 rotates about $\hat{\mathbf{z}}_3$ with angle α_2 relative to frame 3. Namely, the value of α_2 does not affect the end effector position $O_3 = O_4$, but it does affect the orientation, as shown in Figure 2.4. The vector loop closure equation becomes:

$$\begin{aligned} {}^0\mathbf{R}_1\,{}^1\mathbf{b}_i &= \left({}^0\mathbf{t}_1 + {}^0\mathbf{t}_2\right) + {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i + {}^0\mathbf{l}_i \\ &= {}^0\mathbf{E}\mathbf{E}_{pos} + {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i + {}^0\mathbf{l}_i \end{aligned} \tag{2.14}$$

In case 2 ${}^0\mathbf{R}_1$ is an identity matrix so that ${}^1\mathbf{b}_i = {}^0\mathbf{b}_i$. The length of the i^{th} wire:

$$\|{}^0\mathbf{l}_i\| = \|{}^0\mathbf{b}_i - {}^0\mathbf{E}\mathbf{E}_{pos} - {}^0\mathbf{R}_3\,{}^3\mathbf{a}_i\| \tag{2.15}$$

Here ${}^0\mathbf{R}_3$ is unknown. Again, we use the product of exponentials formula to represent the rotation matrix:

$${}^0\mathbf{R}_3 = e^{\alpha_1\,\hat{\mathbf{z}}_0^{\wedge}} e^{\theta_1\,\hat{\mathbf{x}}_1^{\wedge}} e^{\theta_2\,\hat{\mathbf{y}}_2^{\wedge}} \tag{2.16}$$

Since $\alpha_1 = 0$ in case 2, the expression for ${}^0\mathbf{R}_3$ becomes:

$$ {}^0\mathbf{R}_3 = e^{\theta_1 \hat{\mathbf{x}}_1^\wedge} e^{\theta_2 \hat{\mathbf{y}}_2^\wedge} = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 \\ \sin\theta_1 \sin\theta_2 & \cos\theta_1 & -\cos\theta_2 \sin\theta_1 \\ -\cos\theta_1 \sin\theta_2 & \sin\theta_1 & \cos\theta_1 \cos\theta_2 \end{bmatrix} \tag{2.17} $$

As mentioned above, ${}^0\hat{\mathbf{z}}_3$ and ${}^0\hat{\mathbf{z}}_4$ remain the same, the last column of 2.17 that represents ${}^0\hat{\mathbf{z}}_3 = [{}^0z_{3x}, {}^0z_{3y}, {}^0z_{3z}]$ is equal to ${}^0\hat{\mathbf{z}}_4$ which is already known. Then we have three equations:

$$ \begin{bmatrix} {}^0z_{3x} \\ {}^0z_{3y} \\ {}^0z_{3z} \end{bmatrix} = \begin{bmatrix} \sin\theta_2 \\ -\cos\theta_2 \sin\theta_1 \\ \cos\theta_1 \cos\theta_2 \end{bmatrix} \tag{2.18} $$

From the first equation we can list possible solutions for θ_2:

$$ \theta_2 = \begin{cases} \text{Atan2}\left({}^0z_{3x}, \sqrt{1 - {}^0z_{3x}^2}\right) + 2k\pi \ (k = 0, \pm1, \pm2, ...) \\ \text{Atan2}\left({}^0z_{3x}, -\sqrt{1 - {}^0z_{3x}^2}\right) + 2k\pi \ (k = 0, \pm1, \pm2, ...) \end{cases} \tag{2.19} $$

Again, considering the joint limit for θ_2, only one solution of θ_2 is valid. Substitute θ_2 into equations of ${}^0z_{3y}$ and ${}^0z_{3z}$ and we can get a unique value for θ_1.

$$ \theta_1 = \text{Atan2}\left(\frac{{}^0z_{3y}}{-\cos\theta_2}, \frac{{}^0z_{3z}}{\cos\theta_2}\right) \tag{2.20} $$

Substitute θ_1 and θ_2 into 2.17 and 2.15, and we can get the wire lengths. So far we have already known ${}^0\mathbf{R}_4$ and ${}^0\mathbf{R}_3$, and we can compute the rotation angle α_2 using:

$$ {}^0\mathbf{R}_4 = \begin{bmatrix} c\alpha_2 c\theta_2 & -s\alpha_2 c\theta_2 & s\theta_2 \\ s\alpha_2 c\theta_1 + c\alpha_2 s\theta_1 s\theta_2 & c\alpha_2 c\theta_1 - s\alpha_2 s\theta_1 s\theta_2 & -c\theta_2 s\theta_1 \\ s\alpha_2 s\theta_1 - c\alpha_2 c\theta_1 s\theta_2 & c\alpha_2 s\theta_1 + s\alpha_2 c\theta_1 s\theta_2 & c\theta_1 c\theta_2 \end{bmatrix} \tag{2.21} $$

Since θ_1 and θ_2 are both known, we can easily get the value for α_2:

$$\alpha_2 = \text{Atan2}\left(\frac{{}^0x_{4x}}{\cos\theta_2}, -\frac{{}^0y_{4x}}{\cos\theta_2}\right) + 2k\pi \ (k = 0, \pm1, \pm2, ...) \tag{2.22}$$

2.2.2 Direct kinematics

For direct kinematics, the inputs are wire lengths and the 3^{rd} rotation angle α_1 or α_2, and the question is to calculate end effector's position and orientation. Unlike previous analysis, in this section we will analyze the 3-wire mechanism and 4-wire mechanism separately because the equations can be quite different.

2.2.2.1 Direct kinematics for 4-wire mechanism in case 1

The task for direct kinematics is to find the transformation matrix ${}^0\mathbf{T}_3$. Using the product of exponentials formula to express ${}^0\mathbf{T}_3$ gives:

$$\begin{aligned} {}^0\mathbf{T}_3 &= \mathbf{g}_{st}(\alpha_1, \theta_1, \theta_2) = e^{\xi_3^\wedge \alpha_1} e^{\xi_1^\wedge \theta_1} e^{\xi_2^\wedge \theta_2} \mathbf{g}_{st}(0) \\ &= \begin{bmatrix} c\alpha_1 c\theta_2 - s\alpha_1 s\theta_1 s\theta_2 & -s\alpha_1 c\theta_1 & c\alpha_1 s\theta_2 + s\alpha_1 c\theta_2 s\theta_1 & hc\alpha_1 s\theta_2 + hs\alpha_1 c\theta_2 s\theta_1 \\ s\alpha_1 c\theta_2 + c\alpha_1 s\theta_1 s\theta_2 & c\alpha_1 c\theta_1 & s\alpha_1 s\theta_2 - c\alpha_1 c\theta_2 s\theta_1 & hs\alpha_1 s\theta_2 - hc\alpha_1 c\theta_2 s\theta_1 \\ -c\theta_1 s\theta_2 & s\theta_1 & c\theta_1 c\theta_2 & h + hc\theta_1 c\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2.23}$$

$e^{\xi_3^\wedge \alpha_1}$, $e^{\xi_1^\wedge \theta_1}$ and $e^{\xi_2^\wedge \theta_2}$ represent homogeneous transformations from frame $\{1\}$ to base, from frame $\{2\}$ to $\{1\}$ and from frame $\{3\}$ to $\{2\}$ respectively, and $\mathbf{g}_{st}(0)$ is the transformation from frame $\{0\}$ to $\{3\}$ in initial condition. Here $\xi_i^\wedge (i = 1, 2, 3)$ is the wedge form of the twist ξ_i, where:

$$\xi_i^\wedge = \begin{bmatrix} \omega_i^\wedge & \mathbf{v}_i \\ 0 & 0 \end{bmatrix}, \quad \xi_i = \begin{bmatrix} \mathbf{v}_i \\ \omega_i \end{bmatrix} \tag{2.24}$$

ω_i refers to the rotation axis i. $\mathbf{v}_i = -\omega_i \times \mathbf{q}_i$ ($\mathbf{q}_i$ is a vector pointing from the origin of frame $\{0\}$ to an arbitrary point on rotation axis i) when it is a rotation joint, and $\mathbf{v}_i$ is the direction of translation when it is a translation joint.

In the following analysis, the angles θ_1 and θ_2 are unknown. Equation 2.1 can be rewritten as:

$$ {}^1\mathbf{b}_i = {}^1\mathbf{t}_1 + {}^1\mathbf{R}_3\,{}^3\mathbf{t}_2 + {}^1\mathbf{R}_3\,{}^3\mathbf{a}_i + {}^1\mathbf{l}_i \tag{2.25} $$

Here ${}^1\mathbf{b}_i$, ${}^1\mathbf{t}_1$, ${}^3\mathbf{t}_2$ and ${}^3\mathbf{a}_i$ are fixed vectors, and ${}^1\mathbf{R}_3$ has two variables θ_1 and θ_2. When the lengths of wires are given, we can write 4 equations for $i = 1,2,3,4$:

$$ \| {}^1\mathbf{l}_i \| = \| \left({}^1\mathbf{b}_i - {}^1\mathbf{t}_1 \right) - {}^1\mathbf{R}_3 \left({}^3\mathbf{t}_2 + {}^3\mathbf{a}_i \right) \| \tag{2.26} $$

Assume $\| {}^0\mathbf{l}_i \| = l_i$, $\| {}^1\mathbf{b}_i \| = r_1$, $\| {}^3\mathbf{a}_i \| = r_2$, and $\| {}^1\mathbf{t}_1 \| = \| {}^3\mathbf{t}_2 \| = h$, and number the four equations in 2.26 as eq_1, eq_2, eq_3 and eq_4.

$$ \begin{aligned} eq_1 &= -f_1 \cos\theta_2 - l_1^2 - f_2 + f_4 \\ eq_2 &= -f_1 \cos\theta_1 - l_2^2 + f_3 + f_4 \\ eq_3 &= -f_1 \cos\theta_2 - l_3^2 + f_2 + f_4 \\ eq_4 &= -f_1 \cos\theta_1 - l_4^2 - f_3 + f_4 \end{aligned} \tag{2.27} $$

where

$$ \begin{aligned} f_1 &= 2r_1 r_2 \\ f_2 &= 2hr_1 \sin\theta_2 + 2hr_2 \cos\theta_1 \sin\theta_2 \\ f_3 &= 2hr_2 \sin\theta_1 + 2hr_1 \cos\theta_2 \sin\theta_1 \\ f_4 &= 2h^2 + r_1^2 + r_2^2 + 2h^2 \cos\theta_1 \cos\theta_2 \end{aligned} \tag{2.28} $$

Calculating $eq_1 + eq_3 - eq_2 - eq_4$ provides:

$$ \cos\theta_1 = \cos\theta_2 + \frac{l_1^2 + l_3^2 - l_2^2 - l_4^2}{4r_1 r_2} \tag{2.29} $$

Moreover, calculating $eq_1 - eq_3$ yields:

$$-l_1^2 + l_3^2 - 4hr_1 \sin\theta_2 - 4hr_2 \cos\theta_1 \sin\theta_2 = 0 \tag{2.30}$$

Substituting Equation 2.29 into 2.30:

$$\cos\theta_2 = \frac{u}{\sin\theta_2} + v \tag{2.31}$$

where

$$\begin{aligned} u &= \frac{l_3^2 - l_1^2}{4hr_2} \\ v &= \frac{l_2^2 + l_4^2 - l_1^2 - l_3^2}{4r_1 r_2} - \frac{r_1}{r_2} \end{aligned} \tag{2.32}$$

Then substituting Equation 2.31 into $\sin^2\theta_2 + \cos^2\theta_2 = 1$ gives us a fourth order polynomial of $\sin\theta_2$:

$$k_0 \sin^4\theta_2 + k_1 \sin^3\theta_2 + k_2 \sin^2\theta_2 + k_3 \sin\theta_2 + k_4 = 0 \tag{2.33}$$

where $k_i,\ i = 0, 1, ...4$ are functions of $h, r_1, r_2, l_1, l_2, l_3$ and l_4:

$$\begin{aligned} k_0 =& 16h^2 r_1^2 r_2^2 \\ k_1 =& 0 \\ k_2 =& h^2 l_1^4 - 2h^2 l_1^2 l_2^2 + 2h^2 l_1^2 l_3^2 - 2h^2 l_1^2 l_4^2 + 8h^2 l_1^2 r_1^2 + h^2 l_2^4 - 2h^2 l_2^2 l_3^2 + 2h^2 l_2^2 l_4^2 \\ & - 8h^2 l_2^2 r_1^2 + h^2 l_3^4 - 2h^2 l_3^2 l_4^2 + 8h^2 l_3^2 r_1^2 + h^2 l_4^4 - 8h^2 l_4^2 r_1^2 + 16h^2 r_1^4 - 16h^2 r_1^2 r_2^2 \\ k_3 =& 2hl_1^4 r_1 - 2hl_1^2 l_2^2 r_1 - 2hl_1^2 l_4^2 r_1 + 8hl_1^2 r_1^3 + 2hl_2^2 l_3^2 r_1 - 2hl_3^4 r_1 + 2hl_3^2 l_4^2 r_1 - 8hl_3^2 r_1^3 \\ k_4 =& l_1^4 r_1^2 - 2l_1^2 l_3^2 r_1^2 + l_3^4 r_1^2 \end{aligned} \tag{2.34}$$

Equation 2.33 has at most 4 solutions for $\sin\theta_2$. For each $\sin\theta_2$, there should have been 2

sets of solutions for θ_2.

$$\theta_2 = \begin{cases} \text{Atan2}(\sin\theta_2, \sqrt{1-\sin^2\theta_2}) + 2k\pi \ (k = 0, \pm1, \pm2, ...) \\ \text{Atan2}(\sin\theta_2, -\sqrt{1-\sin^2\theta_2}) + 2k\pi \ (k = 0, \pm1, \pm2, ...) \end{cases} \tag{2.35}$$

However, because the range of θ_1 and θ_2 is within $[-\frac{\pi}{2}, \frac{\pi}{2}]$, only 1 solution is valid. Thus θ_2 has at most 4 solutions. Further, we can use Equation 2.29 and $eq2$ (in 2.36) to solve for $\cos\theta_1$ and $\sin\theta_1$ respectively, and finally, get a unique value for θ_1:

$$2h^2 + r_1^2 + r_2^2 - l_2^2 - 2\cos\theta_1 r_1 r_2 + 2hr_2\sin\theta_1 + 2\cos\theta_1\cos\theta_2 h^2 + 2\cos\theta_2\sin\theta_1 hr_1 = 0 \tag{2.36}$$

$$\theta_1 = \text{Atan2}(\sin\theta_1, \cos\theta_1) \tag{2.37}$$

Thus the direct kinematics problem has at most 4 solutions in total.

2.2.2.2 Direct kinematics for 4-wire mechanism in case 2

The task for direct kinematics in case 2 is to find the transformation matrix ${}^0\mathbf{T}_4$. Let's use the product of exponentials formula to express ${}^0\mathbf{T}_4$:

$$\begin{aligned} {}^0\mathbf{T}_4 &= \mathbf{g}_{st}(\theta_1, \theta_2, \alpha_2) = e^{\xi_1^\wedge \theta_1} e^{\xi_2^\wedge \theta_2} e^{\xi_3^\wedge \alpha_2}\, \mathbf{g}_{st}(0) \\ &= \begin{bmatrix} c\alpha_2 c\theta_2 & -s\alpha_2 c\theta_2 & s\theta_2 & hs\theta_2 \\ s\alpha_2 c\theta_1 + c\alpha_2 s\theta_1 s\theta_2 & c\alpha_2 c\theta_1 - s\alpha_2 s\theta_1 s\theta_2 & -c\theta_2 s\theta_1 & -hc\theta_2 s\theta_1 \\ s\alpha_2 s\theta_1 - c\alpha_2 c\theta_1 s\theta_2 & c\alpha_2 s\theta_1 + s\alpha_2 c\theta_1 s\theta_2 & c\theta_1 c\theta_2 & h + hc\theta_1 c\theta_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{2.38}$$

α_2 is known. we need to figure out the values for θ_1 and θ_2 to solve the direct kinematics problem. Equation 2.1 can be rewritten as:

$${}^0\mathbf{R}_1\, {}^1\mathbf{b}_i = {}^0\mathbf{R}_1\, {}^1\mathbf{t}_1 + {}^0\mathbf{R}_3\, {}^3\mathbf{t}_2 + {}^0\mathbf{R}_3\, {}^3\mathbf{a}_i + {}^0\mathbf{l}_i \tag{2.39}$$

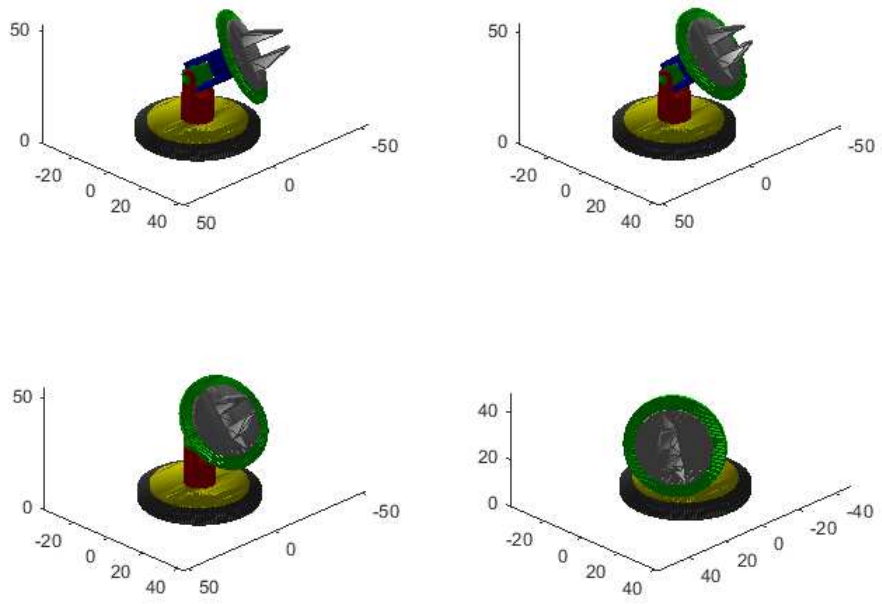

Figure 2.5: Four solutions for direct kinematics problem of 4-wire mechanism in case 1

where ${}^0\mathbf{R}_1$ is an identity matrix, ${}^1\mathbf{b}_i$, ${}^1\mathbf{t}_1$, ${}^3\mathbf{t}_2$, ${}^3\mathbf{a}_i$ are fixed vectors, and ${}^0\mathbf{R}_3$ has 2 variables θ_1 and θ_2. Since the lengths of wires are given, we can use it to list 4 equations for $i = 1,2,3,4$:

$$\| {}^0\mathbf{l}_i \| = \| \left({}^1\mathbf{b}_i - {}^1\mathbf{t}_1 \right) - {}^0\mathbf{R}_3 \left({}^3\mathbf{t}_2 + {}^3\mathbf{a}_i \right) \| \tag{2.40}$$

We can find that Equation 2.40 has the same expression in case 1, so does the procedures of solving equations for θ_1 and θ_2. After solving these equations we substitute θ_1 and θ_2 into Equation 2.38 to compute ${}^0\mathbf{T}_4$. Here we are going to use an example to demonstrate that the direct kinematics problem of 4-wire wrist may have 4 solutions. Suppose that $l_1 = 33.85mm$, $l_2 = 20.07mm$, $l_3 = 35.57mm$, $l_4 = 49.34mm$, and $\alpha_1 = 0$ in case 1 and $\alpha_2 = 0$ in case 2. The results are shown in Figure 2.5 and Figure 2.6. The end effector positions for these four poses in case 1 are $[-10.73, 13.10, 27.62]mm$ with $\theta_1 = -56.67°$, $\theta_2 = -34.38°$, $[-5.95, 13.85, 30.57]mm$ with $\theta_1 = -50.14°$, $\theta_2 = -18.24°$,

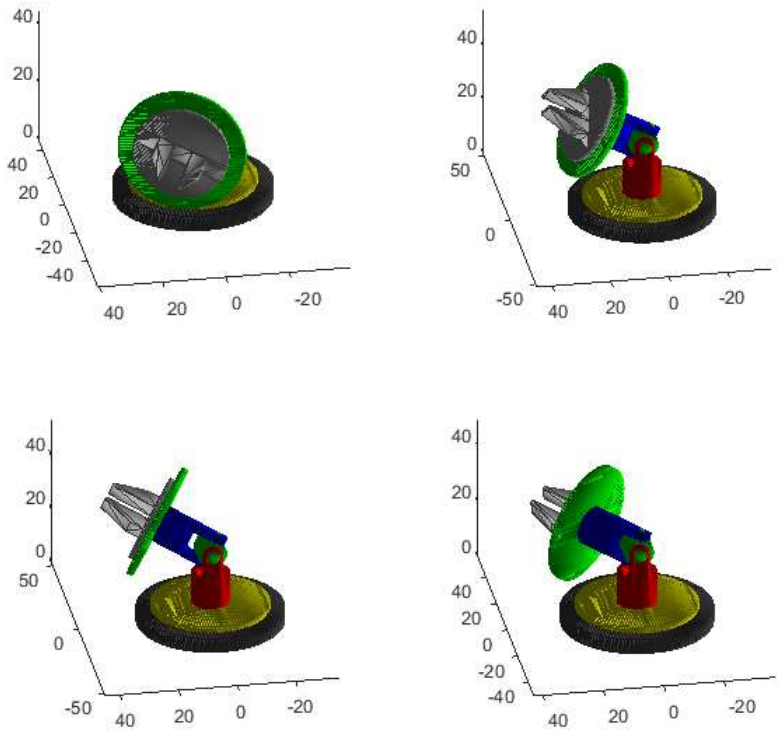

Figure 2.6: Four solutions for direct kinematics problem of 4-wire mechanism in case 2

$[0.9951, 14.10, 31.70]mm$ with $\theta_1 = -48°$, $\theta_2 = 3°$ and $[-15.68, 10.34, 22.86]mm$ with $\theta_1 = -74.56°$, $\theta_2 = 55.63°$.

The end effector positions for these four poses in case 2 are $[-17.88, 6.39, 18.25]mm$, $[-1.99, 16.36, 28.45]mm$, $[7.04, 15.81, 26.84]mm$ and $[12.83, 13.57, 22.51]mm$.

2.2.2.3 Direct kinematics for 3-wire mechanism in case 1

In this subsection, Equations 2.23 and 2.26 remain the same except for $i = 1, 2, 3$ instead of $i = 1, 2, 3, 4$. Since in case 1 α_1 is already known, we will use 2.26 to calculate θ_1 and θ_2. Again, let's assume that $\| {}^0\mathbf{l}_i \| = l_i$, $\| {}^1\mathbf{b}_i \| = r_1$, $\| {}^3\mathbf{a}_i \| = r_2$, and $\| {}^1\mathbf{t}_1 \| = \| {}^3\mathbf{t}_2 \| = h$, and number the three equations in 2.26 as eq_1, eq_2 and eq_3. The difficulty for this subsection is solving the three polynomial equations. We will use resultants method to find solutions for θ_1 and θ_2. First, eq_1, eq_2 and eq_3 can be expressed as:

$$
\begin{aligned}
eq_1 =& F_1 + 2h^2 c\theta_1 c\theta_2 - 2hr_2 c\theta_1 s\theta_2 - l_1^2 - 2r_1 r_2 c\theta_2 - 2r_1 hs\theta_2 \\
eq_2 =& F_1 + F_2 + F_3 - l_2^2 \\
eq_3 =& F_1 + F_2 - F_3 - l_3^2
\end{aligned} \tag{2.41}
$$

where:

$$
\begin{aligned}
F_1 =& 2h^2 + r_1^2 + r_2^2 \\
F_2 =& -\frac{3r_1 r_2}{2}\cos\theta_1 - \frac{r_1 r_2}{2}\cos\theta_2 + hr_1 \sin\theta_2 + 2h^2 \cos\theta_1 \cos\theta_2 + hr_2 \cos\theta_1 \sin\theta_2 \\
F_3 =& \sqrt{3}hr_2 \sin\theta_1 + \sqrt{3}hr_1 \cos\theta_2 \sin\theta_1 + \frac{\sqrt{3}}{2} r_1 r_2 \sin\theta_1 \sin\theta_2
\end{aligned} \tag{2.42}
$$

In eq_1 we use $\sqrt{1-\sin^2\theta_1}$ to replace $\cos\theta_1$. The reason why we do not replace $\cos\theta_1$ with $\pm\sqrt{1-\sin^2\theta_1}$ is that the joint limit for θ_1 is $[-\frac{\pi}{2}, \frac{\pi}{2}]$, and in this range of θ_1, $\cos\theta_1$ has non-negative values. Thus eq1 becomes:

$$
(2h^2 c\theta_2 - 2hr_2 s\theta_2)\sqrt{1-s^2\theta_1} = l_1^2 - r_1^2 - r_2^2 - 2h^2 + 2r_1 r_2 c\theta_2 + 2r_1 hs\theta_2 \tag{2.43}
$$

Further, we square on both sides of 2.43 and get:

$$
(2h^2 c\theta_2 - 2hr_2 s\theta_2)^2 (1-s^2\theta_1) = (l_1^2 - r_1^2 - r_2^2 - 2h^2 + 2r_1 r_2 c\theta_2 + 2r_1 hs\theta_2)^2 \tag{2.44}
$$

Moreover we can write 2.44 as a second order polynomial:

$$
f_0(\theta_2)x^2 + f_1(\theta_2)x + f_2(\theta_2) = 0 \tag{2.45}
$$

where

$$
\begin{aligned}
&x = \sin\theta_1 \\
&f_0(\theta_2) = -(2h^2 c\theta_2 - 2hr_2 s\theta_2)^2 \\
&f_1(\theta_2) = 0 \\
&f_2(\theta_2) = (2h^2 c\theta_2 - 2hr_2 s\theta_2)^2 - (l_1^2 - r_2^2 - r_1^2 - 2h^2 + 2r_1 r_2 c\theta_2 + 2r_1 h s\theta_2)^2
\end{aligned}
\tag{2.46}
$$

In 2.46, the parameters f_2 and f_0 are both functions of θ_2. We will do the same thing on $(eq_2 - eq_3)$, which can be expressed as:

$$
-l_2^2 + l_3^2 + 2\sqrt{3}hr_2 s\theta_1 + 2\sqrt{3}hr_1 c\theta_2 s\theta_1 + \sqrt{3}r_1 r_2 s\theta_1 s\theta_2 = 0 \tag{2.47}
$$

and further we write it as a second order polynomial:

$$
g_0(\theta_2)x^2 + g_1(\theta_2)x + g_2(\theta_2) = 0 \tag{2.48}
$$

where

$$
\begin{aligned}
&x = \sin\theta_1 \\
&g_0(\theta_2) = 12h^2 r_1^2(1 - s^2\theta_2) - (\sqrt{3}r_1 r_2 s\theta_2 + 2\sqrt{3}hr_2)^2 \\
&g_1(\theta_2) = (\sqrt{3}r_1 r_2 s\theta_2 + 2\sqrt{3}hr_2)(l2^2 - l3^2) \\
&g_2(\theta_2) = -(l_2^2 - l_3^2)^2
\end{aligned}
\tag{2.49}
$$

Multiply x on both sides of 2.45 and 2.48, and we will get two more equations:

$$
f_0(\theta_2)x^3 + f_1(\theta_2)x^2 + f_2(\theta_2)x = 0 \tag{2.50}
$$

$$
g_0(\theta_2)x^3 + g_1(\theta_2)x^2 + g_2(\theta_2)x = 0 \tag{2.51}
$$

Equations 2.45, 2.48, 2.50 and 2.51 form a homogeneous linear system:

$$\underbrace{\begin{bmatrix} 0 & f_0(\theta_2) & 0 & f_2(\theta_2) \\ 0 & g_0(\theta_2) & g_1(\theta_2) & g_2(\theta_2) \\ f_0(\theta_2) & 0 & f_2(\theta_2) & 0 \\ g_0(\theta_2) & g_1(\theta_2) & g_2(\theta_2) & 0 \end{bmatrix}}_{D} \begin{bmatrix} x^3 \\ x^2 \\ x \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.52}$$

The necessary and sufficient condition for a non-trivial solution of Equation 2.52 is $det(D) = 0$. Using this condition we can get an $8th$ order polynomial of θ_2:

$$k_0 \sin^8 \theta_2 + k_1 \sin^7 \theta_2 + ... + k_7 \sin \theta_2 + k_8 = 0 \tag{2.53}$$

Here $k_i, i = 0,1,2,...,8$ are functions of constants h, r_1, r_2, l_1, l_2 and l_3. 2.53 has at most 8 solutions for $\sin \theta_2$. For each $\sin \theta_2$, only θ_2 within $[-\frac{\pi}{2}, \frac{\pi}{2}]$ is valid. Then substitute θ_2 into 2.47 and get the unique solution for θ_1.

2.2.2.4 Direct kinematics for 3-wire mechanism in case 2

In this subsection, The 3-wire mechanism's transformation matrix has the same expression with 2.38 containing variables of θ_1, θ_2 and α_2, and 2.40 remain the same except for $i = 1,2,3$ instead of $i = 1,2,3,4$. After listing the 3 equations we found that the expressions for eq_1 eq_2 and eq_3 are exact equations in 2.41. Thus, we can use the same procedures to solve the direct kinematics problems of case 2 as we have done in case 1.

Here we will also demonstrate that the direct kinematic problem of 3-wire mechanism may have 8 solutions for both cases. Supposing $l_1 = 29.68mm$, $l_2 = 41.7mm$, $l_3 = 40.10mm$ and $\alpha_1 = \frac{\pi}{2}$ for case 1 and $\alpha_2 = \frac{\pi}{2}$ for case 2. The results are shown in Figure 2.7 and Figure 2.8.

The end effector positions for these eight poses in case 1 are $[0.82, -17.71, 25.83]mm$,

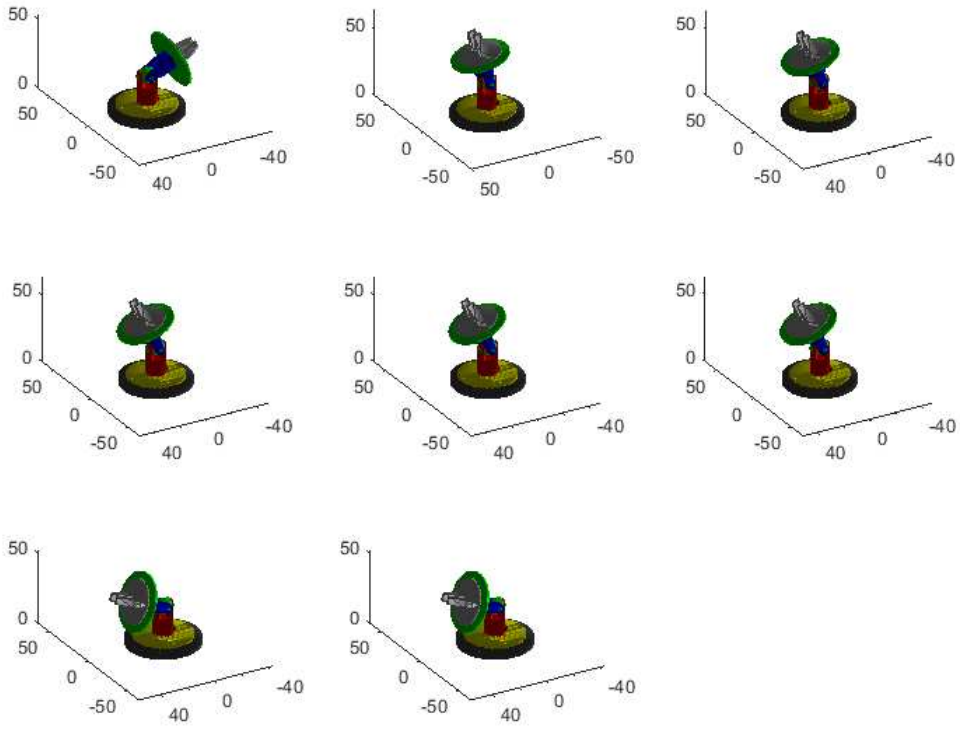

Figure 2.7: Eight solutions for direct kinematics problem of 3-wire mechanism in case 1

$[0.98, 4.22, 37.50]\,mm$, $[0.95, 5.84, 37.06]\,mm$, $[0.92, 7.15, 36.58]\,mm$, $[0.92, 7.18, 36.57]\,mm$, $[0.91, 7.73, 36.33]\,mm$, $[0.42, 17.75, 25.76]\,mm$, and $[0.41, 17.76, 25.75]\,mm$.

2.2.3 Validation of Inverse and Direct Kinematics

In this section, we will validate the models of inverse and direct kinematics by assigning values for variables:

$$h = 19mm,\ r1 = 18mm,\ r2 = 18mm \tag{2.54}$$

Moreover, values for θ_1 and θ_2 are selected from $-\frac{\pi}{3}$ to $\frac{\pi}{3}$ evenly with certain step of $\frac{\pi}{10}$, while $\alpha_1 = \alpha_2 = \frac{\pi}{2}$. In the following validations, we should have 8 situations to discuss: inverse kinematics of 4 wires/3 wires in case 1/case 2 and direct kinematics of 4 wires/3 wires in case 1/case 2. However, the validation procedures of 3-wire mechanism is almost the same with the 4-wire's. Thus we will only give detailed analysis for 4-wire wrist but

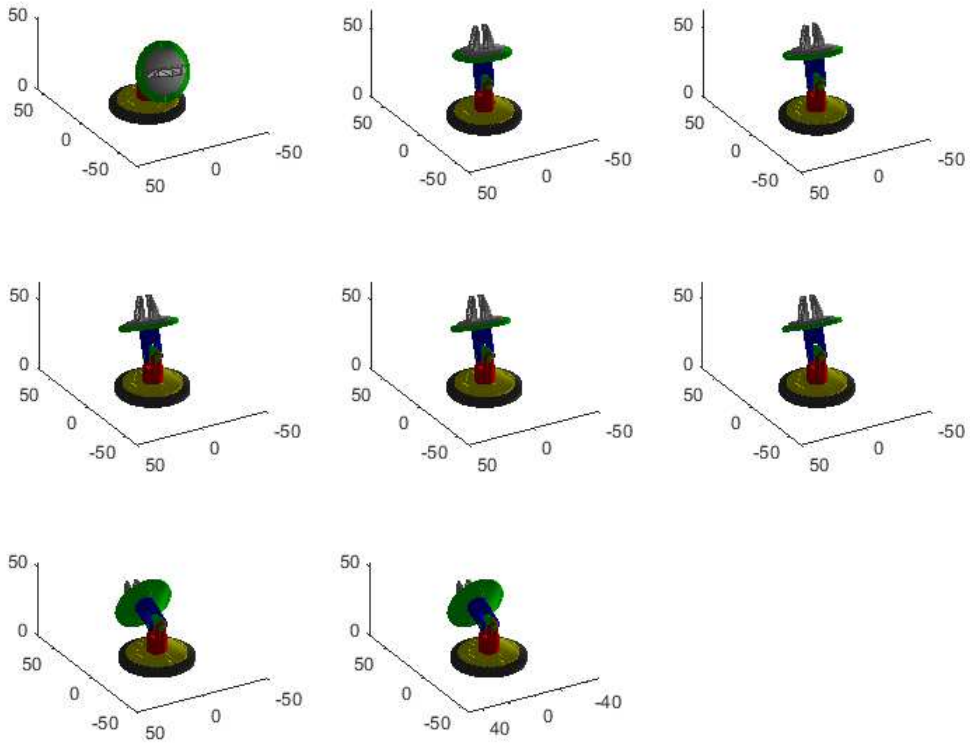

Figure 2.8: Eight solutions for direct kinematics problem of 3-wire mechanism in case 2

will show final results for both 3-wire and 4-wire's.

2.2.3.1 Validation of Inverse Kinematics in case 1

For inverse kinematics, end effector's position ${}^0\mathbf{EE}_{pos}$ and orientation ${}^0\mathbf{R}_3$ are given. We need to compute joint values using models in previous section, and compare those calculated values with the reference ones. The detailed procedures is shown in Figure 2.9.

First, select values for θ_1, $\theta_2 \in [-\frac{\pi}{3}, \frac{\pi}{3}]$, and $\alpha_1 = \frac{\pi}{2}$; then use them to calculate EE_{pos} and ${}^0\mathbf{R}_4$ which are regarded as kinematically consistent inputs. At the same time, compute wire lengths as reference for comparison, written as l_1, l_2, l_3 and l_4. Then use inverse kinematics model, EE_{pos} and ${}^0\mathbf{R}_4$ to compute $\tilde{\theta}_1, \tilde{\theta}_2$ and $\tilde{\alpha}_1$, as well as wire lengths $\tilde{l_1}, \tilde{l_2}, \tilde{l_3}$ and $\tilde{l_4}$; finally, compare the reference and the calculated joint values.

The result of the comparison can be shown in figure 2.10 and 2.11. From these two fig-

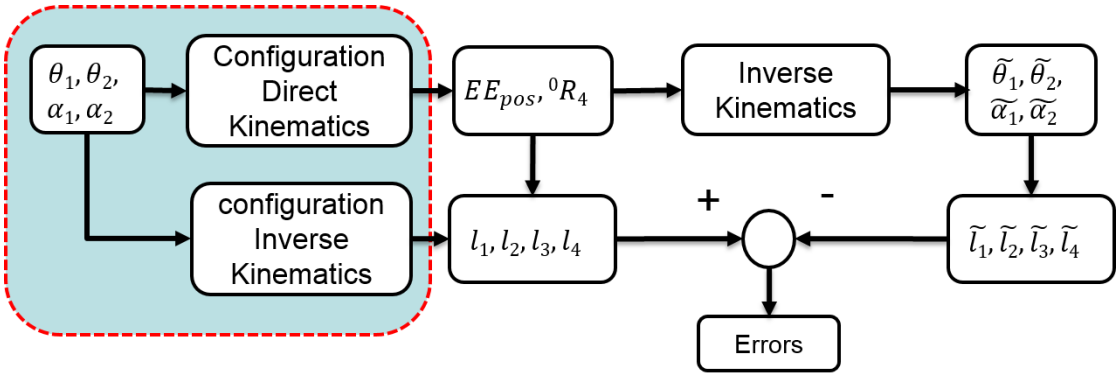

Figure 2.9: Procedures of inverse kinematics validation

ures where the reference values are shown using ”O” and the calculated values are using $*$, we can see that all the $*$ are almost coincident with ”O”, and the greatest error between the reference and calculated values is 1.4211×10^{-14}, which can demonstrate that the model for inverse kinematics is correct.

2.2.3.2 Validation of Inverse Kinematics in case 2

The procedures of validating inverse kinematics in case 2 is almost the same with case 1. The only alteration is that we select value for $\alpha_2 = \frac{\pi}{2}$ instead of α_1. The results are shown in Figure 2.12 and 2.13. Again we can see from the figures that the inverse kinematics model works well.

2.2.3.3 Validation of Direct Kinematics in case 1

For direct kinematics, the inputs are wire lengths and α_1 and we need to compute end effector's position EE_{pos}. The detailed procedures are shown in Figure 2.14.

Similar to inverse kinematics, we first select values for θ_1 and $\theta_2 \in [-\frac{\pi}{3}, \frac{\pi}{3}]$, $\alpha_1 = \frac{\pi}{2}$; then use θ_1, θ_2 and α_1 to calculate EE_{pos} as reference and compute wire lengths as inputs l_1, l_2, l_3 and l_4; what's more, use direct kinematics model, wire lengths and α_1 to obtain values of $\tilde{\theta}_1$ and $\tilde{\theta}_2$, and again get the calculated end effector position $\tilde{EE}_{pos}$; finally, compare those reference and calculated values of end effector's positions. Here we plot the results in

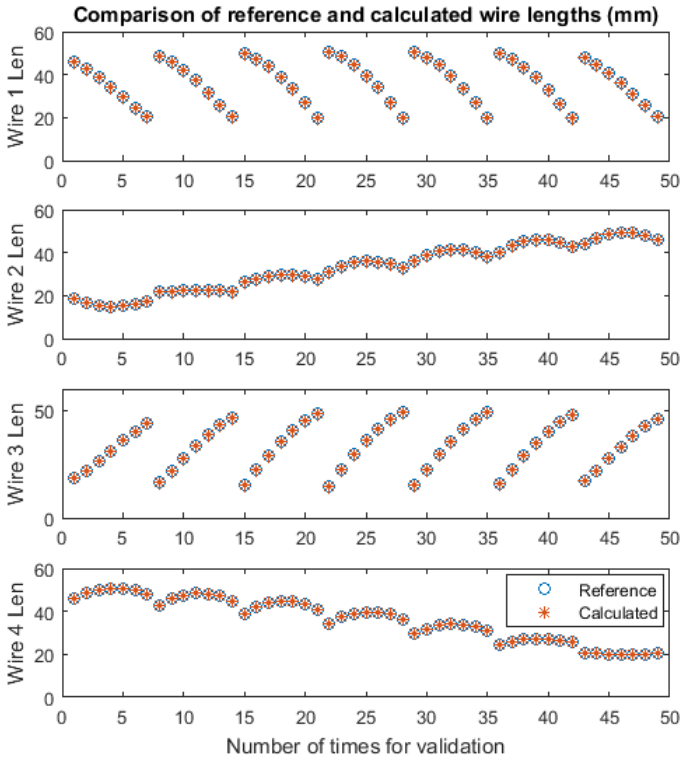

Figure 2.10: Inverse kinematics validation of 4 wires mechanism in case 1

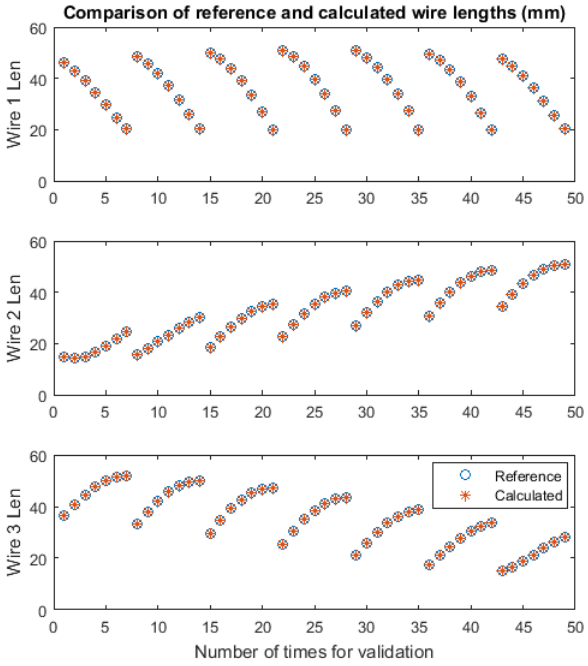

Figure 2.11: Inverse kinematics validation of 3 wires mechanism in case 1

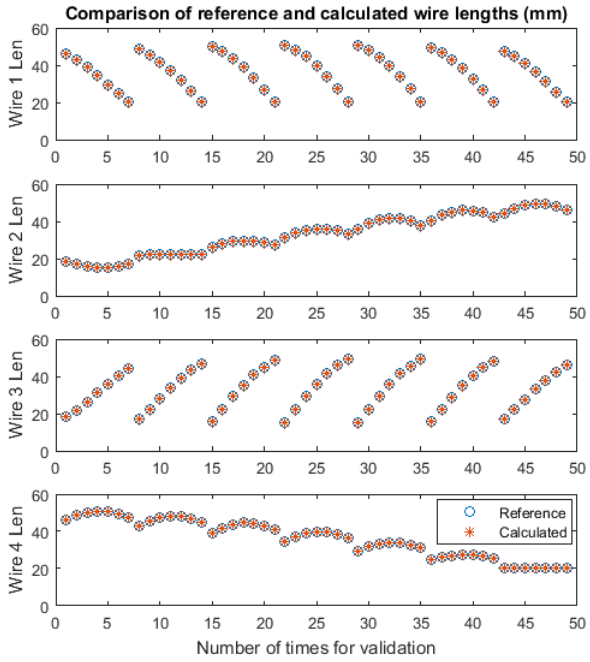

Figure 2.12: Inverse kinematics validation of 4 wires mechanism in case 2

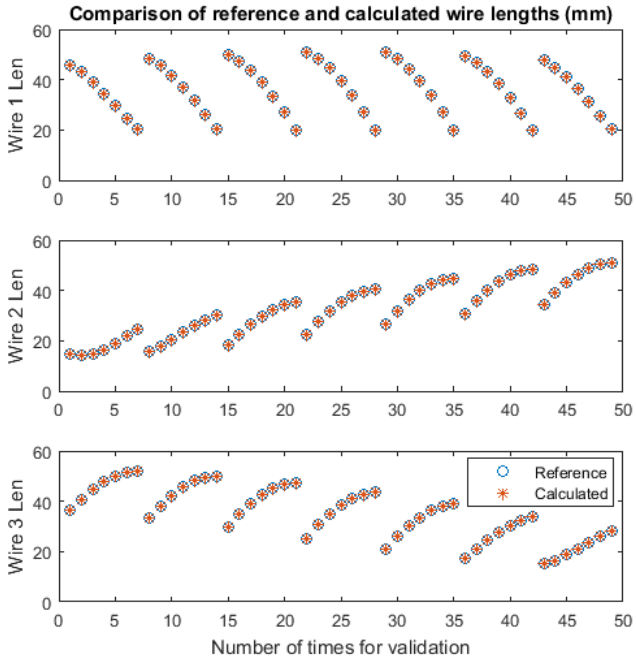

Figure 2.13: Inverse kinematics validation of 3 wires mechanism in case 2

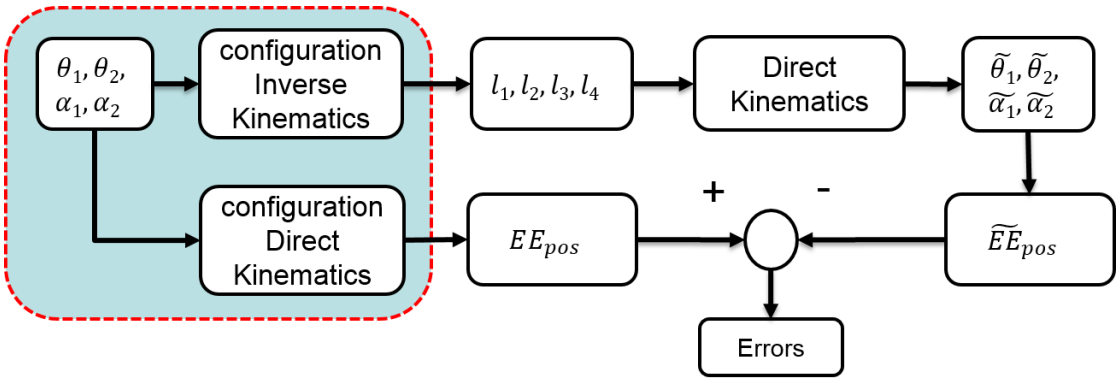

Figure 2.14: Procedures of direct kinematics validation

polar coordinate system where the radius represents the tilt angle, and the azimuth angle represents the direction of tilt. And we have the same definition for polar coordinate system in the following analysis.

Figure 2.15 and 2.16 show the end effector's position in polar coordinate where the radius represents the tilt angle. Here "O" is to indicate reference values and $*$ to indicate calculated values. We can find that all the "O" and $*$ are coincident which means the direct kinematics model in case 1 is correct.

2.2.3.4 Validation of Direct Kinematics in case 2

Based on the procedures in case 1, we replace $\alpha_1 = \frac{\pi}{2}$ with $\alpha_2 = \frac{\pi}{2}$ in case 2. Figure 2.17 and 2.18 showing the comparison results demonstrate that the direct kinematics model is correct.

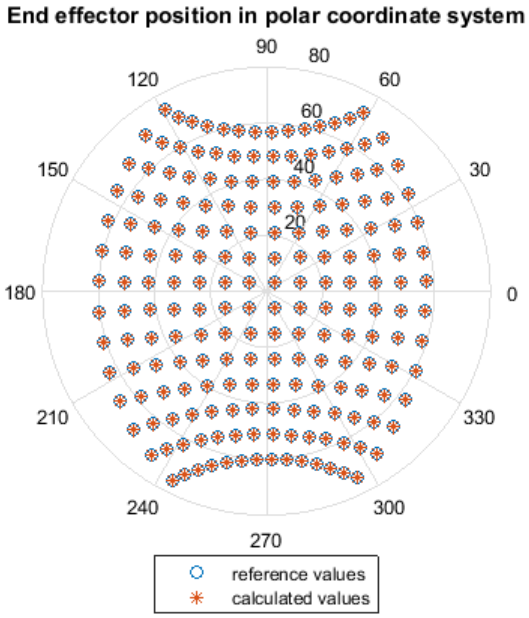

Figure 2.15: Direct kinematics validation of 4 wires mechanism in case 1 (Unit: degree)

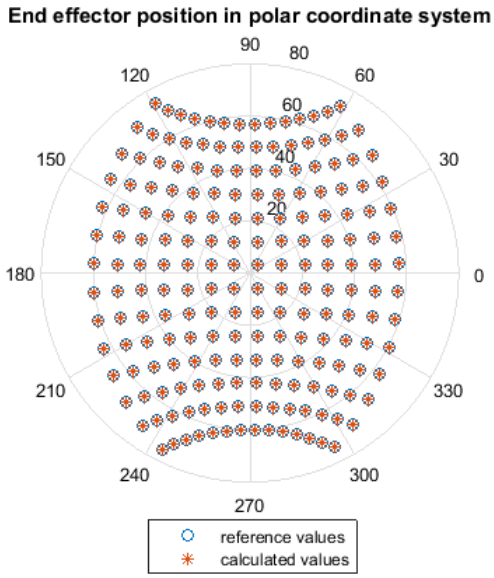

Figure 2.16: Direct Kinematics Validation of 3 wires mechanism in case 1 (Unit: degree)

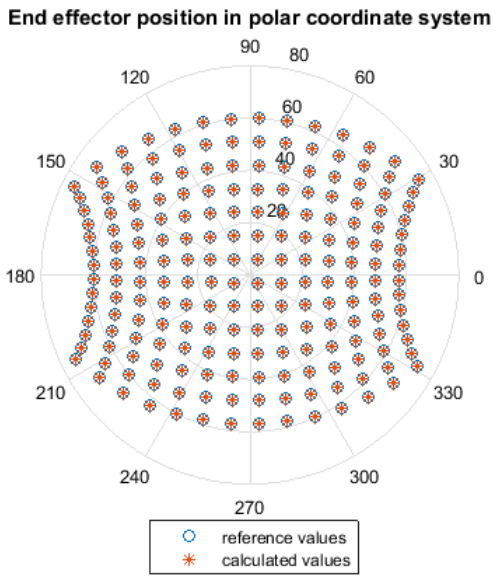

Figure 2.17: Direct Kinematics Validation of 4 wires mechanism in case 2 (Unit: degree)

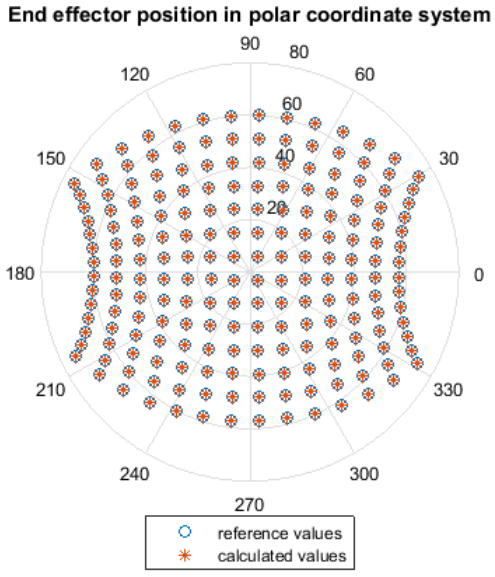

Figure 2.18: Direct Kinematics Validation of 3 wires mechanism in case 2 (Unit: degree)

Chapter 3

Instantaneous Kinematics

For a robot manipulator, the Jacobian matrix, or simply Jacobian is defined as the matrix that transforms the joint rates in the actuator space to end effector velocity [27]. In our project, it is difficult to calculate directly the complete Jacobian for the 3-DOF hybrid mechanism, 2-DOF universal joint with another rotational DOF either sitting on the top plate or lying under the bottom plate. We should notice that no matter where the 3^{rd} DOF is placed, the expression of "partial" Jacobian $\tilde{\mathbf{J}}$ (Here we use the tilde to indicate that it is a partial item) for the 2-DOF universal joint is unchanged, even though the "complete" Jacobian $\mathbf{J}$ expressions are different. Based on this, we split the computation into two parts. In the first part, we compute the "partial" Jacobian $\tilde{\mathbf{J}}$ and then in the second part, we add the "serial" part to it to form a complete Jacobian $\mathbf{J}$.

3.1 Partial Jacobian Calculation

In this section, we will use two methods, the virtual work method and loop closure kinematics method to calculate $\tilde{\mathbf{J}}$. In the first method, we use the virtual work principle that total work done by the applied forces of a mechanical system as it moves through a set of infinitesimal virtual displacements is zero, to acquire the relationship between external wrench and input actuations, namely the transpose of Jacobian. In the second method, we use the geometric relationship to deduce the relationship of joint velocity to end effector velocity, that is, the Jacobian. Since this 2-DOF universal mechanism has three or four inputs, $\tilde{\mathbf{J}}$ should be a 3×2 (3 actuation wires) or 4×2 (4 actuation wires) matrix. No matter which method we use, the result should be the same. Moreover, because there is no significant difference between 3-wire and 4-wire mechanism in terms of Jacobian analysis, we assume the wrist has 4 actuation wires by default.

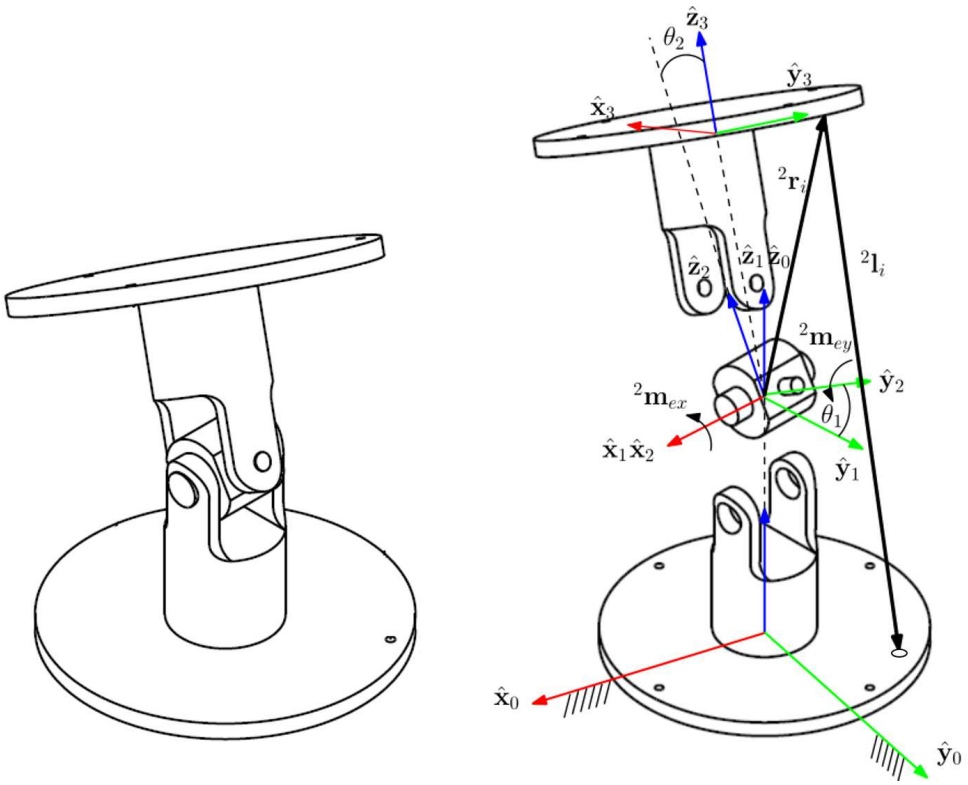

Figure 3.1: Nomenclature for the virtual work method

3.1.1 Partial Jacobian Derivation Using the Virtual Work Method

This method follows the approach of Hamid et al. [28]. First we need to define some vectors. We denote forces in the wires as τ_{wi} $(i = 1,2,3,4)$ and the external moment acting on the end effector as a 2-dimension vector ${}^2\tilde{\mathbf{w}}_{ext} = -{}^2\tilde{\mathbf{w}}_{ee} = [{}^2m_{ex}, {}^2m_{ey}]^T$, where ${}^2m_{ex}$ and ${}^2m_{ey}$ represent projections of ${}^2\tilde{\mathbf{w}}_{ext}$ along $\hat{\mathbf{x}}_2$ and $\hat{\mathbf{y}}_2$ respectively expressed in frame 2. According to the virtual work method, for any arbitrary infinitesimal change $\delta\theta_1$ and $\delta\theta_2$, the sum work of all wrench applied on the mechanism should be 0. Thus we have the following equation expressed in frame $\{2\}$:

$$0 = \left({}^2\tilde{\mathbf{w}}_{ext} + \sum_{i=1}^{4} {}^2\mathbf{r}_i \times \tau_{wi} \right)^T \left(\delta\theta_1 \, {}^2\hat{\mathbf{x}}_2 + \delta\theta_2 \, {}^2\hat{\mathbf{y}}_2 \right) \tag{3.1}$$

where ${}^2\mathbf{r}_i$ is the location of the i^{th} wire anchor point in frame $\{2\}$. Substitute ${}^2\tilde{\mathbf{w}}_{ext}$ using ${}^2m_{ex} + {}^2m_{ey}$ and we get:

$$
\begin{aligned}
0 = &\left[{}^2m_{ex},\ {}^2m_{ey}\right]^T \delta\theta_1\, {}^2\hat{\mathbf{x}}_2 + \left[{}^2m_{ex},\ {}^2m_{ey}\right]^T \delta\theta_2\, {}^2\hat{\mathbf{y}}_2 \\
&+ \left({}^2\mathbf{r}_1 \times {}^2\boldsymbol{\tau}_{w1}\right)^{\mathrm{T}} \left(\delta\theta_1\, {}^2\hat{\mathbf{x}}_2 + \delta\theta_2\, {}^2\hat{\mathbf{y}}_2\right) + \left({}^2\mathbf{r}_2 \times {}^2\boldsymbol{\tau}_{w2}\right)^{\mathrm{T}} \left(\delta\theta_1\, {}^2\hat{\mathbf{x}}_2 + \delta\theta_2\, {}^2\hat{\mathbf{y}}_2\right) \\
&+ \left({}^2\mathbf{r}_3 \times {}^2\boldsymbol{\tau}_{w3}\right)^{\mathrm{T}} \left(\delta\theta_1\, {}^2\hat{\mathbf{x}}_2 + \delta\theta_2\, {}^2\hat{\mathbf{y}}_2\right) + \left({}^2\mathbf{r}_4 \times {}^2\boldsymbol{\tau}_{w4}\right)^{\mathrm{T}} \left(\delta\theta_1\, {}^2\hat{\mathbf{x}}_2 + \delta\theta_2\, {}^2\hat{\mathbf{y}}_2\right)
\end{aligned}
\tag{3.2}
$$

Using the following definition of $\boldsymbol{\tau}_{wi}$ for the tension in the i^{th} wire:

$$
\boldsymbol{\tau}_{wi} = \tau_{wi}\, \hat{\mathbf{l}}_i, \quad i = 1,2,3,4 \tag{3.3}
$$

where $\hat{\mathbf{l}}_i$ is the unit vector of the i^{th} wire. Rewriting 3.2 in matrix matrix gives:

$$
\begin{gathered}
\underbrace{\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}}_{\boldsymbol{A}_{2\times 2}} \underbrace{\begin{bmatrix} {}^2m_{ex} \\ {}^2m_{ey} \end{bmatrix}}_{{}^2\tilde{\mathbf{w}}_{ee\ 2\times 1}} = \\
\underbrace{\begin{bmatrix} -\left({}^2\mathbf{r}_1 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{x}}_2 & -\left({}^2\mathbf{r}_2 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{x}}_2 & -\left({}^2\mathbf{r}_3 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{x}}_2 & -\left({}^2\mathbf{r}_4 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{x}}_2 \\ -\left({}^2\mathbf{r}_1 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{y}}_2 & -\left({}^2\mathbf{r}_2 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{y}}_2 & -\left({}^2\mathbf{r}_3 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{y}}_2 & -\left({}^2\mathbf{r}_4 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{y}}_2 \end{bmatrix}}_{\boldsymbol{B}_{2\times 4}} \underbrace{\begin{bmatrix} \tau_{w1} \\ \tau_{w2} \\ \tau_{w3} \\ \tau_{w4} \end{bmatrix}}_{\boldsymbol{\tau}_{w\ 4\times 1}}
\end{gathered}
\tag{3.4}
$$

We can see that Equation 3.4 provides us with the projection matrix from external moment to joint forces. Combined with the parallel robot Jacobian definition in statics, we have:

$$
\begin{aligned}
\mathbf{A}_{2\times 2}^{\mathrm{T}}\, \mathbf{B}_{2\times 4}\, \boldsymbol{\tau}_{w\ 4\times 1} &= {}^2\tilde{\mathbf{w}}_{ee\ 2\times 1} \\
{}^2\tilde{\mathbf{J}}^T\, \boldsymbol{\tau}_{w\ 4\times 1} &= {}^2\tilde{\mathbf{w}}_{ee\ 2\times 1}
\end{aligned}
\tag{3.5}
$$

where ${}^2\tilde{\mathbf{J}}$ is given by:

$$
{}^2\tilde{\mathbf{J}} \triangleq \left(\mathbf{B}^{\mathrm{T}}\right)_{4\times 2}\, \mathbf{A}_{2\times 2} \tag{3.6}
$$

Then we can get the expression of output hook's angular velocity.

$$ {}^{2}\tilde{\mathbf{J}}\,{}^{2}\tilde{\omega}_{3/1} = \dot{\mathbf{q}}_w \tag{3.7} $$

$$ {}^{2}\tilde{\omega}_{3/1} = {}^{2}\tilde{\mathbf{J}}^{\dagger}\,\dot{\mathbf{q}}_w \tag{3.8} $$

Here ${}^{2}\tilde{\mathbf{J}}^{\dagger}$ is the pseudo inverse of ${}^{2}\tilde{\mathbf{J}}$. $\dot{\mathbf{q}}_w = [\dot{l}_1, \dot{l}_2, \dot{l}_3, \dot{l}_4]$ represents a 4×1 vector representing wires' velocities where $\dot{l}_i$ is the velocity of wire joint, and ${}^{k}\tilde{\omega}_{m/n}$ means a 2×1 vector referring to the angular velocity of frame *m* relative to frame *n* expressed in frame *k*.

3.1.2 Partial Jacobian Derivation Using the Loop Closure Kinematics Method

The geometric relationship is shown in Figure 2.3. In Position Analysis, we have obtained Equation 2.1 and the following analysis will be based on this equation but expressed in frame 2:

$$ {}^{2}\mathbf{b}_i = {}^{2}\mathbf{t}_1 + {}^{2}\mathbf{t}_2 + {}^{2}\mathbf{a}_i + {}^{2}\mathbf{l}_i \;\; (i = 1,2,3,4) \tag{3.9} $$

$$ {}^{2}\mathbf{R}_1\,{}^{1}\mathbf{b}_i = {}^{2}\mathbf{R}_1\,{}^{1}\mathbf{t}_1 + {}^{2}\mathbf{R}_3\,{}^{3}\mathbf{t}_2 + {}^{2}\mathbf{R}_3\,{}^{3}\mathbf{a}_i + {}^{2}\mathbf{l}_i \tag{3.10} $$

Taking the derivatives on both sides of 3.10 results in:

$$ {}^{2}\boldsymbol{\omega}_1 \times {}^{2}\mathbf{b}_i = {}^{2}\boldsymbol{\omega}_1 \times {}^{2}\mathbf{t}_1 + {}^{2}\boldsymbol{\omega}_3 \times {}^{2}\mathbf{t}_2 + {}^{2}\boldsymbol{\omega}_3 \times {}^{2}\mathbf{a}_i + {}^{2}\hat{\mathbf{l}}_i \cdot \dot{l}_i \tag{3.11} $$

Here ${}^{2}\boldsymbol{\omega}_i$ is short for ${}^{2}\boldsymbol{\omega}_{i/2}$ referring to angular velocity of frame i relative to frame 2 expressed in frame 2. Then dot-multiply both sides of 3.11 by unit vector ${}^{2}\hat{\mathbf{l}}_i$.

$$ \left({}^{2}\mathbf{b}_i \times {}^{2}\hat{\mathbf{l}}_i\right)^{T} {}^{2}\boldsymbol{\omega}_1 = \left({}^{2}\mathbf{t}_1 \times {}^{2}\hat{\mathbf{l}}_i\right)^{T} {}^{2}\boldsymbol{\omega}_1 + \left({}^{2}\mathbf{t}_2 \times {}^{2}\hat{\mathbf{l}}_i\right)^{T} {}^{2}\boldsymbol{\omega}_3 + \left({}^{2}\mathbf{a}_i \times {}^{2}\hat{\mathbf{l}}_i\right)^{T} {}^{2}\boldsymbol{\omega}_3 + \dot{l}_i \tag{3.12} $$

Since ${}^{2}\boldsymbol{\omega}_1 = [-1;0;0] \cdot \dot{\theta}_1$ and ${}^{2}\boldsymbol{\omega}_3 = [0;1;0] \cdot \dot{\theta}_2$ We can rewrite 3.12 into matrix form:

$$
\begin{aligned}
&\underbrace{\begin{bmatrix}
\left({}^2\mathbf{b}_1 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{x}}_2 - \left({}^2\hat{\mathbf{t}}_1 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{x}}_2 & \left({}^2\mathbf{t}_2 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{y}}_2 + \left({}^2\mathbf{a}_1 \times {}^2\hat{\mathbf{l}}_1\right)^T {}^2\hat{\mathbf{y}}_2 \\
\left({}^2\mathbf{b}_2 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{x}}_2 - \left({}^2\hat{\mathbf{t}}_1 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{x}}_2 & \left({}^2\mathbf{t}_2 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{y}}_2 + \left({}^2\mathbf{a}_2 \times {}^2\hat{\mathbf{l}}_2\right)^T {}^2\hat{\mathbf{y}}_2 \\
\left({}^2\mathbf{b}_3 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{x}}_2 - \left({}^2\hat{\mathbf{t}}_1 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{x}}_2 & \left({}^2\mathbf{t}_2 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{y}}_2 + \left({}^2\mathbf{a}_3 \times {}^2\hat{\mathbf{l}}_3\right)^T {}^2\hat{\mathbf{y}}_2 \\
\left({}^2\mathbf{b}_4 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{x}}_2 - \left({}^2\hat{\mathbf{t}}_1 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{x}}_2 & \left({}^2\mathbf{t}_2 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{y}}_2 + \left({}^2\mathbf{a}_4 \times {}^2\hat{\mathbf{l}}_4\right)^T {}^2\hat{\mathbf{y}}_2
\end{bmatrix}}_{\boldsymbol{A}_{4\times 2}} \underbrace{\begin{bmatrix} {}^2\dot{\theta}_1 \\ {}^2\dot{\theta}_2 \end{bmatrix}}_{{}^2\tilde{\omega}_{3/1}} \\
&= \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{B}_{4\times 4}} \underbrace{\begin{bmatrix} \dot{l_1} \\ \dot{l_2} \\ \dot{l_3} \\ \dot{l_4} \end{bmatrix}}_{\dot{\boldsymbol{q}}_{w\ 4\times 1}}
\end{aligned}
\tag{3.13}
$$

Namely:

$$
\begin{aligned}
\mathbf{B}_{4\times 4}^T \mathbf{A}_{4\times 2}\, {}^2\tilde{\boldsymbol{\omega}}_{3/1} &= \dot{\mathbf{q}}_{w\ 4\times 1} \\
{}^2\tilde{\mathbf{J}}_{4\times 2}\, {}^2\tilde{\boldsymbol{\omega}}_{3/1} &= \dot{\mathbf{q}}_{w\ 4\times 1}
\end{aligned}
\tag{3.14}
$$

Thus we get:

$$
{}^2\tilde{\mathbf{J}} = \mathbf{B}_{4\times 4}^{\mathrm{T}} \cdot \mathbf{A}_{4\times 2} \tag{3.15}
$$

After comparison we find that the two expressions are the same. The next step is to add the "serial" part to $\tilde{\mathbf{J}}$ to form a complete Jacobian of the hybrid mechanism.

3.2 Jacobian Calculation for the Whole Mechanism

When calculating the complete Jacobian, we need to analyze case 1 and case 2 separately because the expressions are different. In case 1, the 3^{rd} DOF lying under the bottom plate , the end effector's angular velocity in world frame can be expressed as:

$$
{}^0\boldsymbol{\omega}_{3/0} = {}^0\boldsymbol{\omega}_{3/1}(\dot{\mathbf{q}}_w) + {}^0\boldsymbol{\omega}_{1/0}(\dot{q}_r) = \mathbf{J} \begin{bmatrix} \dot{\mathbf{q}}_w \\ \dot{q}_r \end{bmatrix} \tag{3.16}
$$

Let's see the details of 3.16 and get:

$$\begin{aligned} {}^0\boldsymbol{\omega}_{3/0} &= {}^0\boldsymbol{\omega}_{3/1} + {}^0\boldsymbol{\omega}_{1/0} \\ &= {}^0\mathbf{R}_2 \begin{bmatrix} {}^2\tilde{\omega}_{3/1} \\ 0 \end{bmatrix} + {}^0\hat{\mathbf{z}}_0 \dot{q}_r \\ &= {}^0\mathbf{R}_2 \begin{bmatrix} {}^2\tilde{\mathbf{J}}^{\dagger}\,\dot{\mathbf{q}}_w \\ 0 \end{bmatrix} + {}^0\mathbf{R}_2\,{}^2\hat{\mathbf{z}}_0 \dot{q}_r \end{aligned} \tag{3.17}$$

Here $\dot{q}_r$ refers to the angular velocity of the 3^{rd} DOF. Then we need to combine $\dot{\mathbf{q}}_w$ and $\dot{q}_r$ together to form a complete $\dot{\mathbf{q}}$ in world frame for the hybrid manipulator.

$$\begin{aligned} {}^0\boldsymbol{\omega}_{3/0} &= {}^0\mathbf{R}_2 \left(\begin{bmatrix} {}^2\tilde{\mathbf{J}}^{\dagger}\,\dot{\mathbf{q}}_w \\ 0 \end{bmatrix} + {}^2\hat{\mathbf{z}}_0 \dot{q}_r \right) \\ &= {}^0\mathbf{R}_2 \begin{bmatrix} \tilde{J}_{11}^{\dagger} & \tilde{J}_{12}^{\dagger} & \tilde{J}_{13}^{\dagger} & \tilde{J}_{14}^{\dagger} & {}^2z_{0x} \\ \tilde{J}_{21}^{\dagger} & \tilde{J}_{22}^{\dagger} & \tilde{J}_{23}^{\dagger} & \tilde{J}_{24}^{\dagger} & {}^2z_{0y} \\ 0 & 0 & 0 & 0 & {}^2z_{0z} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{q}}_w \\ \dot{q}_r \end{bmatrix} \end{aligned} \tag{3.18}$$

The variables of $\tilde{J}_{11}^{\dagger}, \tilde{J}_{12}^{\dagger}, ..., \tilde{J}_{23}^{\dagger}$ are items in ${}^2\tilde{\mathbf{J}}^{\dagger}$, and $[{}^2z_{0x}, {}^2z_{0y}, {}^2z_{0z}]$ represents the vector of ${}^2\hat{\mathbf{z}}_0$, that is, $\hat{\mathbf{z}}_0$ expressed in frame $\{2\}$. So the complete expression of Jacobian is:

$$\mathbf{J} = {}^0\mathbf{R}_2 \begin{bmatrix} {}^2\tilde{\mathbf{J}}^{\dagger} & {}^2z_{0x} \\ & {}^2z_{0y} \\ \mathbf{0} & {}^2z_{0z} \end{bmatrix} = {}^0\mathbf{R}_2 \begin{bmatrix} \tilde{J}_{11}^{\dagger} & \tilde{J}_{12}^{\dagger} & \tilde{J}_{13}^{\dagger} & \tilde{J}_{14}^{\dagger} & {}^2z_{0x} \\ \tilde{J}_{21}^{\dagger} & \tilde{J}_{22}^{\dagger} & \tilde{J}_{23}^{\dagger} & \tilde{J}_{24}^{\dagger} & {}^2z_{0y} \\ 0 & 0 & 0 & 0 & {}^2z_{0z} \end{bmatrix} \tag{3.19}$$

In case 2, $\hat{\mathbf{z}}_4$ is coincident with $\hat{\mathbf{z}}_3$. The end effector's angular velocity in world frame can be expressed as:

$$\begin{aligned} {}^0\boldsymbol{\omega}_{4/1} &= {}^0\boldsymbol{\omega}_{3/1}(\,\dot{\mathbf{q}}_w) + {}^0\boldsymbol{\omega}_{4/3}(\dot{q}_r) \\ &= \mathbf{J} \begin{bmatrix} \dot{\mathbf{q}}_w \\ \dot{q}_r \end{bmatrix} \end{aligned} \tag{3.20}$$

Further we have:

$$
\begin{aligned}
{}^0\boldsymbol{\omega}_{4/1} &= {}^0\boldsymbol{\omega}_{3/1} + {}^0\boldsymbol{\omega}_{4/3} \\
&= {}^0\mathbf{R}_2 \begin{bmatrix} {}^2\tilde{\boldsymbol{\omega}}_{3/1} \\ 0 \end{bmatrix} + {}^0\hat{\mathbf{z}}_3 \dot{q}_r \\
&= {}^0\mathbf{R}_2 \begin{bmatrix} {}^2\tilde{\mathbf{J}}^{\dagger}\, \dot{\mathbf{q}}_w \\ 0 \end{bmatrix} + {}^0\mathbf{R}_2\, {}^2\hat{\mathbf{z}}_3 \dot{q}_r
\end{aligned}
\tag{3.21}
$$

So the complete expression of Jacobian is:

$$
\mathbf{J} = {}^0\mathbf{R}_2 \begin{bmatrix} & {}^2z_{3x} \\ {}^2\tilde{\mathbf{J}}^{\dagger} & {}^2z_{3y} \\ \mathbf{0} & {}^2z_{3z} \end{bmatrix} = {}^0\mathbf{R}_2 \begin{bmatrix} \tilde{J}^{\dagger}_{11} & \tilde{J}^{\dagger}_{12} & \tilde{J}^{\dagger}_{13} & \tilde{J}^{\dagger}_{14} & {}^2z_{3x} \\ \tilde{J}^{\dagger}_{21} & \tilde{J}^{\dagger}_{22} & \tilde{J}^{\dagger}_{23} & \tilde{J}^{\dagger}_{24} & {}^2z_{3y} \\ 0 & 0 & 0 & 0 & {}^2z_{3z} \end{bmatrix} \tag{3.22}
$$

Here $[{}^2z_{3x}, {}^2z_{3y}, {}^2z_{3z}]$ is the vector of ${}^2\hat{\mathbf{z}}_3$, namely $\hat{\mathbf{z}}_3$ expressed in frame $\{2\}$. By comparing these two expressions of Jacobian, we can see that the first four columns of these two cases are the same. The last columns, which present the contribution of the 3^{rd} joint to the angular velocity of end effector, are different due to different rotation axes in these two cases. After computing the expression of $\mathbf{J}$, we have the following relationships:

$$
\mathbf{J}_{3\times 5}\, \dot{\mathbf{q}}_{5\times 1} = \dot{\mathbf{x}}_{3\times 1} \tag{3.23}
$$

$$
\left(\mathbf{J}^{\mathrm{T}}\right)_{5\times 3}\, \mathbf{w}_{ee\ 3\times 1} = \boldsymbol{\tau}_{5\times 1} \tag{3.24}
$$

$\dot{\mathbf{q}} = [\dot{\mathbf{q}}_w, \dot{q}_r]$ contains all the input velocities, and $\dot{\mathbf{x}}$ refer to end effector's angular velocity. $\boldsymbol{\tau}$ represents actuations including wire tensions and one moment applied on the 3^{rd} DOF, and $\mathbf{w}_{ee}$ is end effector's wrench. Moreover, in the following sections, we make use of two sub-matrices of $\mathbf{J}$ which are associated with the wire controlled wrist. The first matrix is defined as the first four/three (based on the number of actuations) columns of $\mathbf{J}$. Here we

use four wires as an example, and the matrix can be written as:

$$\mathbf{J}_w \triangleq {}^0\mathbf{R}_2 \begin{bmatrix} \tilde{J}_{11}^{\dagger} & \tilde{J}_{12}^{\dagger} & \tilde{J}_{13}^{\dagger} & \tilde{J}_{14}^{\dagger} \\ \tilde{J}_{21}^{\dagger} & \tilde{J}_{22}^{\dagger} & \tilde{J}_{23}^{\dagger} & \tilde{J}_{24}^{\dagger} \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.25}$$

The second matrix is the 4 by 3 Jacobian for parallel wrist expressed in frame $\{0\}$, which is defined as:

$$\mathbf{J}_p \triangleq \begin{bmatrix} & & 0 \\ & & 0 \\ {}^2\tilde{\mathbf{J}}_{4\times 2} & & \\ & & 0 \\ & & 0 \end{bmatrix} {}^0\mathbf{R}_2 \tag{3.26}$$

3.3 Conclusion

In this part, we first use two methods, virtual work method and loop-closure kinematics method to analyze the Jacobian expression ${}^2\tilde{\mathbf{J}}$ for parallel robot. The results of these two methods are the same, which demonstrates the correctness of them. Then use the equation of end effector's velocity to get complete Jacobian $\mathbf{J}$ by partitioning items of joint space velocity. Moreover, we also define $\mathbf{J}_w$ as velocity relationships between end effector and wires, and $\mathbf{J}_p$ as Jacobian for parallel robot expressed in frame $\{0\}$ for later use.

Chapter 4

Stiffness And Compliance Analysis

In the course of the calculation of wrist workspace for wires having finite stiffness we make use of the wrist compliance. We therefore present the stiffness and compliance model in this chapter.

4.1 Parallel Robot Stiffness

The stiffness of parallel robot is defined as:

$$\Delta \mathbf{W}_e = \mathbf{K}\Delta\theta \tag{4.1}$$

In equation 4.1, $\Delta\theta$ is the alteration of position/orientation; $\Delta \mathbf{W}_e$ represents the corresponding change of external wrench, and $\mathbf{K}$ denotes the stiffness matrix. In this project, $\Delta\theta = [\Delta\theta_x, \Delta\theta_y, \Delta\theta_z]^T$ which is the deflection of end effector's orientations in workspace; $\Delta \mathbf{W}_e = [M_{ex}, M_{ey}, M_{ez}]^T$, the change of external moments, and for each item k_{ij} in $\mathbf{K}$, we have:

$$k_{ij} \triangleq \frac{d\,\mathbf{W}_{ei}}{d\theta_j} = \frac{d}{d\theta_j}\left(\mathbf{J}_p^T \boldsymbol{\tau}_w\right)_i = \frac{d}{d\theta_j}\left[\left(\mathbf{J}_p^T\right)_i \boldsymbol{\tau}_w\right] \tag{4.2}$$

Here $\mathbf{J}_p$ is the Jacobian expression for parallel robot, which is defined in Equation 3.26 in Chapter 3. The subscripts i or j means the i^{th} or j^{th} row of $\mathbf{J}_p^T$. Let us expand equation 4.2 and get:

$$k_{ij} = \frac{d}{d\theta_j}\left[\left(\mathbf{J}_p^T\right)_i \boldsymbol{\tau}_w\right] = \frac{d\left(\mathbf{J}_p^T\right)_i}{d\theta_j}\boldsymbol{\tau}_w + \left(\mathbf{J}_p^T\right)_i \frac{d\boldsymbol{\tau}_w}{d\theta_j} \tag{4.3}$$

The first item on the right side of Equation 4.3 is called active stiffness and the second passive stiffness [29]. Let us expand the passive stiffness first:

$$\frac{d\boldsymbol{\tau}}{d\theta_j} = \frac{\partial\boldsymbol{\tau}_w}{\partial q_1}\frac{\partial q_1}{\partial\theta_j} + \frac{\partial\boldsymbol{\tau}_w}{\partial q_2}\frac{\partial q_2}{\partial\theta_j} + \ldots + \frac{\partial\boldsymbol{\tau}_w}{\partial q_4}\frac{\partial q_4}{\partial\theta_j} \tag{4.4}$$

We can see that the items $\frac{\partial \tau_w}{\partial q_1}, \ldots, \frac{\partial \tau_w}{\partial q_4}$ are rows of joint stiffness $\mathbf{K}_d$, and $\frac{\partial q_1}{\partial \theta_j}, \ldots, \frac{\partial q_4}{\partial \theta_j}$ are just the j^{th} column of Jacobian matrix. Thus equation 4.3 can be written as:

$$k_{ij} = \frac{d\left(\mathbf{J}_p^T\right)_i}{d\theta_j}\tau_w + \left(\mathbf{J}_p\right)_i \boldsymbol{K}_d \left(\mathbf{J}_p\right)^j \tag{4.5}$$

Referring to the active stiffness term $\frac{d\left(\mathbf{J}_p^T\right)_i}{d\theta_j}\tau_w$, the vector τ_w is a 4 by 1 vector representing wire tensions, and before we illustrate $\frac{d\left(\mathbf{J}_p^T\right)_i}{d\theta_j}$, we should first define $\frac{d\mathbf{J}_p^T}{d\theta}$ as a 3-dimension matrix which consists of 3 "layers". On each layer is a 2-dimension matrix $\frac{\partial \mathbf{J}_p^T}{\partial \theta_j}, j = x, y, z$. Thus $\frac{d\left(\mathbf{J}_p^T\right)_i}{d\theta_j}$ means the ith row on the jth layer. Based on the analysis of active stiffness and passive stiffness, we can write the ith column of $\mathbf{K}$ as:

$$\left(\mathbf{K}\right)^j = \frac{\partial \mathbf{J}_p^T}{\partial \theta_j}\tau_w + \mathbf{J}_p^T \boldsymbol{K}_d \left(\mathbf{J}_p\right)^j \tag{4.6}$$

Moreover, since the active stiffness is relatively small compared with passive stiffness according to Simaan, et al. [29], we neglect the active item and use passive stiffness to approximate $\mathbf{K}$, that is:

$$\mathbf{K} = \mathbf{J}_p^T \mathbf{K}_d \mathbf{J}_p = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix} \tag{4.7}$$

And the external moment can be expressed as:

$$\begin{aligned} \Delta M_{ex} &= k_{xx}\Delta\theta_x + k_{xy}\Delta\theta_y + k_{xz}\Delta\theta_z \\ \Delta M_{ey} &= k_{yx}\Delta\theta_x + k_{yy}\Delta\theta_y + k_{yz}\Delta\theta_z \\ \Delta M_{ez} &= k_{zx}\Delta\theta_x + k_{zy}\Delta\theta_y + k_{zz}\Delta\theta_z \end{aligned} \tag{4.8}$$

The coefficient k_{ij} affects the robot reaction in i for a perturbation in j direction. Figure 4.3, 4.4 and 4.5 shows the contours of k_{xx}, k_{yy} and k_{zz} for 4-wire mechanism. Figure 4.6,

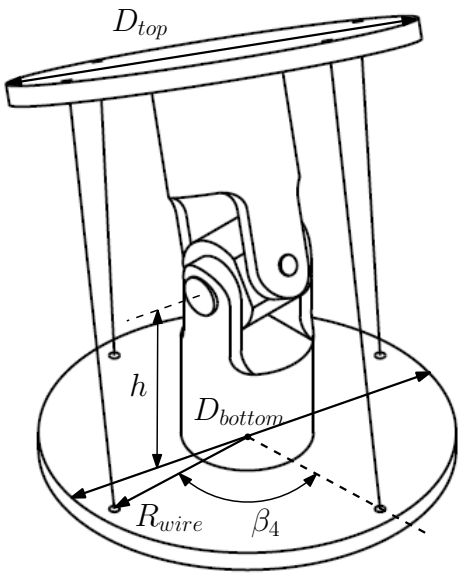

Figure 4.1: 4-wire parallel robot dimensions

Table 4.1: 4-Wire Parallel Robot Dimensions

Item	Symbol	Dimension
Top platform diameter	D_{top}	40mm
Bottom platform diameter	D_{bottom}	40mm
Wires separation distance	R_{wire}	18mm
Wires separation angle	β_4	90°
Hook heights	h	19mm

4.7 and 4.8 are stiffness contours for 3-wire wrist.

In these simulations, we assumed $\mathbf{K}_{di} = 1N/mm$ for each wire and the robot dimensions are give in Figure 4.1 and Table 4.1 for 4-wire mechanism, while for 3-wire robot the dimensions are given in Figure 4.2 and Table 4.2.

These figures can give us a more intuitive sense of the stiffness. As we have explained, coefficients k_{xx}, k_{yy}, k_{zz} present the extent to which the robot react in directions of x, y and z due to perturbations in x, y and z respectively. Moreover, we can also see that the 4-wire mechanism has higher stiffness than 3-wire wrist.

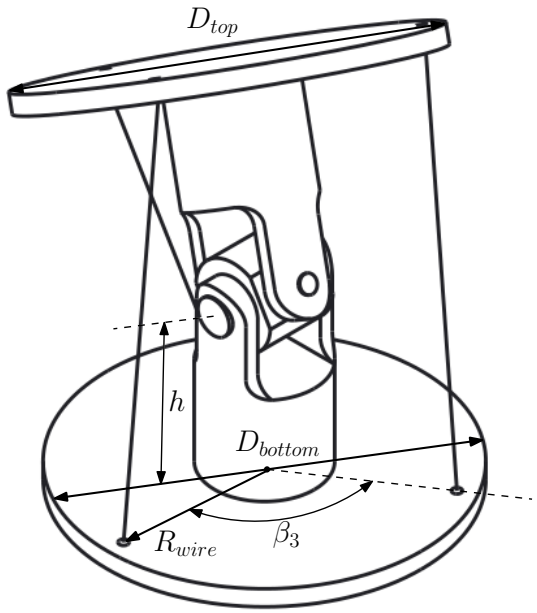

Figure 4.2: 3-wire parallel robot dimensions

Table 4.2: 3-Wire Parallel Robot Dimensions

Item	Symbol	Dimension
Top platform diameter	D_{top}	40mm
Bottom platform diameter	D_{bottom}	40mm
Wires separation distance	R_{wire}	18mm
Wires separation angle	β_3	120°
Hook heights	h	19mm

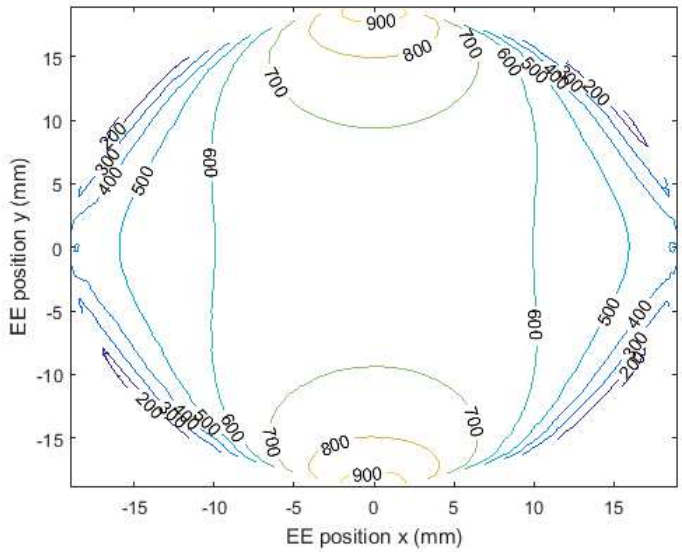

Figure 4.3: Stiffness contours of k_{xx} from 4-wire mechanism(unit:N/mm)

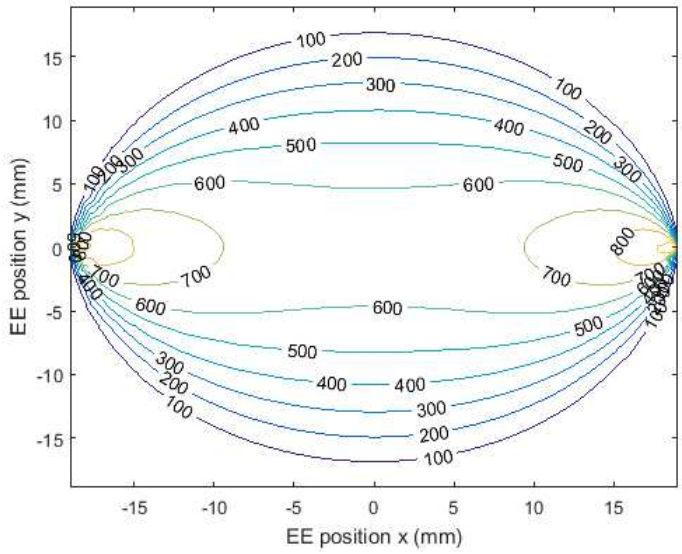

Figure 4.4: Stiffness contours of k_{yy} from 4-wire mechanism(unit:N/mm)

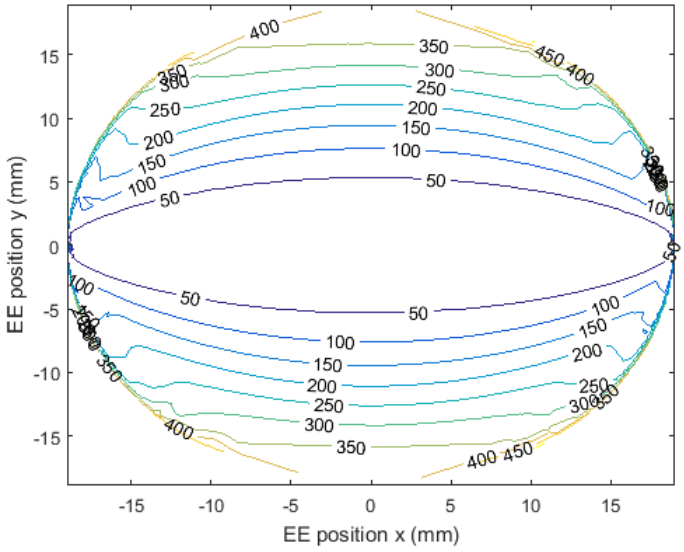

Figure 4.5: Stiffness contours of k_{zz} from 4-wire mechanism(unit:N/mm)

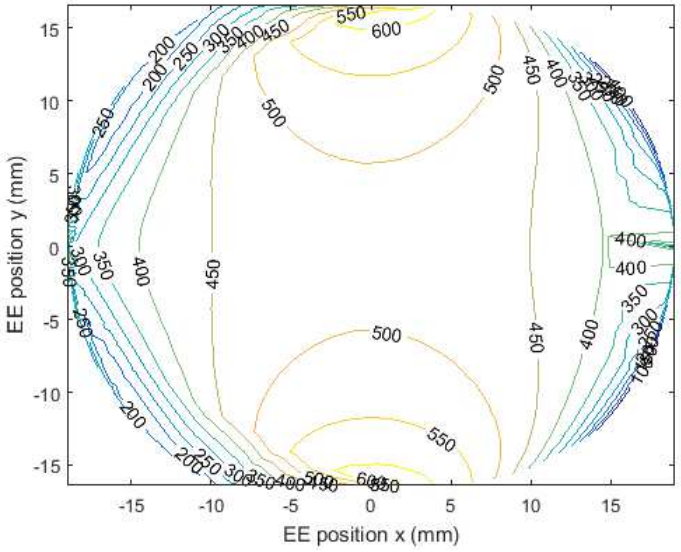

Figure 4.6: Stiffness contours of k_{xx} from 3-wire mechanism(unit:N/mm)

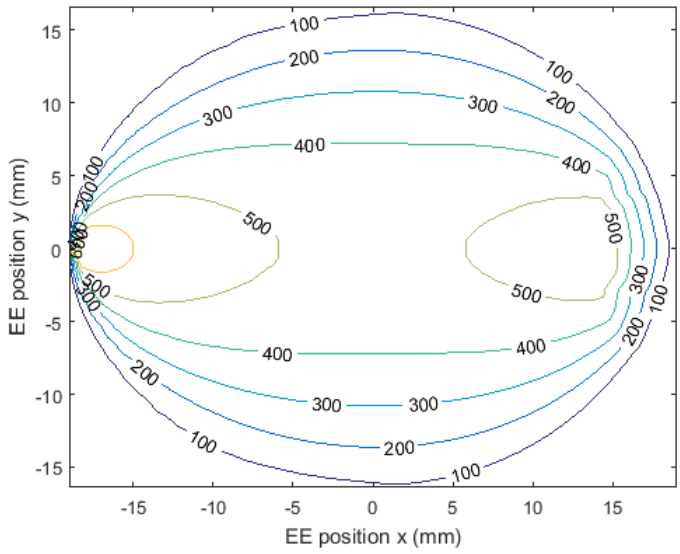

Figure 4.7: Stiffness contours of k_{yy} from 3-wire mechanism(unit:N/mm)

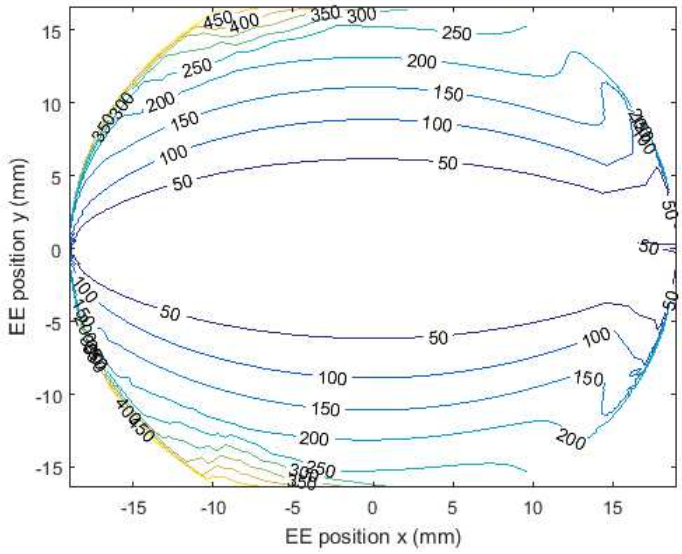

Figure 4.8: Stiffness contours of k_{zz} from 3-wire mechanism(unit:N/mm)

4.2 Hybrid robot compliance

The stiffness of the hybrid wire-actuated manipulator is contributed by both the cable stiffness and the internal forces in the system which refer to cable tensions [30]. Because the procedures that derive the stiffness model are the same for either 4 wires or 3 wires, again we will only discuss the 4 wire mechanism.

In this project, the wrist is a hybrid parallel-serial mechanism so that compliance matrix should be used in stiffness modeling. That is, what we want to find is the relationship between end effector torque and displacement as:

$$\mathbf{C}\Delta\,\mathbf{w}_{ee\ 3\times1} = \Delta\,\mathbf{x}_{3\times1} \tag{4.9}$$

Here $\mathbf{C}$ is the compliance matrix for the hybrid wrist and $\Delta\,\mathbf{x}_{3\times1}$ is defined as $\Delta\,\mathbf{x}_{3\times1} = [\Delta\theta_x, \Delta\theta_x, \Delta\theta_x]^T$. Since we have the relationship between joint actuations τ and $\mathbf{w}_{ee}$ as:

$$\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\,\mathbf{w}_{ee} \tag{4.10}$$

Taking the derivative of $\mathbf{w}_{ee}$ on both sides:

$$\frac{\delta\boldsymbol{\tau}}{\delta\,\mathbf{w}_{ee}} = \frac{\delta\mathbf{J}^{\mathrm{T}}}{\delta\,\mathbf{w}_{ee}}\,\mathbf{w}_{ee} + \mathbf{J}^{\mathrm{T}} \tag{4.11}$$

If we neglect pre-load, equation 4.11 can be approximately written as:

$$\Delta\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\Delta\,\mathbf{w}_{ee} \tag{4.12}$$

which, namely, is the first order approximation of $\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\,\mathbf{w}_{ee}$. Moreover, equation 4.12 can be expanded to:

$$\Delta\boldsymbol{\tau} = \mathbf{J}^{\mathrm{T}}\Delta\,\mathbf{w}_{ee} = \mathbf{K}_{d\ 5\times5}\Delta\,\mathbf{q}_{5\times1} = \mathbf{K}_{d\ 5\times5}\,\mathbf{J}^{\dagger}_{5\times3}\Delta\,\mathbf{x}_{3\times1} \tag{4.13}$$

Here $\mathbf{K}_{d\ 5\times5} = diag([k_{w1}, k_{w2}, k_{w3}, k_{w3}, k_r]^T)$ is the hybrid joint space stiffness, in which $k_{wi} = \frac{E_i A_i}{l_i} (i = 1,2,3,4)$ is the stiffness of wire and k_r is the stiffness of the 3^{rd} DOF input joint. Then we can derive the following equation from equation 4.13:

$$\mathbf{J}_{3\times5}\,\mathbf{K}_{d\ 5\times5}^{-1}\,\mathbf{J}_{5\times3}^{\mathrm{T}}\Delta\mathbf{w}_{ee\ 3\times1} = \Delta\mathbf{x}_{3\times1} \tag{4.14}$$

So the expression for compliance matrix is:

$$\mathbf{C}_{3\times3} = \mathbf{J}_{3\times5}\,\mathbf{K}_{d\ 5\times5}^{-1}\,\mathbf{J}_{5\times3}^{\mathrm{T}} \tag{4.15}$$

The expression for $\mathbf{J}$ was given in Equation 3.19 and 3.22 depending on where the active revolute joint is positioned.

Chapter 5

Workspace Analysis

This chapter will analyze the workspace of wire actuated universal joint mechanism based on the analysis of Jacobian and stiffness/compliance, based on the work of Hamid and Simaan [28]. We will first explain how we define workspace, and then come up with methods for computing both the workspace supposing that the wires have infinite stiffness and finite stiffness. Second, for various dimension configurations, we calculate the maximum tilt angles for each one to give a sense of workspace comparison. At the end of this chapter, we will give methods for wire tension optimization.

5.1 Workspace Analysis with Infinite Stiffness Wires

In Equation 3.24, not every solution for $\boldsymbol{\tau}$ is feasible for an arbitrarily oriented $\mathbf{w}_{ee}$. Because wires are used, the value of $\tau_i, i = 1,2,3,4$ must be nonnegative. When no such positive $\boldsymbol{\tau}$ exist, it is impossible to hold the robot in static equilibrium at that configuration [31]. For equation 3.24, we need to extract a sub-equation that only contains relationships between wire tensions and end effector wrench $\mathbf{w}_{ee}$, which is the first four lines of equation 3.24.

$$\mathbf{J}_w^{\mathrm{T}} \mathbf{w}_{ee} = \boldsymbol{\tau}_w \tag{5.1}$$

$$\mathbf{w}_{ee} = \left(\mathbf{J}_w^{\mathrm{T}}\right)^{\dagger} \boldsymbol{\tau}_w \tag{5.2}$$

$\mathbf{J}_w^{\mathrm{T}}$ means partial $\mathbf{J}^{\mathrm{T}}$, containing only the first 4 rows of the hybrid robot Jacobian, which was defined in Equation 3.25 in Chapter 3. The solution for equation 5.2 can be written as:

$$\boldsymbol{\tau}_w = \boldsymbol{\tau}_{wp} + \lambda\, \boldsymbol{\tau}_{wh} \tag{5.3}$$

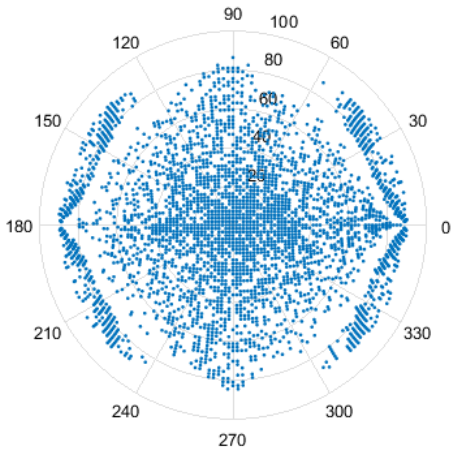

Figure 5.1: Example of 4-wire wrist workspace with infinitely stiffness wires in polar coordinate system (unit: degree)

where $\boldsymbol{\tau}_{wp}$ is the particular solution and $\boldsymbol{\tau}_{wh}$ is the homogenous solution which is in the null space of $\left(\mathbf{J}_w^{\mathrm{T}}\right)^{\dagger}$. If there is a left null vector of $\left(\mathbf{J}_w^{\mathrm{T}}\right)^{\dagger}$ with strictly positive components, then the robot can achieve static equilibrium. A nonsingular configuration is kinematically fully constrained if and only if there is a left null vector of $\left(\mathbf{J}_w^{\mathrm{T}}\right)^{\dagger}$ with the property that each of its components is positive [31]. That is to say, in order to guarantee tensions in the wires, the components of $\boldsymbol{\tau}_{wh}$ must have the same sign [28].

Figure 5.1 is an example of workspace given a certain configuration in which the diameters of both top and bottom plate are 40mm and hook height is 19mm. In this polar coordinate system, the length of radius represents the maximum magnitude of tilt angle in that direction. Moreover, the workspace, namely the maximum tilt angle will vary when we change the ratio of height over top diameter or the ratio of bottom diameter over top diameter. As a comparison, Figure 5.2 shows the workspace when the wrist actuation has only three wires which are evenly arranged around center axis. From these two figures we can find that 4-wire wrist can achieve greater tilt angle and have larger workspace than 3-wire wrist.

Moreover, we should also take physical collision into consideration when calculating workspace. Basically the collision will happen when upper plate and lower plate touch

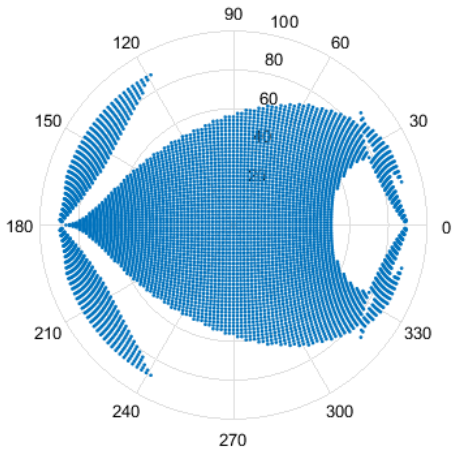

Figure 5.2: Example of 3-wire wrist workspace with infinitely stiffness wires in polar coordinate system (unit: degree)

each other. In this case, the maximum tilt angle of end effector should be:

$$\beta = \pi - 2\,\text{Atan2}\left(\frac{min\left(D_{top}, D_{bottom}\right)}{2 \cdot height}\right) \tag{5.4}$$

In order to get reasonable design atlas, we use various ratios of $height/D_{bottom}$ and D_{top}/D_{bottom} when calculating maximum tilt angles. Then we get the superimposed result when considering both wire tensions and physical collision, as shown in Figure 5.3. Again, as a comparison, Figure 5.4 shows the workspace when the wrist has 3 actuation wires. We can see that the two figures are similar.

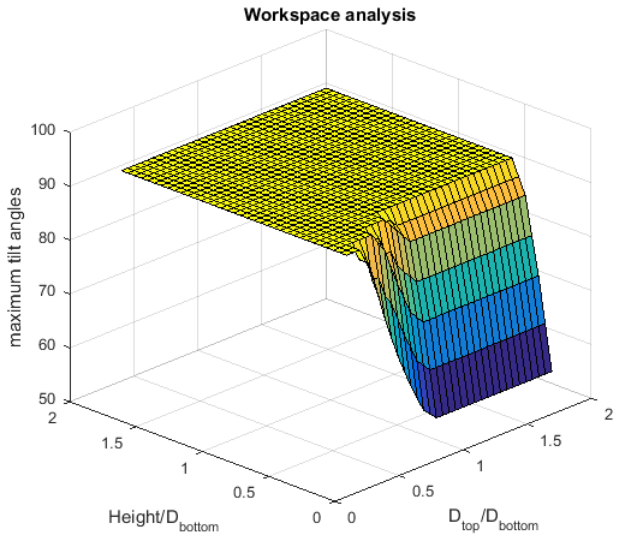

Figure 5.3: Superimposed workspace of 4 wires wrist when considering both wrench closure and physical collision

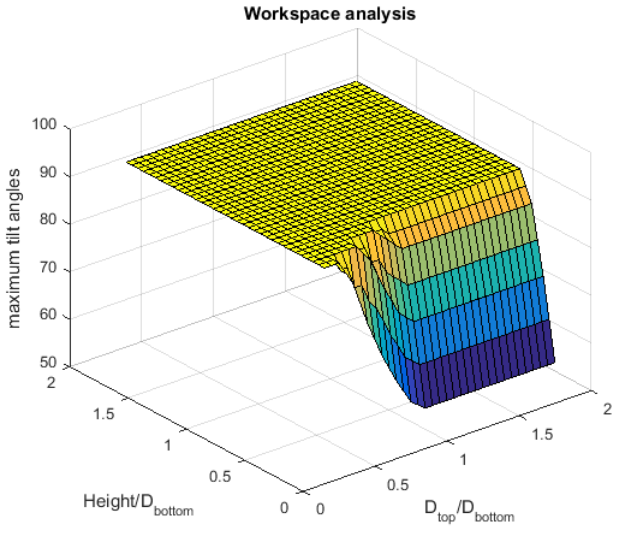

Figure 5.4: Superimposed workspace of 3 wires wrist when considering both wrench closure and physical collision

5.2 Workspace Analysis with Finite Stiffness Wires

In the previous section, we compute workspace assuming wrist has infinite stiffness joints. However, since wires have relatively higher extensibility than other actuation methods, it is critical to take finite stiffness into account when calculating wrench closure workspace [28]. Thus in this section, we will modify the workspace in previous section based on the stiffness/compliance model.

We can calculate alterations in joint space when given certain external wrench using compliance matrix. Moreover, we can also know the direction where the wrist has the least stiffness by finding the greatest eigenvalue's eigenvector of $\mathbf{C}$. According to these qualities, we can modify the workspace obtained in the previous section.

The reason why we need to modify workspace is that when we take finite stiffness of wires into consideration, the location of end effector will probably be changed to an invalid position where one cannot guarantee tensions in all four wires. Thus the recalculated workspace must make up for end effector's deflection.

However, it is not easy to get accurate modified workspace because the system stiffness depends greatly on Jacobian and moreover, the expression of Jacobian varies according to different configurations. But we can still get a reasonable approximate result assuming adjacent positions have same Jacobian expressions and external wrench is small enough. Here we use two methods to get an approximate workspace. The purpose of the two method is the same, that is, to subtract the variation from original workspace for a given external torque. In the first method, we calculate the variation for once and assume Jacobian expression is constant while in the second method, we split the variation into several steps and recalculate the Jacobian for each step. The procedure for the first method is listed below:

1. We calculate initial workspace supposing that wire stiffness is infinite;

2. For each point in the workspace, compute the compliance matrix and the greatest possible deflection of end effector's position;

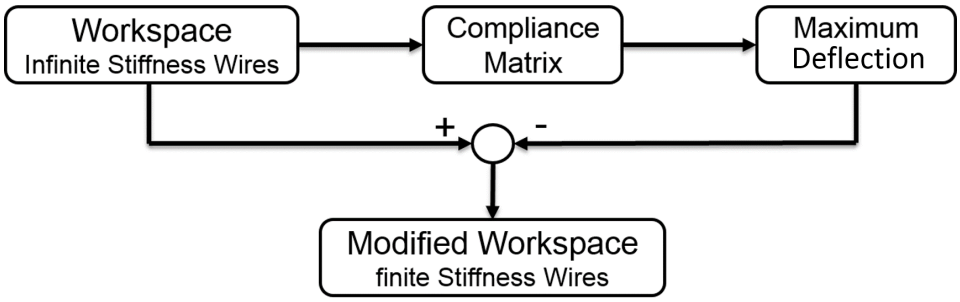

Figure 5.5: Procedures of workspace calculation in method I

3. Subtract the alteration from initial workspace to get the approximation of ”real” workspace. We need to be careful that it is not simply just subtraction.

From the formation of compliance matrix we can know that end effector wrench $\Delta \mathbf{w}_{ee\ 3\times1}$ refers to moments in X, Y and Z directions and joint space deflection $\Delta \mathbf{x}_{3\times1}$ refers to a small rotation about a fixed axis where the orientation of the axis is the unit vector of $\Delta \mathbf{x}_{3\times1}$ and the rotation angle is the norm of $\Delta \mathbf{x}_{3\times1}$. Thus we can use the rotation matrix in Equation 5.5 to help revise workspace. In Equation 5.5, vector $[u,v,w]$ is the unit vector of rotation axis and $\delta\alpha$ is the rotation angle.

$$\mathbf{R} = \begin{bmatrix} u^2+(v^2+w^2)\cos\delta\alpha & uv(1-\cos\delta\alpha)-w\sin\delta\alpha & uw(1-\cos\delta\alpha)+v\sin\delta\alpha \\ uv(1-\cos\delta\alpha)+w\sin\delta\alpha & v^2+(u^2+w^2)\cos\delta\alpha & vw(1-\cos\delta\alpha)-u\sin\delta\alpha \\ uw(1-\cos\delta\alpha)-v\sin\delta\alpha & vw(1-\cos\delta\alpha)+u\sin\delta\alpha & w^2+(u^2+v^2)\cos\delta\alpha \end{bmatrix} \tag{5.5}$$

The first two steps are the same in method II. But for step 3, instead of supposing constant Jacobian and computing Jacobian only once, we divide $\Delta \mathbf{x}$ into ”small” parts and recalculate Jacobian and external wrench for each step separately. Then subtract the displacement each time until the sum of ”small” variations equals $\Delta \mathbf{x}$. Figure 5.6 is an example of workspace comparison when $height/Dtop = 0.5, Dbottom/Dtop = 1$, the norm of external force is $50N \cdot mm$, stiffness of wires is $1N/mm$. Figure 5.7 is the result of workspace using second method with the same structure configuration as Figure 5.6.

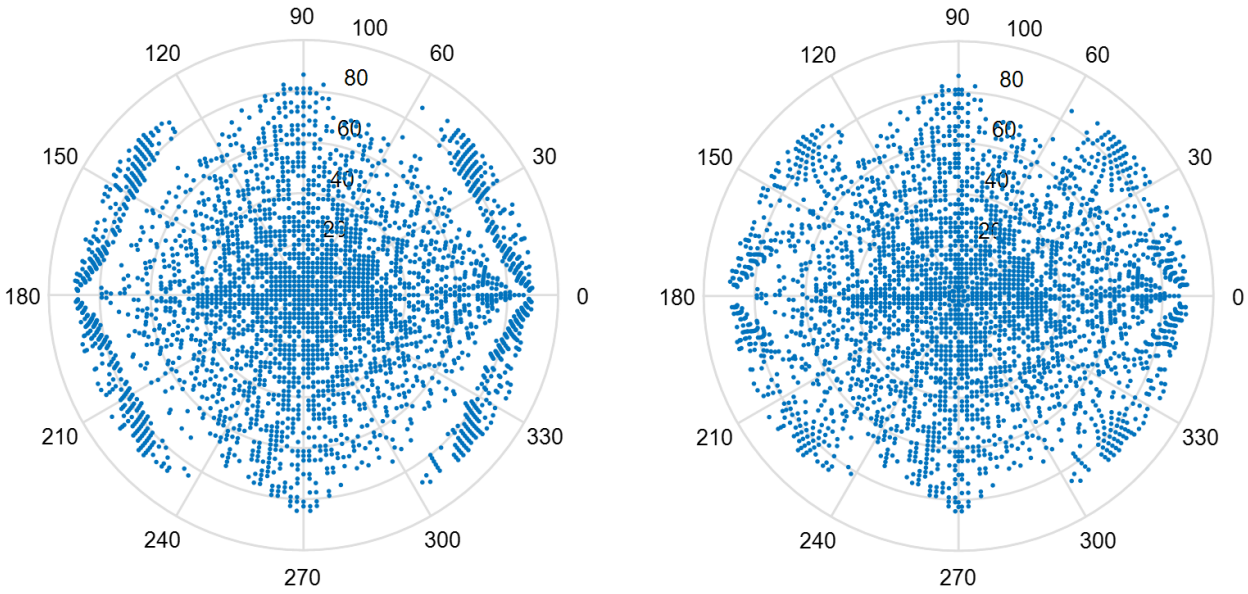

Figure 5.6: Comparison of 4-wire workspace with infinite stiffness wires (left) and finite stiffness wires (right) in method I in polar coordinate system (unit: degree)

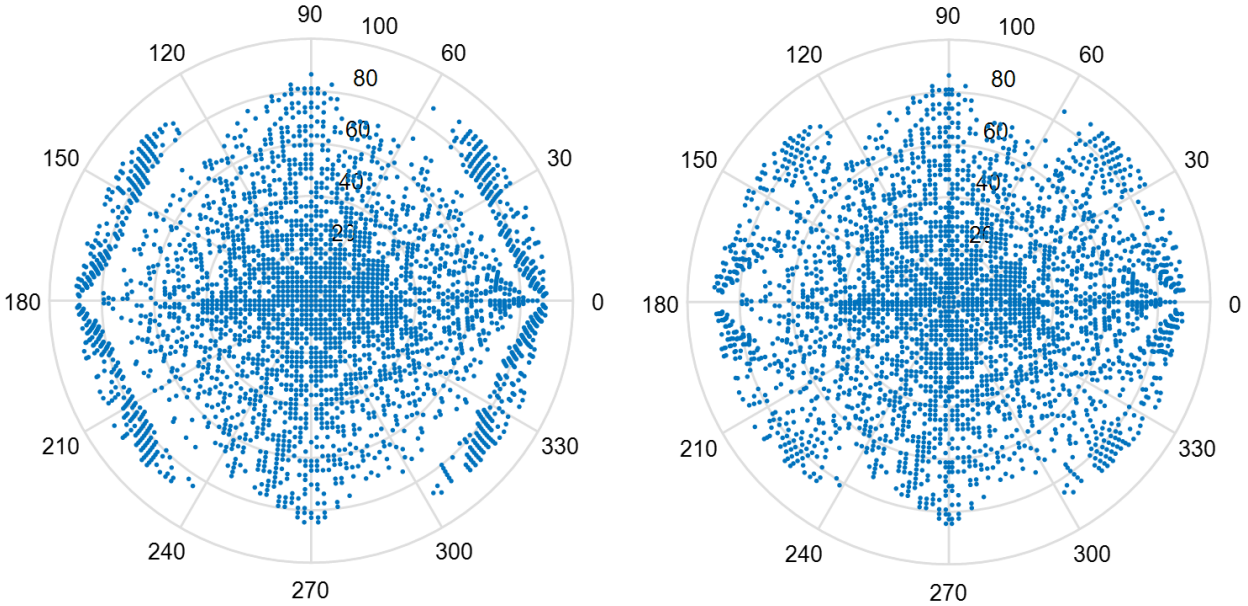

Figure 5.7: Comparison of 4-wire workspace with infinite stiffness wires (left) and finite stiffness wires (right) in method II in polar coordinate system (unit: degree)

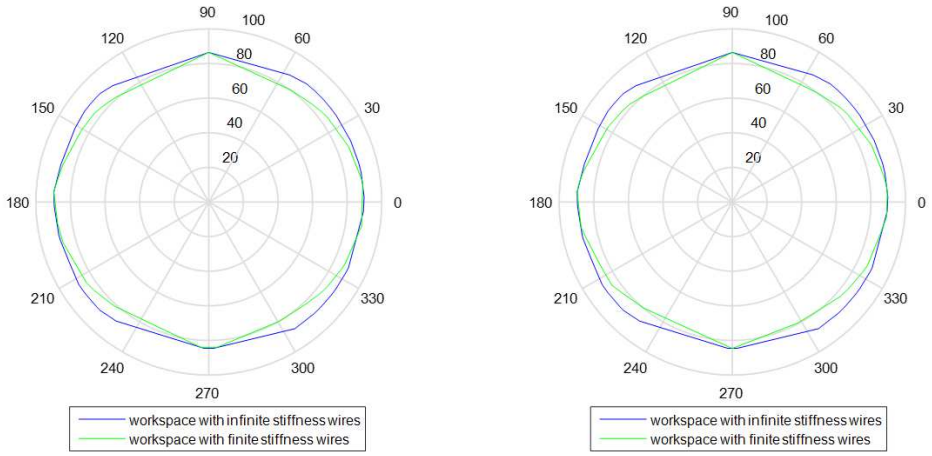

Figure 5.8: Comparison of workspace boundaries using method I (left) and method II (right) in polar coordinate system (unit: degree)

In order to see the difference clearly between these two methods, two boundaries of workspace are plotted in Figure 5.8. For a 3-wire wrist, the results can be seen in Figure 5.9 (method I), Figure 5.10 (method II) and figure 5.11. In both 3-wire and 4-wire simulations we can see the two methods have almost the same results.

From these comparisons we can see that it is not sufficient to consider only wrench closure when analyzing workspace because wires must have finite stiffness which makes actual feasible workspace areas smaller. Moreover, by comparing workspace when wires are of different stiffness as shown in Figure 5.12, we can find that the smaller wire stiffness can result in greater deduction from work closure workspace.

We also scan various ratio combinations of $height/D_{bottom}$ and D_{top}/D_{bottom}, and get the modified maximum tilt angles when assuming finite stiffness wires, as shown in Figure 5.13 and 5.14 for 4 wires and 3 wires respectively. From these two figures we can find that when assuming finite stiffness wires for both 4-wire and 3- wire mechanism, the maximum tilt angles that the wrist can achieve are similar.

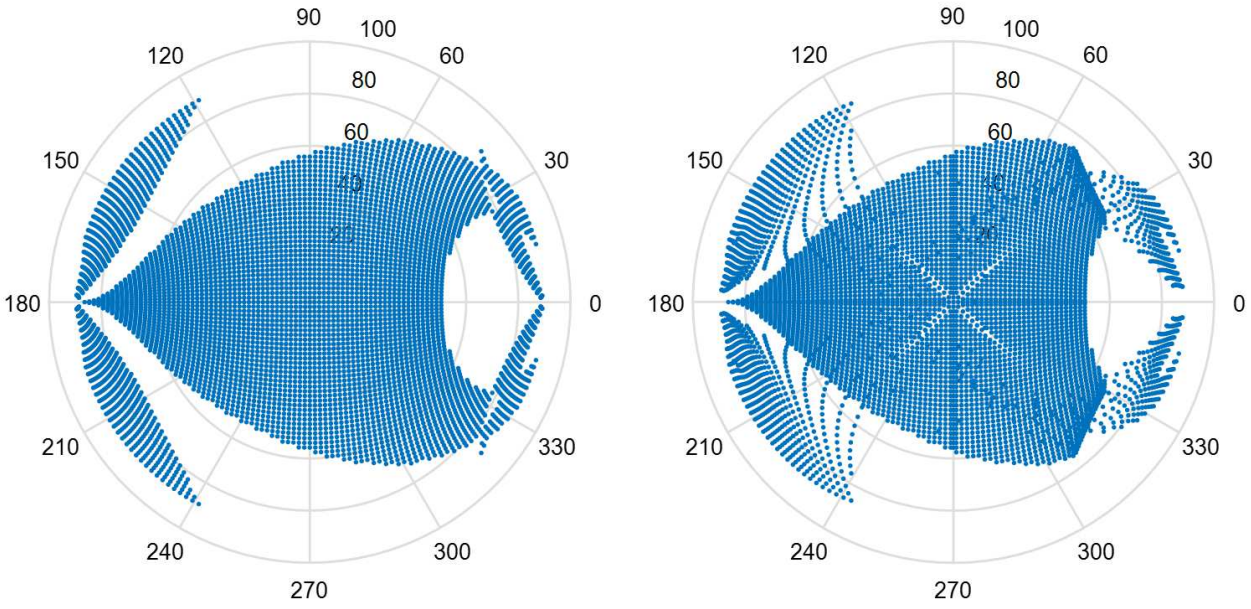

Figure 5.9: Comparison of 3 wires workspace with infinite stiffness wires (left) and finite stiffness wires (right) in method I in polar coordinate system (unit: degree)

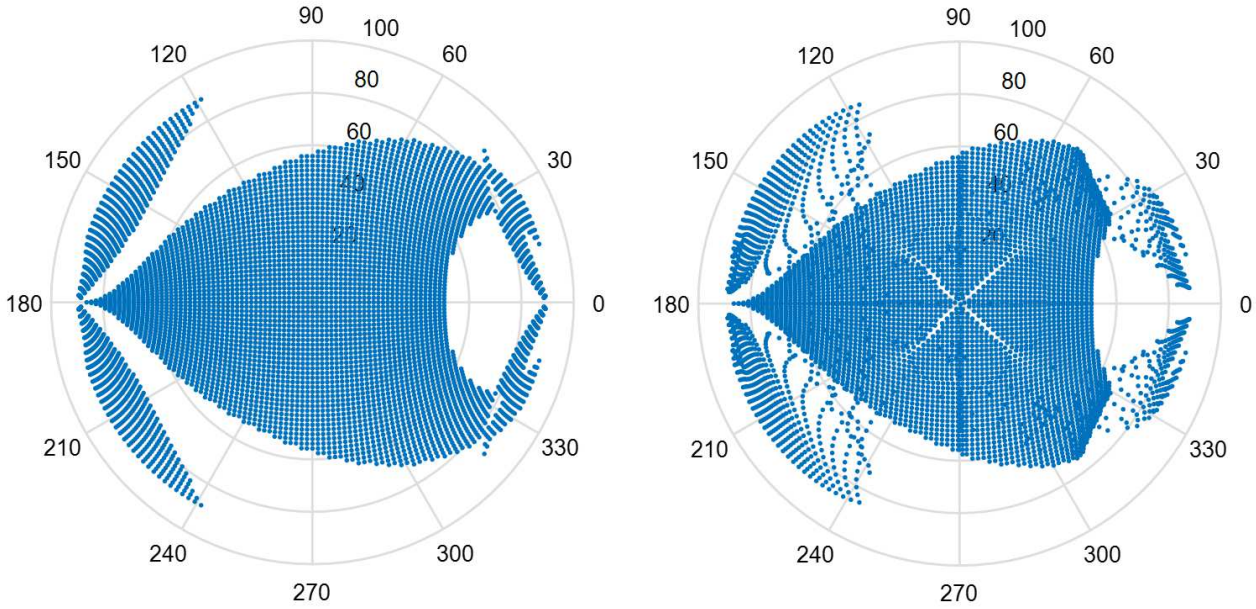

Figure 5.10: Comparison of 3 wires workspace with infinite stiffness wires (left) and finite stiffness wires (right) in method II in polar coordinate system (unit: degree)

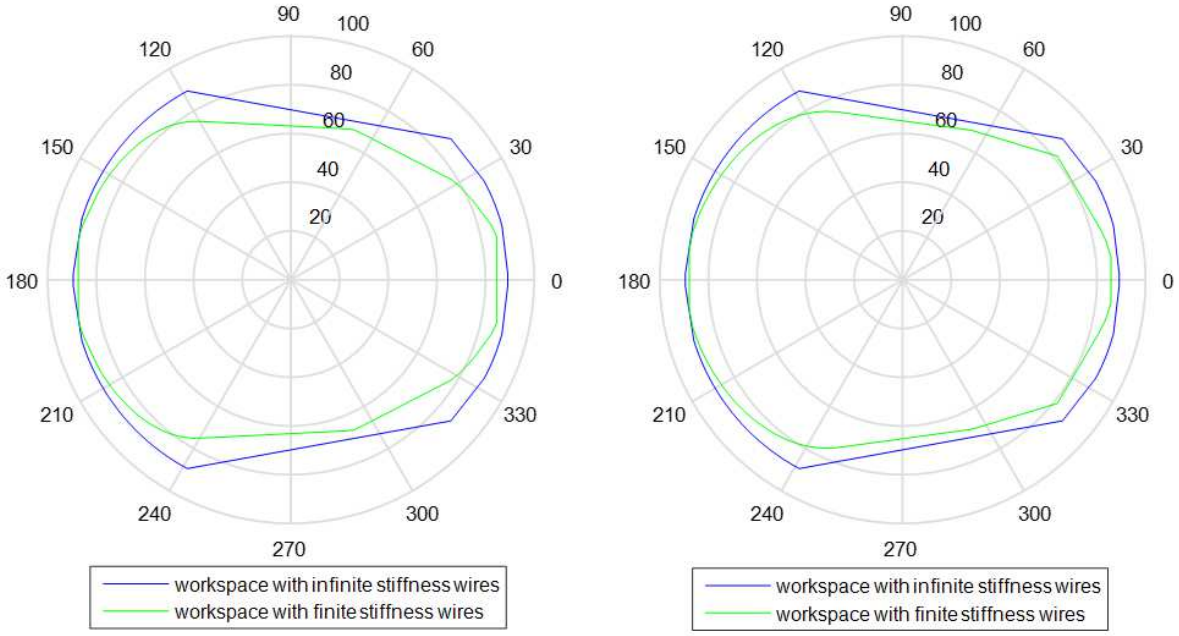

Figure 5.11: Comparison of 3 wires workspace boundaries using method I (left) and method II (right) in polar coordinate system (unit: degree)

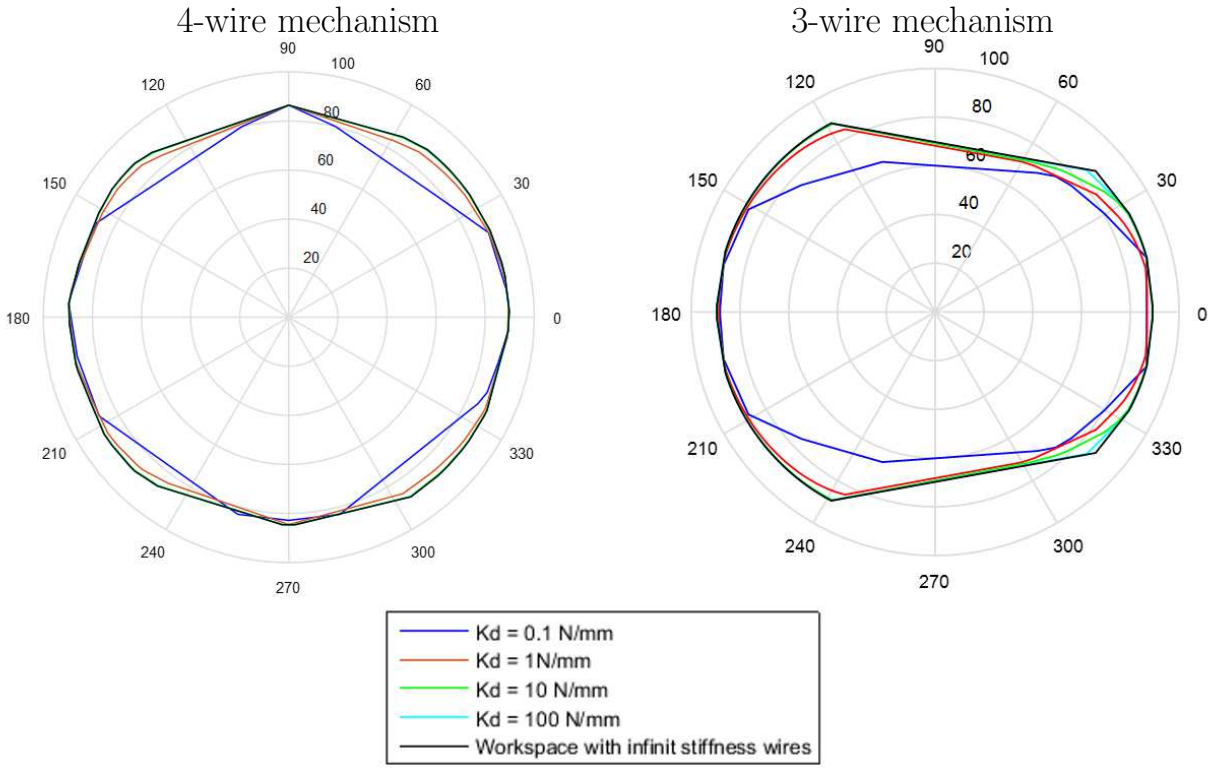

Figure 5.12: Comparison of 4-wire workspace (left) and 3-wire workspace (right) for wires of various stiffness in polar coordinate system (unit: degree)

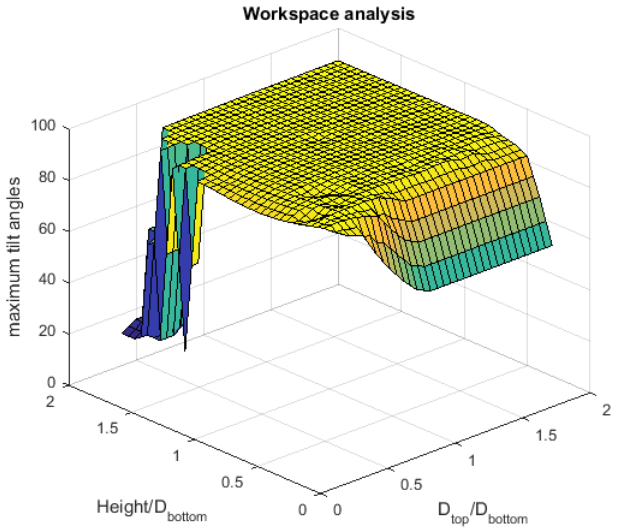

Figure 5.13: Superimposed workspace of 4-wire wrist assuming finite stiffness wires

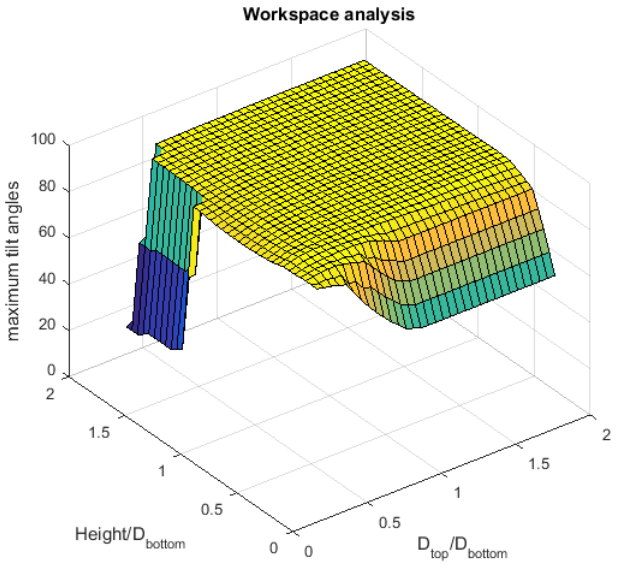

Figure 5.14: Superimposed workspace of 3-wire wrist assuming finite stiffness wires

5.3 Wire Tension Analysis

In order to provide rational design atlases, we need to make sure that wire tensions' magnitudes are within reasonable range.

From equation 5.1 we can see that the wire tensions depend on end effector torque and expression of Jacobian. Moreover, we can use equation 5.6 to approximately express the relationship of magnitudes between τ_w and $\mathbf{w}_{ee}$:

$$\mathbf{w}_{ee}^{\mathrm{T}}\,\mathbf{J}_w\,\mathbf{J}_w^{\mathrm{T}}\,\mathbf{w}_{ee} = \tau_w^{\mathrm{T}}\,\tau_w \tag{5.6}$$

Here $\tau_w^{\mathrm{T}}\,\tau_w$ is actually the sum of wire tensions' squares, but in a sense we can use this item to represent the sum of wire tensions. In this case, the eigenvalues of $\mathbf{J}_w\,\mathbf{J}_w^{\mathrm{T}}$ indicate the scaling factor between $\|\,\mathbf{w}_{ee}\|$ and $\|\,\tau_w\|$. Thus when external torque is given, the maximum eigenvalue of $\mathbf{J}_w\,\mathbf{J}_w^{\mathrm{T}}$ will set an upper bound for wire tensions.

In order to choose proper scaling factors, we find out the maximum eigenvalues of $\mathbf{J}_w\,\mathbf{J}_w^{\mathrm{T}}$ for each ratio pair configuration: ratio of height over bottom plate diameter and ratio of top plate diameter over the bottom. In every configuration, we scan the whole workspace and take the average of maximum eigenvalues as the representative of that ratio pair. Figure 5.15 and Figure 5.16 show the results of average maximum eigenvalues for 4 wires wrist and 3 wire wrist respectively, and smaller values mean better performance, indicating smaller scaling factors from end effector moment to wire tensions.

However, it is not sufficient to choose smaller maximum eigenvalue to optimize wire tensions. It is also very important to use isotropy as criteria to evaluate the performance of static manipulability. Here we define isotropy by using inverse condition number $\frac{1}{\kappa}$.

$$\frac{1}{\kappa} = \frac{\sigma_{min}}{\sigma_{max}}, \qquad 0 \leq \frac{1}{\kappa} \leq 1 \tag{5.7}$$

σ_{min} and σ_{max} represent the minimum and maximum eigenvalue of $\mathbf{J}_w\,\mathbf{J}_w^{\mathrm{T}}$ respectively. Ac-

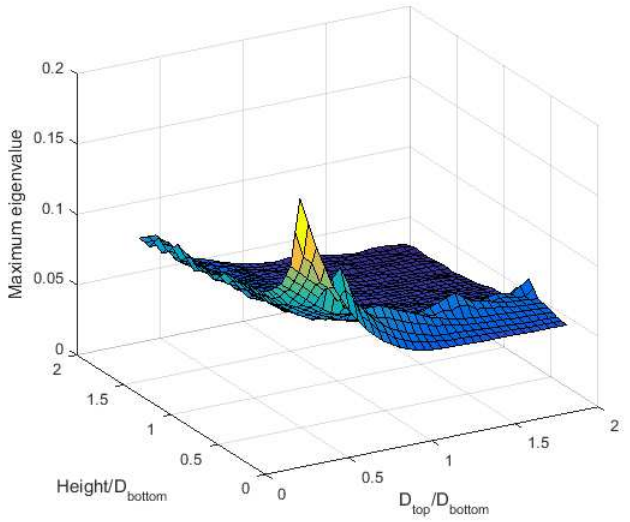

Figure 5.15: Maximum eigenvalue for 4-wire wrist

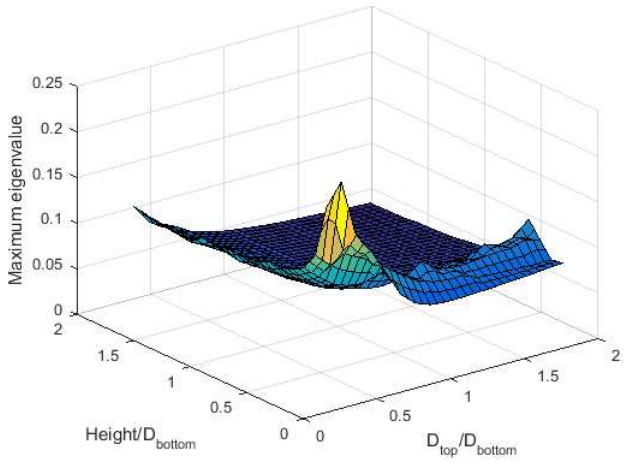

Figure 5.16: Maximum eigenvalue for 3-wire wrist

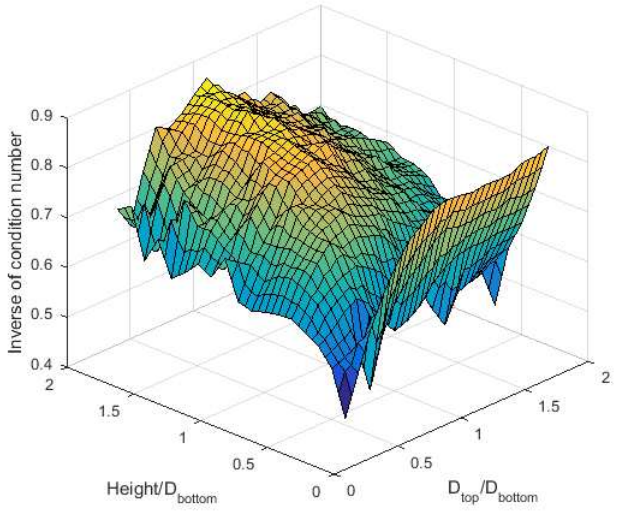

Figure 5.17: Inverse Condition Number for 4-wire wrist

cording to Klein and Blaho [32], condition number indicates the uniformity of the Jacobian transformation with respect to direction. That is to say, given external torque $\mathbf{w}_{ee}$ with certain magnitude, the more $\frac{1}{\kappa}$ approaches 1, the less τ_w's magnitude will change due to alteration of $\mathbf{w}_{ee}$'s direction. The value of $\frac{1}{\kappa}$ is very important to wire tensions' optimization because if $\frac{1}{\kappa}$ is very small, a tiny alteration of external wrench's direction, even though the magnitude stays the same, may cause great change in wires' tensions. We again scan the whole workspace for each ratio pair and get the average values of $\frac{1}{\kappa}$ as shown in figure 5.17 for 4-wire wrist and figure 5.18 for 3-wire wrist where higher values are preferred.

Moreover, in order to properly control the wrist within workspace, we need to find the "smallest" λ for each point that can guarantee tensions in wires. Since external wrench is given, I can get both specific and homogenous solutions using equation 5.2. The purpose is to compute "λ" to make the result of equation 5.3 positive. Thus if all of the first four/three elements of $\tilde{\tau_p}$ are positive, λ can be assigned 0; if not, we will figure out the values of λ for each wire and select the one with maximum absolute value which would be the "smallest" λ for that point within workspace. The procedure is listed in Table 5.1. An example of wire tensions is given when external torque is $10N \cdot mm$. Simulation results

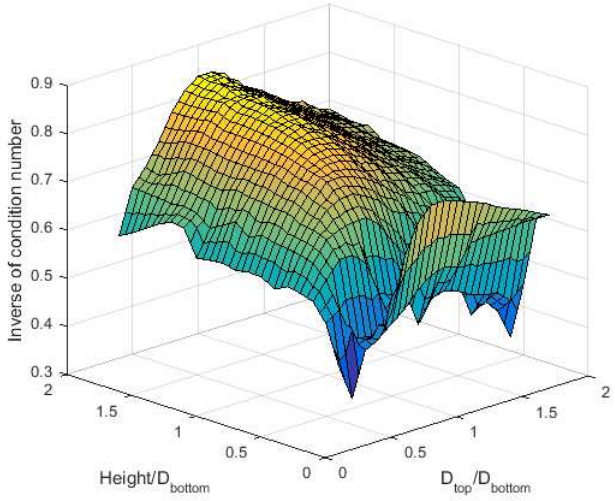

Figure 5.18: Inverse Condition Number for 3-wire wrist

Table 5.1: Calculation of minimum wire tensions and corresponding λ

If all 4 items of $\tilde{\tau}_p$ are positive	If not all $\tilde{\tau}_p$ are positive	
$\lambda = 0,$	if $\tilde{\tau}_p(i) < 0,$	if $\tilde{\tau}_p(i) \geq 0,$
$\tilde{\tau} = \tilde{\tau}_p$	$\lambda(i) = \frac{-\tilde{\tau}_p(i)}{nj(i)}$	$\lambda(i) = 0$
	$\lambda = max(\lambda(i))$ $(i = 1,2,3,4)$	

show that most tensions are within reasonable range except for several points sitting on the workspace edges near singular position. The maximum wire tension for 4-wire mechanism is around 600 N and for 3-wire wrist is around 800 N, as shown in figure 5.19 and 5.20.

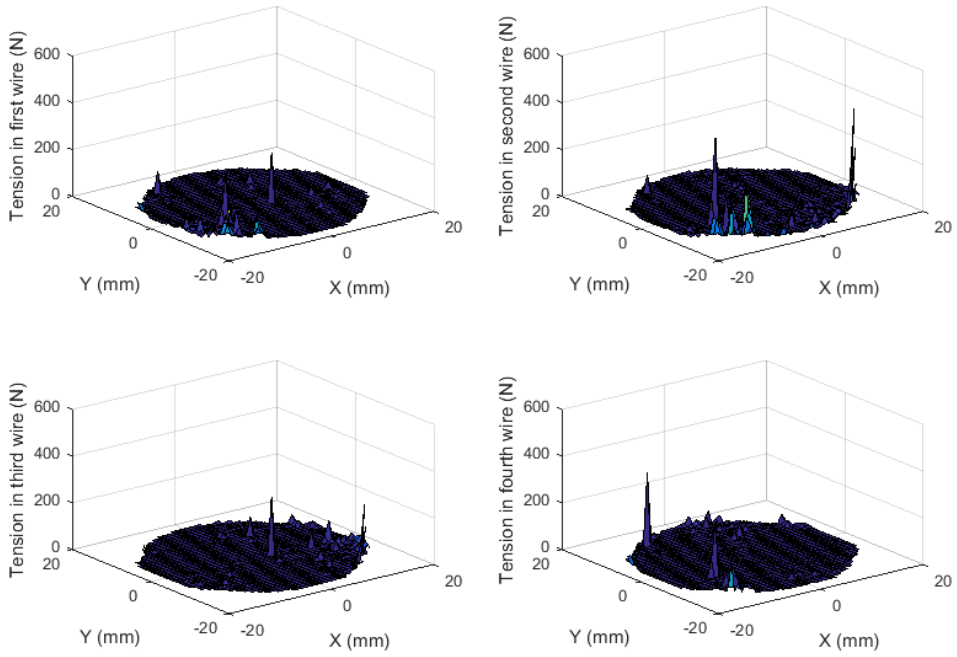

Figure 5.19: Example of wire tensions for 4-wire wrist within workspace when external moment is 10 $N \cdot mm$

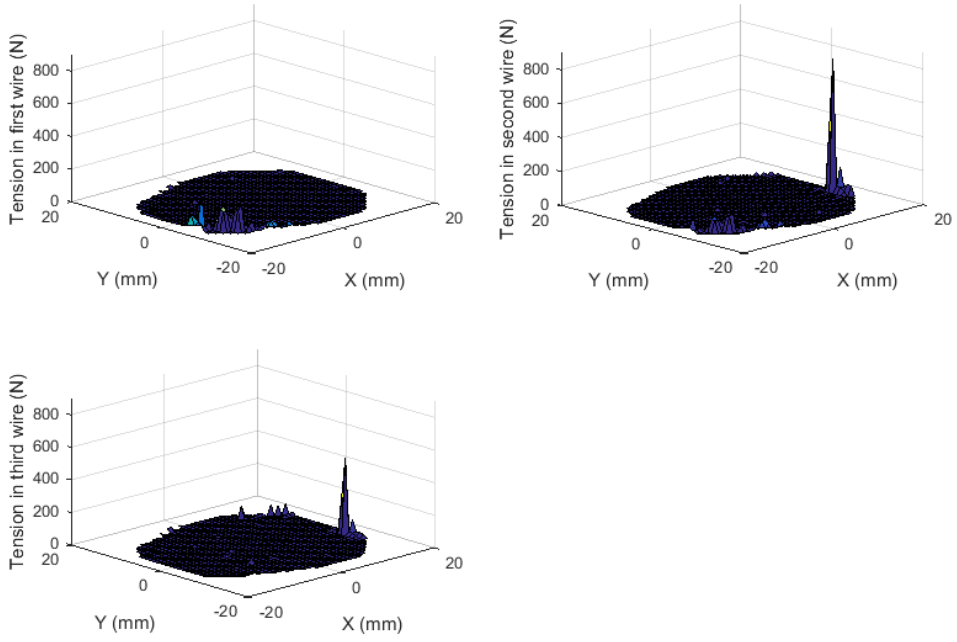

Figure 5.20: Example of wire tensions for 3-wire wrist within workspace when external moment is 10 $N \cdot mm$

Chapter 6

Experimental Validation

6.1 Experimental Setup

This chapter presents the design, fabrication and control of an experimental setup meant to validate our kinematic and static models.

6.1.1 Prototype Design and Manufacture

In this section, we manufactured a prototype of wire-actuated parallel wrist with universal joint for experiment. This mechanism which has 4 actuation wires is simplified compared with that in theory work that we did not add the 3rd rotation joint on it. The real photo of the experiment is shown in Figure 6.1 and Creo models of the setup can be seen in Figure 6.2, where the left figure is the whole setup for the wrist, including actuation parts, the base, the wrist and NDI trackers, and the right figure is an exploded view of the wrist mechanism. The actuation parts consist of four Velmex linear slides, Maxon motors and linear potentiometers. The Velmex slides driven by RE16 Maxon motors convert rotary motions into linear motions. The potentiometers can record the slides' position for operations such as joint control and homing. Actuation wires are connected to the spring fixed on Velmex slides so that the slides can control wire lengths directly. The reason why we use springs for connection is that springs can provide preloads for the wires. Moreover, the wires' stiffness are thus dominated by springs if $K_{wire} \gg K_{spring}$, as shown in Equation 6.1 and 6.2.

$$\begin{aligned} F_{ext} &= K_{wire}\Delta x_w = K_{spring}\Delta x_s \\ \Delta x_s &= \frac{K_{wire}}{K_{spring}}\Delta x_w \end{aligned} \tag{6.1}$$

Here F_{ext} is the force applied on the "wire-spring" system; K_{wire} and K_{spring} represent

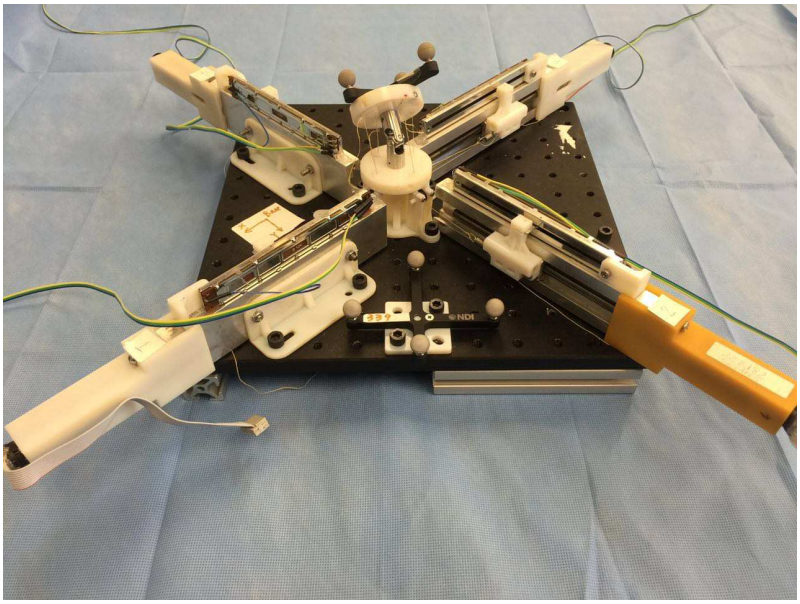

Figure 6.1: Experiment setup for wire-actuated wrist with universal joint

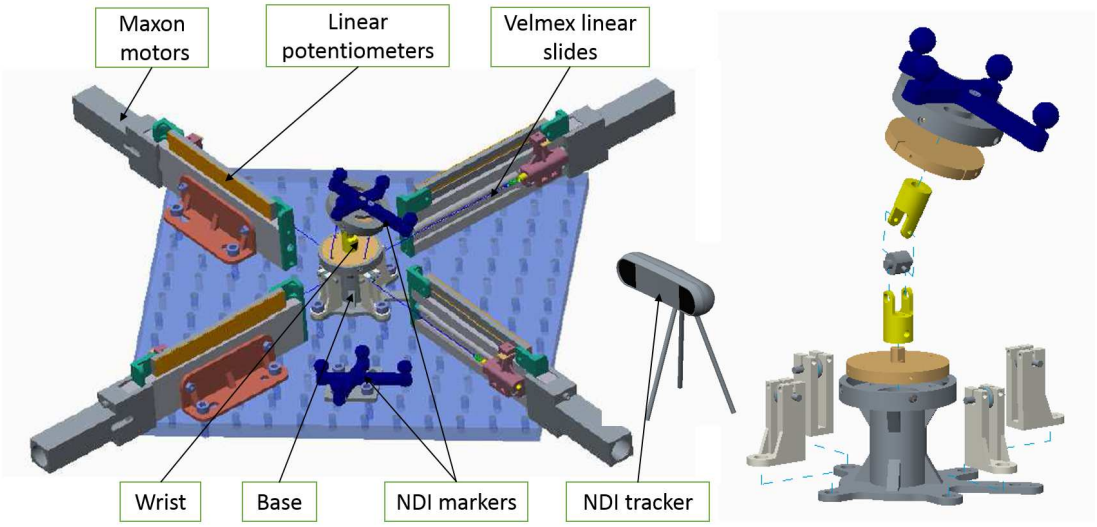

Figure 6.2: Creo Models of assembly for experiment setup (left) and exploded view of wire-actuated wrist with universal joint (right)

wire stiffness and spring stiffness respectively, while Δx_w and Δx_s are the displacements of the wire and spring. Thus, the final stiffness of the system K can be expressed as:

$$\begin{aligned} K_d &= \frac{F_{ext}}{\Delta x_s + \Delta x_w} \\ &= \frac{K_{wire}\Delta x_w}{\Delta x_w + \frac{K_{wire}}{K_{spring}}\Delta x_w} \\ &= \frac{K_{spring}}{1 + \frac{K_{spring}}{K_{wire}}} \end{aligned} \tag{6.2}$$

In our experiment setup, $K_{spring} = 1.72N/mm$. The Young's Modulus of teflon wire is $E = 0.5GPa$, the cross-sectional area $A = 0.0628mm^2$ and initial wire length $L_0 = 118mm$. Thus the stiffness of the wire is

$$K_{wire} = \frac{EA}{L_0} = 2.661 \times 10^5 N/mm \gg K_{spring} \tag{6.3}$$

According to 6.2, the stiffness of "wire-spring" connection is approximately equal to K_{spring}:

$$K \approx K_{spring} = 1.72N/mm \tag{6.4}$$

Moreover, the wrist is fixed on the base. Four pulleys are used to change wires' directions for convenience of control. Two NDI markers (Marker I and Marker II) and one optical measurement camera are used to track the position and orientation of end effector. Marker I is fixed in base as a reference marker, and Marker II which is installed on top plate can record the relative position and rotation from itself to Marker I.

6.1.2 Real-time Control Using MatLab xPC

The setup for the experiment is borrowed from project of Large Snake developed by Andrea Bajo and Long Wang, and the control code and stateflow in Simulink was designed by Long Wang and Nima Sarli. The control part is executed using MatLab xPC Target,

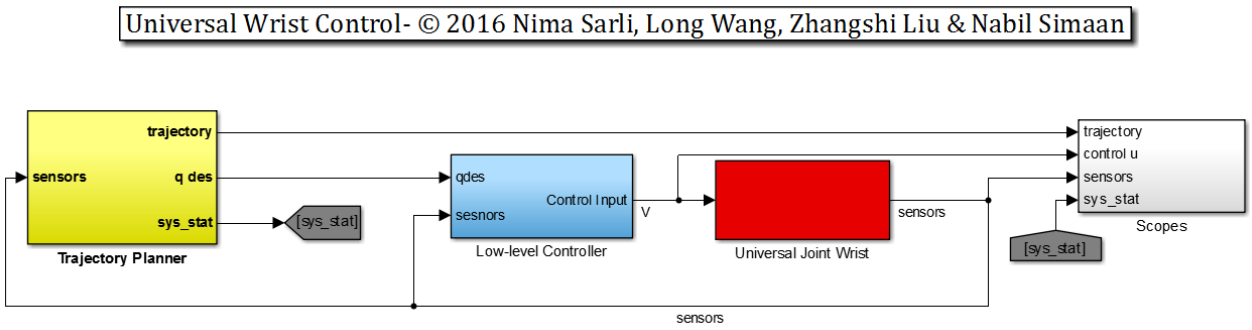

Figure 6.3: Stateflow of universal joint wrist controller

which is a host-target prototyping environment provided by MatLab. The reason why we use xPC Target is that we can implant the kinematics models into Simulink and Stateflow to enable rapid real-time testing.

The control system consists of a host machine and a target machine. The host machine is where the controller is built in Stateflow, implant kinematics model and modify the control parameters. The target machine is in charge of code execution and information communication with encoders, potentiometers and servo amplifiers, etc. Figure 6.3 shows the structure of the whole control system.

In the figure we can see the controller mainly consists of four blocks: Trajectory Planner, Low-level Controller, Universal Joint Wrist and Scopes. We will next make a brief illustration for each block:

- The trajectory planner is a high-level controller that process data from such as motor encodes, potentiometers and user-input desired end effector orientations, etc., to acquire desired joint values. Then use fifth order polynomial interpolation method to calculate real-time joint values and output them to the low-level controller. This block has 4 modes: mode 0, mode 1, mode 2 and mode 3. When mode 0 is activated, the robot will maintain the current joint configuration. Mode $=$ 1 is for homing procedure that moves the robot to a pre-defined homing position. If mode 2 is selected, we can control the joint space directly and thus it can be used to test direct kinemat-

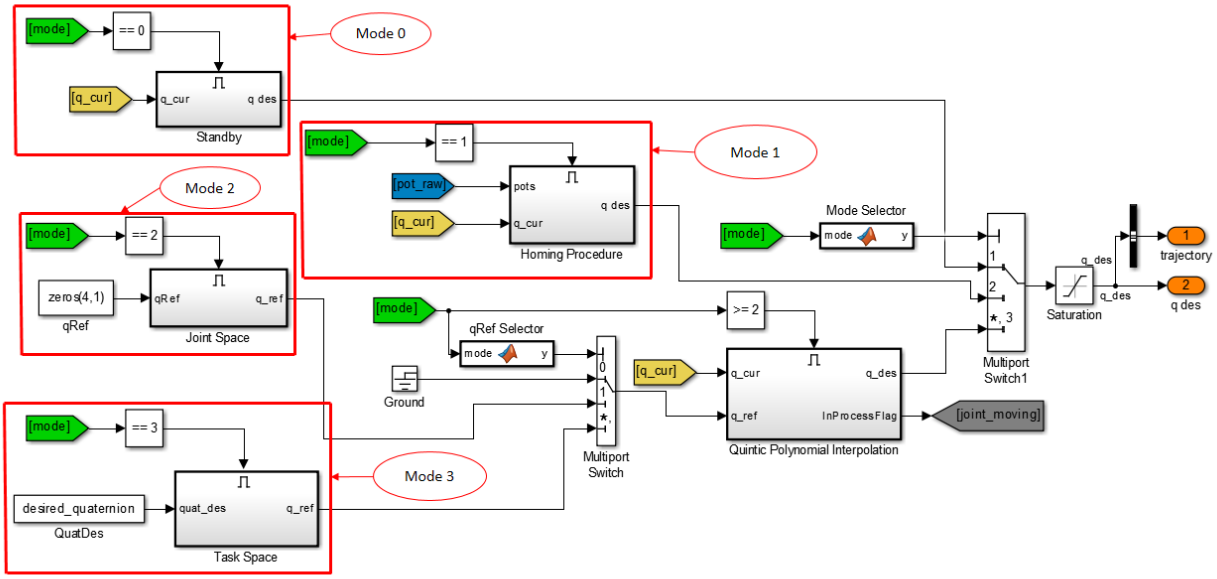

Figure 6.4: Structure of trajectory planner in controller

ics model. Moreover, mode 3 is for validation of inverse kinematics that desired end effector's orientation is imported into the "task space" block, and the block outputs corresponding calculated joint values. The scheme is shown in figure 6.4.

- The low-level Controller is the PID controller, which accepts the desired joint values and outputs the control signal.

- The Universal Joint Wrist block in Figure 6.3 acts like an interface between controllers and target machine. It transfers the control signal from Low-level Controller to D/A card. At the same time, it receives digitalized encoders and potentiometers' signals as inputs for controllers.

- The last block "Scopes" shows the current status of joint values, control signals, sensors, etc. on one computer screen.

Moreover, we also make a MatLab GUI for the control part of the experiment in figure 6.5.

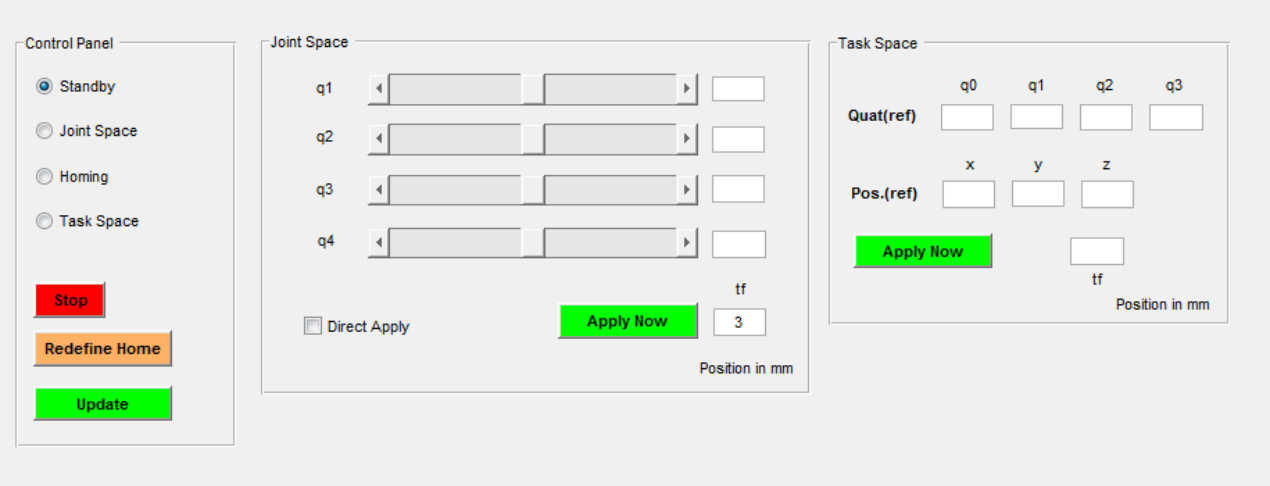

Figure 6.5: MatLab GUI for control system

6.2 Validation of Inverse Kinematics

In this section, we will test the inverse kinematics of the wrist. The input is a given orientation of end effector in the form of quaternion as obtained from an optical tracker. The controller will output the corresponding motor control signals to the amplifiers to drive the motors. At the same time, signals from the encoders and potentiometers which record joints' positions and velocities will be sent back to xPC controller as feedback. Finally, when the feedback shows that the actual joint values equal the desired ones, we compare the Theoretical wire lengths and actual wire lengths measured by caliper. The procedures can be shown in figure 6.6.

The results are shown in Table 6.1 and plotted in Figure 6.7.

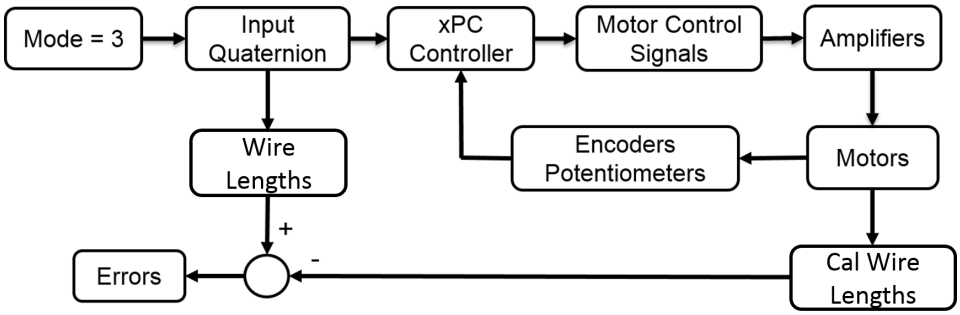

Figure 6.6: Procedures for experiment of inverse kinematics validation

Table 6.1: Experiment Results of Inverse kinematics (unit: mm)

	Quaternion	Theoretical Wire Lens	Actual Wire Lens	Errors
1	[0.884, -0.268, 0.136, -0.358]	[43.1, 47.33, 26.14, 22.05]	[42.55, 47.57, 24.75, 20.82]	[-0.55, 0.24, -1.39, -1.23]
2	[0.978, -0.208, 0.010, 0.007]	[36.38, 37.80, 38.21, 36.80]	[36.53, 37.20, 37.25, 37.31]	[0.15, -0.60, -0.96, 0.51]
3	[0.921, -0.161, 0.284, 0.215]	[20.82, 39.60, 48.58, 29.80]	[19.82, 39.33, 48.41, 28.96]	[-1.00, -0.27, -0.17, -0.84]
4	[0.913, -0.234, 0.191, -0.274]	[39.66, 47.89, 30.65, 22.48]	[39.36, 48.64, 28.68, 21.09]	[-0.30, 0.75, -1.97, -1.39]
5	[0.930, -0.222, -0.045, -0.291]	[45.98, 41.04, 25.52, 30.48]	[46.15, 41.07, 25.55, 29.38]	[0.17, 0.03, 0.03, -1.10]
6	[0.960, -0.183, -0.160, -0.138]	[44.23, 34.60, 28.90, 38.52]	[44.11, 32.80, 27.17, 38.68]	[-0.12, -1.80, -1.73, 0.16]
7	[0.887, -0.255, 0.170, -0.346]	[41.86, 48.13, 27.13, 21.00]	[40.97, 48.55, 26.01, 19.79]	[-0.89, 0.42, -1.12, -1.21]
8	[0.941, -0.150, 0.298, 0.053]	[26.63, 44.41, 44.23, 26.44]	[26.69, 44.93, 43.54, 24.75]	[0.06, 0.52, -0.69, -1.69]
9	[0.959, -0.177, 0.222, -0.014]	[31.35, 44.24, 41.16, 28.27]	[31.67, 45.34, 40.34, 27.10]	[0.32, 1.10, -0.82, -1.17]
10	[0.967, -0.186, -0.129, -0.118]	[43.14, 35.43, 30.47, 38.18]	[43.65, 34.13, 28.90, 38.67]	[0.51, -1.30, -1.57, 0.49]

Table 6.2: RMS Error of Inverse Kinematics (unit: mm)

	Wire 1	Wire 2	Wire 3	Wire 4	Overall
RMSE	0.5115	0.8741	1.2093	1.0759	0.9547

6.3 Validation of Stiffness Model

In this section, we will verify the stiffness model of the wrist. Since we cannot use experiments to acquire stiffness/compliance matrix directly, the end effector's displacement will be used to evaluate the model. The inputs for this part are initial end effector's position and orientation, as well as external moment applied on the wrist, and the output is the end effector's position after deflection. The external moment is produced by a 500g weight applied on a fixed point on top plate.

The detailed procedures are illustrated in Figure 6.8 First of all, before adding the weight to the system, we record the position and orientation of end effector. Then apply the moment and get the new end effector position by processing data from NDI tracker. At the same time, we use MatLab to compute the theoretical results according to previous theory work. Finally, we compare the theoretical results and experiment results in Table 6.3 and in Figure 6.9.

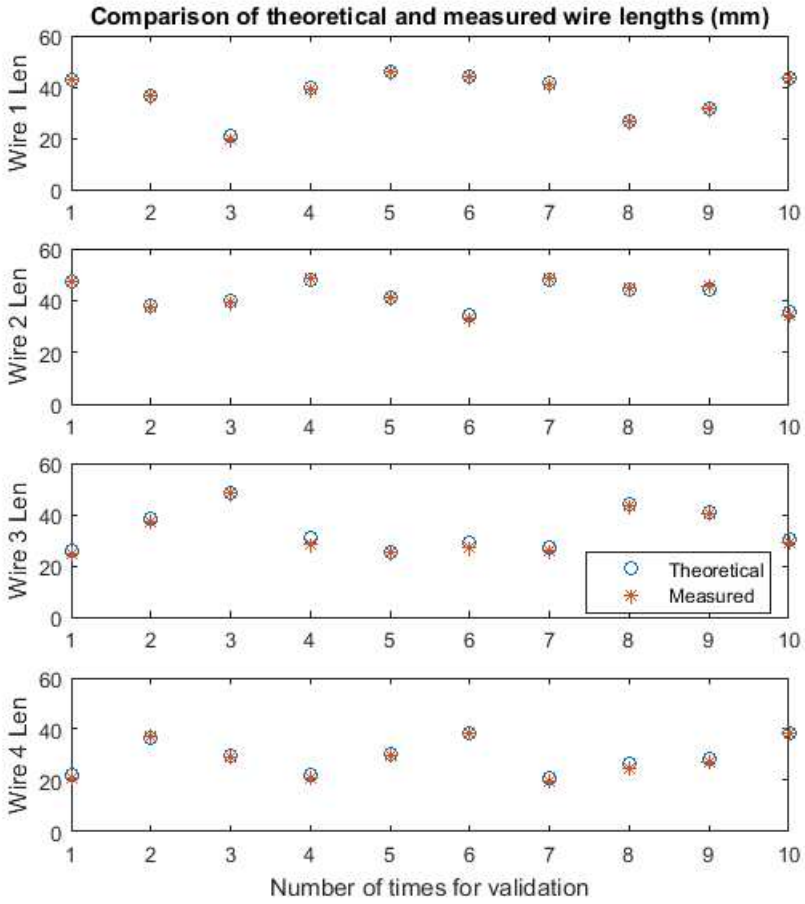

Figure 6.7: Comparison of wire lengths for inverse kinematics validation

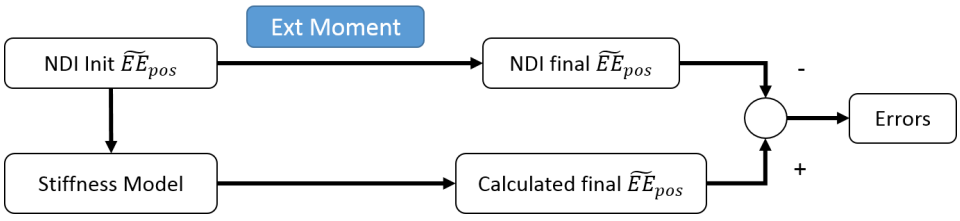

Figure 6.8: Procedures for experiment of stiffness validation

Table 6.3: Comparison of simulation and experiment results for stiffness validation (unit: mm)

	Initial EE_{pos}	NDI EE_{pos}	Cal EE_{pos}	EE_{pos} Error
1	[3.6747, 9.5413, 33.0947]	[3.7313, 8.4284, 33.7425]	[5.0209, 9.2065, 34.1735]	[1.2896, 0.7781, 0.4310]
2	[-1.7313, 7.9118, 34.7244]	[-1.1563, 5.9023, 35.8445]	[-1.0169, 7.2460, 35.8470]	[0.1394, 1.3438, 0.0025]
3	[-7.8545, 7.5485, 3.6678]	[-7.3012, 6.2523, 34.5697]	[-7.7788, 6.7711, 34.2859]	[-0.4776, 0.5188, -0.2839]
4	[-8.8840, 4.6238, 34.7273]	[-8.4469, 3.4470, 35.0684]	[-8.7365, 3.2202, 34.8840]	[-0.2896, -0.2268, -0.1844]
5	[-6.8048, -4.3606, 35.3499]	[-6.9810, -2.8768, 35.6112]	[-7.1549, -1.9410, 35.8068]	[-0.1738, 0.9357, 0.1956]
6	[-8.8843, -6.3621, 33.6760]	[-9.0284, -4.8673, 34.1410]	[-8.9968, -3.9617, 34.5842]	[0.0316, 0.9056, 0.4433]
7	[-9.1245, -9.1724, 32.2234]	[-9.2246, -7.9415, 32.7559]	[-9.0422, -6.8066, 33.5977]	[0.1824, 1.1349, 0.8418]
8	[-4.0902, -12.2696, 32.3223]	[-4.1633, -10.9862, 33.1506]	[-4.5135, -10.0520, 3.8123]	[-0.3501, 0.9342, 0.6617]

Table 6.4: RMS Error of Stiffness Model (unit: mm)

Position	x	y	z	Overall
RMSE	0.5222	0.9075	0.4585	0.6599

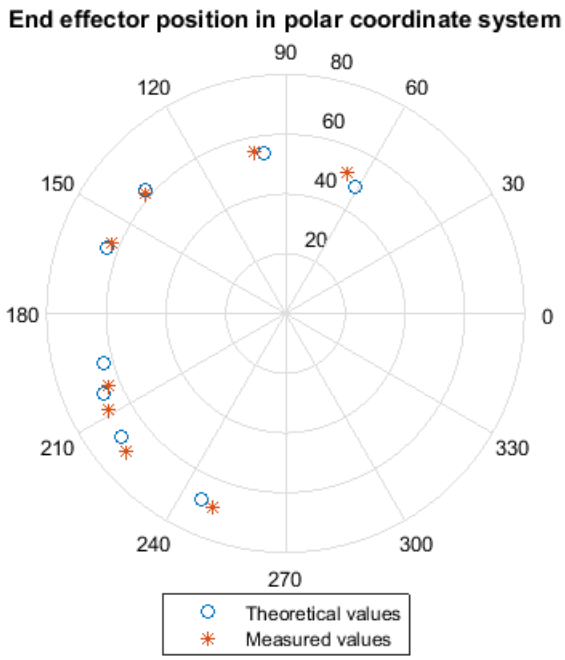

Figure 6.9: Comparison of end effector positions for stiffness validation (Unit: degree)

6.4 Experiment Conclusion

The inverse kinematics experiment data shows that the actual wire lengths match the theoretical values well. The mean error is 0.5mm, which is about 1.4% of average wire length of 35.72mm. The maximum error is -1.97mm, the 3^{rd} wire in the 4^{th} orientation, which is about 6.4% of its corresponding reference value 30.65mm. The main reason for this greater error is probably the deflection during measurements using caliper because the springs connected to wires are easy to change lengths. Moreover, some measuring positions are difficult to be reached by caliper which may also have negative effects on the accuracy. Last, there are still some other reasons such as manufacturing errors, measurement errors, etc.

The data of end effector position acquired in stiffness experiment also matches the theoretical values well. The mean error is 0.6527mm, while the maximum error is 1.56mm in y of the 6^{th} position. Besides the deflection that we have discussed in inverse kinematics experiment, another main reason for the error is that there may be some error when we

calculate the norm and direction of the applied external moment.

Generally speaking, the experiments have demonstrated the inverse kinematics model and stiffness model, which also indirectly prove that the Jacobian calculation is correct.

Chapter 7

Conclusion

This thesis presented an investigation into the modeling of kinematics, statics and wrench closure workspace for wire actuated parallel robots with a constraint leg comprised of a universal joint. The concept of wrench closure workspace has been known in the literature of wire actuated robots. Generally, the modeling frameworks do not account for wrench closure workspace restrictions due to wire extension. This thesis has built on an earlier exploration of the concept of wrench closure of wire actuated robots with elastic actuation wires (Hamid and Simaan [28]). The thesis has presented instantaneous kinematics modeling frameworks using virtual work principle and using loop closure constraint method. Both inverse and direct kinematics of wire actuated universal joint wrists with a revolute joint at the base or at the moving platform have been modeled and validated by simulation. The inverse kinematics method has been validated also by experiments. A model of the stiffness of these wrists has been presented based on prior art in the literature of parallel robots [29]. This model has been used to define the wrench closed workspace while accounting for maximal deflections subject to a norm-bounded load on the wrist. The method relied on the use of the compliance matrix of the hybrid robot comprised of a parallel two degrees of freedom wrist attached in series to a revolute joint. Using singular value decomposition of a sub-matrix of the overall Jacobian of the hybrid robot we were able to define the safe workspace boundaries of the wrench closure workspace such that even when the wrist deflects due to external norm-bounded force the requirement of wrench closure is still maintained. The analysis also compared the effect of using three or four actuation wires on the kinematics, statics, wrench closure workspace and stiffness. Results suggest that using four wires provide one degree of actuation redundancy that can be explored for enhancing stiffness and for enlarging the wrench closure workspace. These results can help

guide the design of wire actuated parallel robots and surgical parallel wrists.

BIBLIOGRAPHY

[1] Russell H. Taylor. A perspective on medical robotics, 2006.

[2] a.J. Madhani, G. Niemeyer, and Jr. Salisbury, J.K. The Black Falcon: a teleoperated surgical instrument for minimally invasive surgery. *Proceedings. 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications (Cat. No.98CH36190)*, 2(October):936–944, 1998.

[3] Andrea Bajo, Roger E. Goldman, Long Wang, Dennis Fowler, and Nabil Simaan. Integration and preliminary evaluation of an Insertable Robotic Effectors Platform for Single Port Access Surgery. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3381–3387, 2012.

[4] Jienan Ding, Roger E Goldman, Kai Xu, Peter K. Allen, Dennis L. Fowler, and Nabil Simaan. Design and Coordination Kinematics of an Insertable Robotic Effectors Platform for Single-Port Access Surgery. *IEEE/ASME Transactions on Mechatronics*, 18(5):1612–1624, 2013.

[5] Jusuk Lee, Jiyoung Kim, Kwang-kyu Lee, Seungyong Hyung, Yong-jae Kim, Woong Kwon, Kyungshik Roh, and Jung-yun Choi. Modeling and control of robotic surgical platform for single-port access surgery. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, number Iros, pages 3489–3495, 2014.

[6] Johannes Bodner, Florian Augustin, Heinz Wykypiel, John Fish, Gilbert Muehlmann, Gerold Wetscher, and Thomas Schmid. The da Vinci robotic system for general surgical applications: A critical interim appraisal. *Swiss Medical Weekly*, 135(45-46):674–678, 2005.

[7] M. Minor and R. Mukherjee. A Mechanism for Dexterous End-Effector Placement During Minimally Invasive Surgery, 1999.

[8] Yong Jae Kim, Shanbao Cheng, Sangbae Kim, and Karl Iagnemma. A Stiffness-Adjustable Hyperredundant Manipulator Using a Variable Neutral-Line Mechanism for Minimally Invasive Surgery, 2013.

[9] Paul Breedveld and Shigeo Hirose. Design of Steerable Endoscopes to Improve the Visual Perception of Depth During Laparoscopic Surgery, 2004.

[10] Chi Min Seow, Wei Jian Chin, Carl a. Nelson, Akiko Nakamura, Shane M. Farritor, and Dmitry Oleynikov. Articulated Manipulator With Multiple Instruments for Natural Orifice Transluminal Endoscopic Surgery. *Journal of Medical Devices*, 7(4):041004, 2013.

[11] Kanako Harada, Kota Tsubouchi, Masakatsu G. Fujie, and Toshio Chiba. Micro manipulators for intrauterine fetal surgery in an open MRI. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2005, pages 502–507, 2005.

[12] J.-P. Merlet. Optimal design for the micro parallel robot MIPS. *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, 2(May), 2002.

[13] Ryoichi Nakamura, Etsuko Kobayashi, Ken Masamune, and Ichiro Sakuma. Multi-DOF Forceps Manipulator System for Laparoscopic Surgery. In *Medical Image Computing and Computer-Assisted Intervention MICCAI 2000*, pages 653–660. 2000.

[14] Shorya Awtar, Tristan T. Trutna, Jens M. Nielsen, Rosa Abani, and James Geiger. FlexDex: A Minimally Invasive Surgical Tool With Enhanced Dexterity and Intuitive Control, 2010.

[15] Hiroki Takahashi, Shin'ichi Warisawa, Mamoru Mitsuishi, Jumpei Arata, and Makoto Hashizume. Development of high dexterity minimally invasive surgical system

with augmented force feedback capability. In *Proceedings of the First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006, BioRob 2006*, volume 2006, pages 284–289, 2006.

[16] G.S. Guthart and Jr. Salisbury, J.K. The IntuitiveTM telesurgery system: overview and application. *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, 1(April), 2000.

[17] Kotaro Tadano and Kenji Kawashima. Development of 4-DOFs forceps with force sensing using pneumatic servo system. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2006, pages 2250–2255, 2006.

[18] N. Simaan, R. Taylor, and P. Flint. A dexterous system for laryngeal surgery. *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, 1:351–357, 2004.

[19] Purang Abolmaesumi, Gabor Fichtinger, Terry M. Peters, Ichiro Sakuma, and Guang-Zhong Yang. Introduction to special section on surgical robotics. *IEEE transactions on bio-medical engineering*, 60(4):887–91, 2013.

[20] Russell H. Taylor and Dan Stoianovici. Medical Robotics in Computer-Integrated Surgery, 2003.

[21] John a. Kaufman, Jim a. Reekers, James P. Burnes, Aghiad Al-Kutoubi, Curtis a. Lewis, Brian W. Hardy, Sachio Kuribayashi, and Sanjiv Sharma. Global statement defining interventional radiology. *Journal of Interventional Radiology*, 19(8):593–597, 2010.

[22] M B Cohn, Lara S Crawford, J M Wendlandt, and S S Sastry. Surgical applications of milli-robots. *J. Robotic Systems*, 12(6):401–416, 1995.

[23] Valentina Vitiello, Su Lin Lee, Thomas P. Cundy, and Guang Zhong Yang. Emerging robotic platforms for minimally invasive surgery. *IEEE Reviews in Biomedical Engineering*, 6(5):111–126, 2013.

[24] Marc Gouttefarde and Clément M. Gosselin. Analysis of the wrench-closure workspace of planar parallel cable-driven mechanisms. *IEEE Transactions on Robotics*, 22(3):434–445, 2006.

[25] Nabil Simaan. Snake-like units using flexible backbones and actuation redundancy for enhanced miniaturization. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 2005, pages 3012–3017, 2005.

[26] Filip Jelínek, Ewout a. Arkenbout, Paul W. J. Henselmans, Rob Pessers, and Paul Breedveld. Classification of Joints Used in Steerable Instruments for Minimally Invasive SurgeryA Review of the State of the Art. *Journal of Medical Devices*, 9(1):010801, 2015.

[27] Lung-Wen Tsai. *Robot Analysis: The Mechanics of Serial and Parallel Manipulators*. John Wiley and Sons, Inc, 1999.

[28] Saleem Abdul Hamid and Nabil Simaan. Design and synthesis of wire-actuated universal-joint wrists for surgical applications. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1807–1813, 2009.

[29] N. Simaan and M. Shoham. Geometric Interpretation of the Derivatives of Parallel Robots Jacobian Matrix With Application to Stiffness Control. *Journal of Mechanical Design*, 125(March):33, 2003.

[30] W. B. Lim, S. H. Yeo, G. Yang, and I. M. Chen. Design and analysis of a cable-driven manipulator with variable stiffness. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 4519–4524, 2013.

[31] Rg Roberts, Todd Graham, and Thomas Lippitt. On the inverse kinematics, statics, and fault tolerance of cablesuspended robots. *Journal of Robotic Systems*, 15(10):581–597, 1998.

[32] Charles a Klein and Bruce E Blaho. Dexterity Measures for the Design and Control of Kinematically Redundant Manipulators. *The International Journal of Robotics Research*, 6(2):72–83, 1987.

APPENDIX

Chapter 8

Appendix

8.1 MATLAB Code for Inverse Kinematics

8.1.1 Inverse Kinematics for 3-Wire Mechanism

```
% This code is to verify inverse kinematics 3 wires case 1
% Inputs: EEpos and R03
% Outputs: l1 l2 l3 and alpha1
% 20160127 Zhangshi Liu
clc;
clear;
close all;
initialization_3wires_case1_inverse;    %initialization
n1 = length(theta1);
n2 = length(theta2);
n3 = length(alpha1);

theta1_cal_set = [];    %set for storing calculated values for theta1
theta2_cal_set = [];    %set for storing calculated values for theta2
alpha1_cal_set = [];    %set for storing calculated values for alpha1
wire_lengths = [];      %set for storing reference values for wire lengths
wire_lengths_cal = [];  %set for storing calculated values for wire lengths

for i1 = 1: n1
    for i2 = 1: n2
        for i3 = 1: n3
```

```
% Use configuration Direct kinematics to calculate R03
R03 = [cos(alpha1(i3))*cos(theta2(i2)) - sin(alpha1(i3))...
    *sin(theta1(i1))*sin(theta2(i2)),...
    -sin(alpha1(i3))*cos(theta1(i1)),...
    cos(alpha1(i3))*sin(theta2(i2))...
    + sin(alpha1(i3))*cos(theta2(i2))*sin(theta1(i1));
       sin(alpha1(i3))*cos(theta2(i2))...
       + cos(alpha1(i3))*sin(theta1(i1))*sin(theta2(i2)),...
       cos(alpha1(i3))*cos(theta1(i1)),...
       sin(alpha1(i3))*sin(theta2(i2))...
       - cos(alpha1(i3))*cos(theta2(i2))*sin(theta1(i1));
       -cos(theta1(i1))*sin(theta2(i2)),...
       sin(theta1(i1)), cos(theta1(i1))*cos(theta2(i2))];
R01 = rotr([0;0;1],alpha1(i3));

% vectors of the wires
l1_in0 = R01*t1_in1 + R03*t2_in3 + R03*a1_in3 - R01*b1_in1;
l2_in0 = R01*t1_in1 + R03*t2_in3 + R03*a2_in3 - R01*b2_in1;
l3_in0 = R01*t1_in1 + R03*t2_in3 + R03*a3_in3 - R01*b3_in1;
% norms of wires
l1 = norm(l1_in0);
l2 = norm(l2_in0);
l3 = norm(l3_in0);

wire_lengths = [wire_lengths, [l1, l2, l3]'];

% Use Inverse kinematics Model to calculate theta1_cal,
% theta2_cal and alpha1_cal
theta1_cal = asin(R03(3,2));
theta1_cal_set = [theta1_cal_set, theta1_cal];
theta2_cal = asin(-R03(3,1)/cos(theta1_cal));
theta2_cal_set = [theta2_cal_set, theta2_cal];
sin_alpha1 = -R03(1,2)/cos(theta1_cal);
```

```
            cos_alpha1 = R03(2,2)/cos(theta1_cal);
            alpha1_cal = atan2(sin_alpha1, cos_alpha1);
            alpha1_cal_set = [alpha1_cal_set, alpha1_cal];
        end
    end
end
%store all the calculated angle in one set
angles_cal_set = [theta1_cal_set; theta2_cal_set; alpha1_cal_set];
[m,n] = size(angles_cal_set);
for i = 1: n
    % use configuration inverse kinematics to calculate wire lengths
    R01_cal = rotr([0;0;1],angles_cal_set(3,i));
    R12_cal = rotr([1;0;0],angles_cal_set(1,i));
    R23_cal = rotr([0;1;0],angles_cal_set(2,i));
    R03_cal = R01_cal*R12_cal*R23_cal;

    l1_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a1_in3...
        - R01_cal*b1_in1;
    l2_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a2_in3...
        - R01_cal*b2_in1;
    l3_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a3_in3...
        - R01_cal*b3_in1;

    l1_cal = norm(l1_in0_cal);
    l2_cal = norm(l2_in0_cal);
    l3_cal = norm(l3_in0_cal);
    % store calculated wire lengths in a set
    wire_lengths_cal = [wire_lengths_cal, [l1_cal, l2_cal, l3_cal]'];
end
fig = 1;
% draw comparison result of reference wire lengths and calculated wire lengths
drawwirelengths_3wires(wire_lengths, wire_lengths_cal,fig)
```

```
% Initialization for verification 3 wire case 1 inverse kinematics
% Zhangshi 20160115

% set limits for angles
angle_max = pi/3;
angle_min = -angle_max;
% set steps within angle limits
step1 = pi/10;
step2 = pi/10;

theta1 = angle_min:step1:angle_max;
theta2 = angle_min:step2:angle_max;

alpha1 = pi/2;
alpha2 = pi/2;

angle = 120/180*pi; %wire distribution on plates.
h = 19; %hook height
r1 = 18;    %bottom radius
r2 = 18;    %top radius

t1_in1 = [0;0;h];   % vector t1
t2_in3 = [0;0;h];   % vector t2

a1_in3 = [r2;0;0];  % position of wire 1 on top plate
a2_in3 = [r2*cos(angle);r2*sin(angle);0];   %position of wire 2 on top plate
a3_in3 = [r2*cos(-angle);r2*sin(-angle);0]; %position of wire 3 on top plate

b1_in1 = [r1;0;0]; % position of wire 1 on bottom plate
b2_in1 = [r1*cos(angle);r1*sin(angle);0];   % position of wire 2 on bottom plate
b3_in1 = [r1*cos(-angle);r1*sin(-angle);0]; % position of wire 3 on bottom plate
```

```
function drawwirelengths_3wires(wire_lengths, wire_lengths_cal, fig)
    [m,n] = size(wire_lengths);
    figure(fig)

    % first wire
    subplot(3,1,1);
    x = 1:n ;
    plot(x, wire_lengths(1,:),'O'); % use O to present reference values
    hold on
    plot(x, wire_lengths_cal(1,:),'*'); % use * to present calculated values
    ylim([0,60])
    ylabel('Wire 1 Len')
    title('Comparison of reference and calculated wire lengths (mm)')
    % second wire
    subplot(3,1,2);
    plot(x, wire_lengths(2,:),'O');
    hold on
    plot(x, wire_lengths_cal(2,:),'*');
    ylabel('Wire 2 Len')
    % third wire
    subplot(3,1,3);
    plot(x, wire_lengths(3,:),'O');
    hold on
    plot(x, wire_lengths_cal(3,:),'*');
    ylabel('Wire 3 Len')
    legend('Reference','Calculated');
    xlabel('Number of times for validation')

end
```

8.1.2 Inverse Kinematics for 4-Wire Mechanism

```
% This code is to verify inverse kinematics 4 wires case 1
% Inputs: EEpos and R03
% Outputs: l1 l2 l3 l4 and alpha1
% 20160127 Zhangshi Liu
clc;
clear;
close all;
initialization_4wires_case1_inverse;    %initialization
n1 = length(theta1);
n2 = length(theta2);
n3 = length(alpha1);

theta1_cal_set = [];    % set for storing calculated values for theta1
theta2_cal_set = [];    % set for storing calculated values for theta2
alpha1_cal_set = [];    % set for storing calculated values for alpha1
wire_lengths = [];      % set for storing reference values for wire lengths
wire_lengths_cal = [];  % set for storing calculated values for wire lengths

for i1 = 1: n1
    for i2 = 1: n2
        for i3 = 1: n3


            % Use configuration Direct kinematics to calculate R03
            R03 = [cos(alpha1(i3))*cos(theta2(i2))...
                - sin(alpha1(i3))*sin(theta1(i1))*sin(theta2(i2)),...
                -sin(alpha1(i3))*cos(theta1(i1)),...
                cos(alpha1(i3))*sin(theta2(i2))...
                + sin(alpha1(i3))*cos(theta2(i2))*sin(theta1(i1));
                   sin(alpha1(i3))*cos(theta2(i2))...
                   + cos(alpha1(i3))*sin(theta1(i1))*sin(theta2(i2)),...
                   cos(alpha1(i3))*cos(theta1(i1)),...
                   sin(alpha1(i3))*sin(theta2(i2))...
```

```
                - cos(alpha1(i3))*cos(theta2(i2))*sin(theta1(i1));
                -cos(theta1(i1))*sin(theta2(i2)),...
                sin(theta1(i1)), cos(theta1(i1))*cos(theta2(i2))];
            R01 = rotr([0;0;1],alpha1(i3));

            % vectors of the wires
            l1_in0 = R01*t1_in1 + R03*t2_in3 + R03*a1_in3 - R01*b1_in1;
            l2_in0 = R01*t1_in1 + R03*t2_in3 + R03*a2_in3 - R01*b2_in1;
            l3_in0 = R01*t1_in1 + R03*t2_in3 + R03*a3_in3 - R01*b3_in1;
            l4_in0 = R01*t1_in1 + R03*t2_in3 + R03*a4_in3 - R01*b4_in1;
            % norms of wires
            l1 = norm(l1_in0);
            l2 = norm(l2_in0);
            l3 = norm(l3_in0);
            l4 = norm(l4_in0);

            wire_lengths = [wire_lengths, [l1, l2, l3, l4]'];

            % Use Inverse kinematics Model to calculate theta1_cal,
            % theta2_cal and alpha1_cal
            theta1_cal = asin(R03(3,2));
            theta1_cal_set = [theta1_cal_set, theta1_cal];
            theta2_cal = asin(-R03(3,1)/cos(theta1_cal));
            theta2_cal_set = [theta2_cal_set, theta2_cal];
            sin_alpha1 = -R03(1,2)/cos(theta1_cal);
            cos_alpha1 = R03(2,2)/cos(theta1_cal);
            alpha1_cal = atan2(sin_alpha1, cos_alpha1);
            alpha1_cal_set = [alpha1_cal_set, alpha1_cal];
        end
    end
end
%store all the calculated angle in one set
angles_cal_set = [theta1_cal_set; theta2_cal_set; alpha1_cal_set];
```

```
[m,n] = size(angles_cal_set);
for i = 1: n
    % use configuration inverse kinematics to calculate wire lengths
    R01_cal = rotr([0;0;1],angles_cal_set(3,i));
    R12_cal = rotr([1;0;0],angles_cal_set(1,i));
    R23_cal = rotr([0;1;0],angles_cal_set(2,i));
    R03_cal = R01_cal*R12_cal*R23_cal;
    l1_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a1_in3...
        - R01_cal*b1_in1;
    l2_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a2_in3...
        - R01_cal*b2_in1;
    l3_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a3_in3...
        - R01_cal*b3_in1;
    l4_in0_cal = R01_cal*t1_in1 + R03_cal*t2_in3 + R03_cal*a4_in3...
        - R01_cal*b4_in1;
    l1_cal = norm(l1_in0_cal);
    l2_cal = norm(l2_in0_cal);
    l3_cal = norm(l3_in0_cal);
    l4_cal = norm(l4_in0_cal);
    % store calculated wire lengths in a set
  wire_lengths_cal = [wire_lengths_cal, [l1_cal, l2_cal, l3_cal, l4_cal]'];
end
fig = 1;
% draw comparison result of reference wire lengths and calculated wire lengths
drawwirelengths(wire_lengths, wire_lengths_cal,fig)


% Initialization for verification 4 wire case 1 direct kinematics
% Zhangshi 20160115
angle_max = pi/3;
angle_min = -angle_max;
step1 = pi/10;
step2 = pi/10;
```

```
theta1 = angle_min:step1:angle_max;
theta2 = angle_min:step2:angle_max;

% alpha1 = 0:pi/2:pi;
alpha1 = pi/2;
alpha2 = 0;

h = 19;
r1 = 18;
r2 = 18;

t1_in1 = [0;0;h];
t2_in3 = [0;0;h];

a1_in3 = [r2;0;0];
a2_in3 = [0;r2;0];
a3_in3 = [-r2;0;0];
a4_in3 = [0;-r2;0];

b1_in1 = [r1;0;0];
b2_in1 = [0;r1;0];
b3_in1 = [-r1;0;0];
b4_in1 = [0;-r1;0];


function drawwirelengths(wire_lengths, wire_lengths_cal, fig)
    [m,n] = size(wire_lengths);
    fig = fig + 1;
    figure(fig)

    subplot(4,1,1);
    x = 1:n ;
```

```
    plot(x, wire_lengths(1,:),'O');
    hold on
    plot(x, wire_lengths_cal(1,:),'*');
    ylim([0,60])
    ylabel('Wire 1 Len')
    title('Comparison of reference and calculated wire lengths (mm)')

    subplot(4,1,2);
    plot(x, wire_lengths(2,:),'O');
    hold on
    plot(x, wire_lengths_cal(2,:),'*');
    ylim([0,60])
    ylabel('Wire 2 Len')


    subplot(4,1,3);
    plot(x, wire_lengths(3,:),'O');
    hold on
    plot(x, wire_lengths_cal(3,:),'*');
    ylim([0,60])
    ylabel('Wire 3 Len')


    subplot(4,1,4);
    plot(x, wire_lengths(4,:),'O');
    hold on
    plot(x, wire_lengths_cal(4,:),'*');
    ylim([0,60])
    ylabel('Wire 4 Len')
    legend('Reference','Calculated');
    xlabel('Number of times for validation')

end
```

8.2 MATLAB Code for Direct Kinematics

8.2.1 Direct Kinematics for 3-Wire Mechanism

```
% verify 3-wire case 1 (bottom) direct kinematics
% Zhangshi 20160115
clc;
clear all;
syms x real % variable for redundant method
theta2_cal_set = []; % set storing calculated theta2
theta1_cal_set = []; % set storing calculated theta1
alpha1_set = [];     % set storing alpha1
ee_pos_cal_set = []; % set storing calculated ee pos
ee_pos_set = [];     % set stroing referene ee pos
initialization_3wires_case1_direct; %initialization
n1 = length(theta1);
n2 = length(theta2);
n3 = length(alpha1);
% start calculation
for i = 1:n1
    for j = 1:n2
        for m = 1:n3
            % Use configuration direct kinematics to calculate R03
            R01 = rotr([0;0;1],alpha1(m));
            R12 = rotr([1;0;0],theta1(i));
            R23 = rotr([0;1;0],theta2(j));
            R03 = R01*R12*R23;

            % vectors of the wires
            l1_in0 = R01*t1_in1 + R03*t2_in3 + R03*a1_in3 - R01*b1_in1;
            l2_in0 = R01*t1_in1 + R03*t2_in3 + R03*a2_in3 - R01*b2_in1;
```

```
    l3_in0 = R01*t1_in1 + R03*t2_in3 + R03*a3_in3 - R01*b3_in1;
    % get wire lengths as input
    l1 = norm(l1_in0);
    l2 = norm(l2_in0);
    l3 = norm(l3_in0);

    % redundant method to solve equations, x= sin(theta2)
    f0 = -(2*h^2*(1 - x^2)^0.5 - 2*h*r2*x)^2;
    f2 = (2*h^2*(1 - x^2)^0.5 - 2*h*r2*x)^2 - ...
(l1^2 - r2^2 - r1^2 - 2*h^2 + 2*r1*r2*(1 - x^2)^0.5 + 2*r1*h*x)^2;

    g0 = (12*h^2*r1^2)*(1-x^2) - (3^0.5*r1*r2*x + 2*3^0.5*h*r2)^2;
    g1 = 2*(3^0.5*r1*r2*x + 2*3^0.5*h*r2)*(l2^2 - l3^2);
    g2 = -(l2^2 - l3^2)^2;
    D = [0 f0 0 f2;
         0 g0 g1 g2;
         f0 0 f2 0;
         g0 g1 g2 0];
    eq = det(D);    % use det(D) == 0 to solve for x

    sol2 = solve(vpa(eq),x);
    sol = double(sol2);

    sol(sol > 1) = [];
    sol(sol < -1) = [];

    k1 = length(sol);
    theta2_poss = asin(double(sol)); % possible solutions for theta2
    for k = 1:k1
        if abs(theta2_poss(k) - theta2(j)) <= 0.001
            theta2_cal_set = [theta2_cal_set, theta2_poss(k)];
            sin_theta1 = (l2^2 - l3^2)/(2*3^0.5*h*r2 +...
2*3^0.5*h*r1*cos(theta2_poss(k)) + 3^0.5*r1*r2*sin(theta2_poss(k)));
```

```
                    theta1_poss = asin(sin_theta1);
                    theta1_cal_set = [theta1_cal_set, theta1_poss];
                    alpha1_set = [alpha1_set, alpha1(m)];
                    % Reference end effector positions
                     T03 = [
                            cos(alpha1(m))*cos(theta2(j)) -...
                            sin(alpha1(m))*sin(theta1(i))*sin(theta2(j)),...
                            -sin(alpha1(m))*cos(theta1(i)),...
                            cos(alpha1(m))*sin(theta2(j)) +...
                            sin(alpha1(m))*cos(theta2(j))*sin(theta1(i)),...
                            h*cos(alpha1(m))*sin(theta2(j)) +...
                            h*sin(alpha1(m))*cos(theta2(j))*sin(theta1(i));
                            sin(alpha1(m))*cos(theta2(j)) +...
                            cos(alpha1(m))*sin(theta1(i))*sin(theta2(j)),...
                            cos(alpha1(m))*cos(theta1(i)),...
                            sin(alpha1(m))*sin(theta2(j)) -...
                            cos(alpha1(m))*cos(theta2(j))*sin(theta1(i)),...
                            h*sin(alpha1(m))*sin(theta2(j)) -...
                            h*cos(alpha1(m))*cos(theta2(j))*sin(theta1(i));
                            -cos(theta1(i))*sin(theta2(j)),...
                            sin(theta1(i)),...
                            cos(theta1(i))*cos(theta2(j)),...
                            h*(cos(theta1(i))*cos(theta2(j)) + 1);
                            0,0,0,1];
                    ee_pos = T03(1:3,4);    % get reference ee pos
                    ee_pos_set = [ee_pos_set, ee_pos];
                    break;
                end
            end
        end
    end
end
angles_set = [theta1_cal_set;theta2_cal_set;alpha1_set];
```

```
[size1, size2] = size(angles_set);
for i = 1 : size2
    theta1_cal = angles_set(1,i);
    theta2_cal = angles_set(2,i);
    alpha1 = angles_set(3,i);
    % Using direct kinematics to calculate end effector's positions
    T03_cal = [
                cos(alpha1)*cos(theta2_cal) -...
                sin(alpha1)*sin(theta1_cal)*sin(theta2_cal),...
                -sin(alpha1)*cos(theta1_cal),...
                cos(alpha1)*sin(theta2_cal) +...
                sin(alpha1)*cos(theta2_cal)*sin(theta1_cal),...
                h*cos(alpha1)*sin(theta2_cal) +...
                h*sin(alpha1)*cos(theta2_cal)*sin(theta1_cal);
                sin(alpha1)*cos(theta2_cal) +...
                cos(alpha1)*sin(theta1_cal)*sin(theta2_cal),...
                cos(alpha1)*cos(theta1_cal),...
                sin(alpha1)*sin(theta2_cal) -...
                cos(alpha1)*cos(theta2_cal)*sin(theta1_cal),...
                h*sin(alpha1)*sin(theta2_cal) -...
                h*cos(alpha1)*cos(theta2_cal)*sin(theta1_cal);
                -cos(theta1_cal)*sin(theta2_cal),...
                sin(theta1_cal),...
                cos(theta1_cal)*cos(theta2_cal),...
                h*(cos(theta1_cal)*cos(theta2_cal) + 1);
                0,0,0,1];
 ee_pos_cal = T03_cal(1:3,4);   % get calculated ee pos
 ee_pos_cal_set = [ee_pos_cal_set, ee_pos_cal];
end
% draw errors
drawEEerrors(ee_pos_set, ee_pos_cal_set, t1_in1)
```

```
% Initialization for verification 3 wire case 1 direct kinematics
% Zhangshi 20160115

angle_max = pi/3;
angle_min = -angle_max;
step1 = pi/20;
step2 = pi/20;

theta1 = angle_min:step1:angle_max;
theta2 = angle_min:step2:angle_max;
alpha1 = pi/2;
alpha2 = 0;

% theta1 = pi/3;     %Define other two rotational angles
% theta2 = pi/10;
% alpha1 = 0;
% alpha2 = pi/6;

angle = 120/180*pi;
h = 19;
r1 = 18;
r2 = 18;

t1_in1 = [0;0;h];
t2_in3 = [0;0;h];

a1_in3 = [r2;0;0];
a2_in3 = [r2*cos(angle);r2*sin(angle);0];
a3_in3 = [r2*cos(-angle);r2*sin(-angle);0];

b1_in1 = [r1;0;0];
b2_in1 = [r1*cos(angle);r1*sin(angle);0];
b3_in1 = [r1*cos(-angle);r1*sin(-angle);0];
```

```
% draw figures for ee errors
function drawEEerrors(EE_pos_set, EE_pos_cal_set, t1)

tilt_angle_set = [];
tilt_angle_cal_set = [];
% angle of each actual point expressed in a row
polar_angle = atan2(EE_pos_set(2,:),EE_pos_set(1,:));
% angle of each calculated point
polar_angle_cal = atan2(EE_pos_cal_set(2,:),EE_pos_cal_set(1,:));
[m,n] = size(EE_pos_set);
for i = 1:n
    cos_tilt_angle = [0, 0, 1]*uvec(EE_pos_set(:,i) - t1);
    tilt_angle = acos(cos_tilt_angle);
    tilt_angle_set = [tilt_angle_set, tilt_angle];
end

for i = 1:n
    cos_tilt_angle_cal = [0, 0, 1]*uvec(EE_pos_cal_set(:,i) - t1);
    tilt_angle_cal = acos(cos_tilt_angle_cal);
    tilt_angle_cal_set = [tilt_angle_cal_set, tilt_angle_cal];
end

polar_r = (tilt_angle_set)*180/pi;% r of each point referring to tilt angle
polar_r_cal = (tilt_angle_cal_set)*180/pi;
polar(polar_angle,polar_r,'O'); % use O to present reference values
hold on
polar(polar_angle_cal, polar_r_cal,'*'); % use * to present calcualted values
legend('reference values', 'calculated values')
title('End effector position in polar coordinate system')
end
```

8.2.2 Inverse Kinematics for 4-Wire Mechanism

```
% verify 4-wire case 1 (bottom) direct kinematics
% Zhangshi 20160115
clc;
clear all;
syms x real     % variable for redundant method
theta2_cal_set = [];    % set storing calculated theta2
theta1_cal_set = [];    % set storing calculated theta1
alpha1_set = [];        % set storing alpha1
ee_pos_cal_set = [];    % set storing calculated ee pos
ee_pos_set = [];        % set stroing referene ee pos
initialization_4wires_case1_direct; %initialization
n1 = length(theta1);
n2 = length(theta2);
n3 = length(alpha1);
% start calculation
for i = 1:n1
    for j = 1:n2
        for m = 1:n3
            % Use configuration direct kinematics to calculate R03
            R01 = rotr([0;0;1],alpha1(m));
            R12 = rotr([1;0;0],theta1(i));
            R23 = rotr([0;1;0],theta2(j));
            R03 = R01*R12*R23;

            % vectors of the wires
            l1_in0 = R01*t1_in1 + R03*t2_in3 + R03*a1_in3 - R01*b1_in1;
            l2_in0 = R01*t1_in1 + R03*t2_in3 + R03*a2_in3 - R01*b2_in1;
            l3_in0 = R01*t1_in1 + R03*t2_in3 + R03*a3_in3 - R01*b3_in1;
            l4_in0 = R01*t1_in1 + R03*t2_in3 + R03*a4_in3 - R01*b4_in1;
            % get wire lengths as input
            l1 = norm(l1_in0);
            l2 = norm(l2_in0);
            l3 = norm(l3_in0);
```

```
            l4 = norm(l4_in0);
            % solve for sin(theta2)
            u = (l3^2 - l1^2)/(4*h*r2);
            v = (l2^2 + l4^2 - l1^2 - l3^2)/(4*r1*r2)- r1/r2;
            eq = x^4 + (v^2 - 1)*x^2 + 2*u*v*x + u^2 == 0;
            sol = solve(vpa(eq),x);
            sol = double(sol)
            sol(sol > 1) = [];
            sol(sol < -1) = [];

            k1 = length(sol);
            theta2_poss = asin(double(sol));% possible solutions for theta2
            for k = 1:k1
                if abs(theta2_poss(k) - theta2(j)) <= 0.001
                    theta2_cal_set = [theta2_cal_set, theta2_poss(k)];
                    cos_theta1 = cos(theta2_poss(k))...
                        + (l1^2 + l3^2 - l2^2 - l4^2)/(4*r1*r2);
                    sin_theta1 = -(2*h^2 + r1^2 + r2^2 - l2^2 -...
                        2*cos_theta1*r1*r2 + ...
2*cos_theta1*cos(theta2_poss(k))*h^2)/(2*h*r2 + 2*cos(theta2_poss(k))*h*r1);
                    theta1_poss = atan2(sin_theta1, cos_theta1);
                    theta1_cal_set = [theta1_cal_set, theta1_poss];
                    alpha1_set = [alpha1_set, alpha1(m)];
                    % Reference end effector positions
                    T03 = [
                            cos(alpha1(m))*cos(theta2(j))...
                            - sin(alpha1(m))*sin(theta1(i))*sin(theta2(j)),...
                            -sin(alpha1(m))*cos(theta1(i)),...
                            cos(alpha1(m))*sin(theta2(j))...
                            + sin(alpha1(m))*cos(theta2(j))*sin(theta1(i)),...
                            h*cos(alpha1(m))*sin(theta2(j))...
                         + h*sin(alpha1(m))*cos(theta2(j))*sin(theta1(i));...
                            sin(alpha1(m))*cos(theta2(j))...
```

```
                        + cos(alpha1(m))*sin(theta1(i))*sin(theta2(j)),...
                        cos(alpha1(m))*cos(theta1(i)),...
                        sin(alpha1(m))*sin(theta2(j))...
                        - cos(alpha1(m))*cos(theta2(j))*sin(theta1(i)),...
                        h*sin(alpha1(m))*sin(theta2(j))...
                      - h*cos(alpha1(m))*cos(theta2(j))*sin(theta1(i));...
                        -cos(theta1(i))*sin(theta2(j)),...
                        sin(theta1(i)),...
                        cos(theta1(i))*cos(theta2(j)),...
                        h*(cos(theta1(i))*cos(theta2(j)) + 1);...
                        0,0,0,1];

                    ee_pos = T03(1:3,4);
                    ee_pos_set = [ee_pos_set, ee_pos];
                    break;
                end
            end
        end
    end
end
angles_set = [theta1_cal_set;theta2_cal_set;alpha1_set];
[size1, size2] = size(angles_set);
for i = 1 : size2
    theta1_cal = angles_set(1,i);
    theta2_cal = angles_set(2,i);
    alpha1 = angles_set(3,i);
    T03_cal = [
              cos(alpha1)*cos(theta2_cal)...
              - sin(alpha1)*sin(theta1_cal)*sin(theta2_cal),...
              -sin(alpha1)*cos(theta1_cal),...
              cos(alpha1)*sin(theta2_cal)...
              + sin(alpha1)*cos(theta2_cal)*sin(theta1_cal),...
              h*cos(alpha1)*sin(theta2_cal)...
```

```
                + h*sin(alpha1)*cos(theta2_cal)*sin(theta1_cal);
                sin(alpha1)*cos(theta2_cal)...
                + cos(alpha1)*sin(theta1_cal)*sin(theta2_cal),...
                cos(alpha1)*cos(theta1_cal),...
                sin(alpha1)*sin(theta2_cal)...
                - cos(alpha1)*cos(theta2_cal)*sin(theta1_cal),...
                h*sin(alpha1)*sin(theta2_cal)...
                - h*cos(alpha1)*cos(theta2_cal)*sin(theta1_cal);
                -cos(theta1_cal)*sin(theta2_cal),...
                sin(theta1_cal),...
                cos(theta1_cal)*cos(theta2_cal),...
                h*(cos(theta1_cal)*cos(theta2_cal) + 1);
                0,0,0,1];
     ee_pos_cal = T03_cal(1:3,4);   % get calculated ee pos
     ee_pos_cal_set = [ee_pos_cal_set, ee_pos_cal];
end
% draw errors
drawEEerrors(ee_pos_set, ee_pos_cal_set, t1_in1)


% Initialization for verification 4 wire case 1 direct kinematics
% Zhangshi 20160115
angle_max = pi/3;
angle_min = -angle_max;
step1 = pi/20;
step2 = pi/20;

theta1 = angle_min:step1:angle_max;
theta2 = angle_min:step2:angle_max;

alpha1 = pi/2;
alpha2 = pi/2;
```

```
h = 19;
r1 = 18;
r2 = 18;

t1_in1 = [0;0;h];
t2_in3 = [0;0;h];

a1_in3 = [r2;0;0];
a2_in3 = [0;r2;0];
a3_in3 = [-r2;0;0];
a4_in3 = [0;-r2;0];

b1_in1 = [r1;0;0];
b2_in1 = [0;r1;0];
b3_in1 = [-r1;0;0];
b4_in1 = [0;-r1;0];



% draw figures for ee errors
function drawEEerrors(EE_pos_set, EE_pos_cal_set, t1)

tilt_angle_set = [];
tilt_angle_cal_set = [];
% angle of each actual point expressed in a row
polar_angle = atan2(EE_pos_set(2,:),EE_pos_set(1,:));
% angle of each calculated point
polar_angle_cal = atan2(EE_pos_cal_set(2,:),EE_pos_cal_set(1,:));
[m,n] = size(EE_pos_set);
for i = 1:n
    cos_tilt_angle = [0, 0, 1]*uvec(EE_pos_set(:,i) - t1);
    tilt_angle = acos(cos_tilt_angle);
    tilt_angle_set = [tilt_angle_set, tilt_angle];
end
```

```
for i = 1:n
    cos_tilt_angle_cal = [0, 0, 1]*uvec(EE_pos_cal_set(:,i) - t1);
    tilt_angle_cal = acos(cos_tilt_angle_cal);
    tilt_angle_cal_set = [tilt_angle_cal_set, tilt_angle_cal];
end

polar_r = (tilt_angle_set)*180/pi;% r of each point referring to tilt angle
polar_r_cal = (tilt_angle_cal_set)*180/pi;
polar(polar_angle,polar_r,'O'); % use O to present reference values
hold on
polar(polar_angle_cal, polar_r_cal,'*'); % use * to present calcualted values
legend('reference values', 'calculated values')
title('End effector position in polar coordinate system')
end
```

8.3 MATLAB Code for Jacobian Calculation

8.3.1 Partial Jacobian Using Virtual Work Method for 3-Wire Mechanism

```
% This function is to use virtual work to calculate Jacobian.
% When using static method to get
% Jacobian, the end effector forces is the force applied BY end
% effector TO the environment.
% This code does not consider the 3rd dof input torque applied on either
% the base or the upper plate. In this case,
% it is a 2 DOF univeral joint and the Jacobian(3 by 2) is in frame 2.
% The input are wire positions on top and bottom plate
% Zhangshi Liu, 20150629
function J = CalVW_Jac(input_hook_AR,cross_c,output_hook_AR)

%Remember, the last three columns of input/output_hook is the wire points'
```

```
%positions coordinates in frame 2
wirepos_out_1=output_hook_AR(1:3,7);
wirepos_out_2=output_hook_AR(1:3,8);
wirepos_out_3=output_hook_AR(1:3,9);


wirepos_in_1=input_hook_AR(1:3,7);
wirepos_in_2=input_hook_AR(1:3,8);
wirepos_in_3=input_hook_AR(1:3,9);


% li represents the vector of wire expressed in frame 2.
l1 = wirepos_in_1 - wirepos_out_1;
l2 = wirepos_in_2 - wirepos_out_2;
l3 = wirepos_in_3 - wirepos_out_3;


% ri represents the vector pointing from cross_c's origin to
% output_hook's wire point expressed in frame 2.
r1 = output_hook_AR(1:3,7) - cross_c(1:3,5);
r2 = output_hook_AR(1:3,8) - cross_c(1:3,5);
r3 = output_hook_AR(1:3,9) - cross_c(1:3,5);


x2_in2 = [1;0;0];
y2_in2 = [0;1;0];

A = -eye(2);

B11 = (cross(r1,uvec(l1)))'*uvec(x2_in2);
B12 = (cross(r2,uvec(l2)))'*uvec(x2_in2);
B13 = (cross(r3,uvec(l3)))'*uvec(x2_in2);
```

```
B21 = (cross(r1,uvec(l1)))'*uvec(y2_in2);
B22 = (cross(r2,uvec(l2)))'*uvec(y2_in2);
B23 = (cross(r3,uvec(l3)))'*uvec(y2_in2);


B = [-B11,-B12,-B13;
     -B21,-B22,-B23];

J_trans = A'*B;     % J_trans a is 2 by 3 matrix
J = J_trans';
end
```

8.3.2 Partial Jacobian Using Closed Loop Method for 3-Wire Mechanism

```
% This function is to use closed loop kinematics method to calculate
% partial Jacobian
% it is a 2 DOF univeral joint and the Jacobian is a 3 by 2 matrix in frame 2.
% Zhangshi Liu, 20150629
function J = CalCL_Jac(input_hook_AR,output_hook_AR,T03,R20,h)
% l1, l2 and l3 expressed in frame 2
l1 = -output_hook_AR(1:3,7)+input_hook_AR(1:3,7);
l2 = -output_hook_AR(1:3,8)+input_hook_AR(1:3,8);
l3 = -output_hook_AR(1:3,9)+input_hook_AR(1:3,9);
% unit vector of l1, l2 and l3 in frame 2
l1_uvec = uvec(l1);
l2_uvec = uvec(l2);
l3_uvec = uvec(l3);

% b1, b2 and b3 expressed in frame 2
b1 = input_hook_AR(1:3,7)-input_hook_AR(1:3,3);
b2 = input_hook_AR(1:3,8)-input_hook_AR(1:3,3);
b3 = input_hook_AR(1:3,9)-input_hook_AR(1:3,3);
```

```
% a1, a2 and a3 expressed in frame 2
a1 = output_hook_AR(1:3,7)-output_hook_AR(1:3,6);
a2 = output_hook_AR(1:3,8)-output_hook_AR(1:3,6);
a3 = output_hook_AR(1:3,9)-output_hook_AR(1:3,6);

x2_in2 = [1;0;0];
y2_in2 = [0;1;0];

t = T03*[0,0,0,1]';          %end effector's position in frame 0
t1 = R20*[0;0;h];            %The first link vector in frame 2
t2 = R20*t(1:3) - t1;        %The second link vector in frame 2

% Formation of A and B: Ax=Bq
% A is a 3 by 2 matrix, B is 3 by 3
A_kin=[(dot(cross(b1,l1_uvec),x2_in2)-dot(cross(t1,l1_uvec),x2_in2)), (dot(cross(t2,
       (dot(cross(b2,l2_uvec),x2_in2)-dot(cross(t1,l2_uvec),x2_in2)), (dot(cross(t2,
       (dot(cross(b3,l3_uvec),x2_in2)-dot(cross(t1,l3_uvec),x2_in2)), (dot(cross(t2,

B_kin=eye(3);
J = B_kin'*A_kin;
end
```

8.3.3 Calculate Complete Jacobian

```
% This code is to calculate the complete Jacobian for the hybrid wrist in
% case 1.
% The universal joint can be regarded as a 2-DOF parallel robot, and it
% sits on a base which rotates about z=[0;0;1] in world frame.
% 1. The parallel jacobian, Jp, which is a 3X2 matrix: Jp*x_dot = q_dot,
% can be get from any of the two methods.
% 2. Since Jp is tall, we can left-multiply pinv_Jp to express x_dot (2 by 1 matrix)
```

```
% 3. Use x_dot3/0_in0 = x_dot3/1_in0 + x_dot1/0_in0 to express x_dot3/0_in0.
% 4. Get the compelet Jacobian for the hybrid manipulator.
% 20150702, Nabil and Zhangshi
function J_in0 = CalCompleteJac_rotin(input_hook_AR,cross_c,...
    output_hook_AR,T02)

R02 = T02(1:3,1:3);
R20 = R02';
Jp_in2 = CalVW_Jac(input_hook_AR,cross_c,output_hook_AR);
z0_in2 = R20*[0;0;1];
pinv_Jp_in2 = pinv(Jp_in2);
J_in0 = R02*[[pinv_Jp_in2;0,0,0],z0_in2];
end
```

8.4 Workspace Calculation

8.4.1 MATLAB code for Workspace of Infinite Stiffness Wires

```
% This code is to calculate workspace of infinite stiffness wires given
% certain theta by scanning all the phi and beta
% inputs are: configuration angles, wire positions on top/bottom plate,
% hook heights.
% outputs are: workspace points and corresponding configuration angles as
% well as tilt angles.
% Zhangshi Liu, 2015/05/27, based on Saleem's code.

function [phi_point,beta_point,locations,tilt_angle] = CalWS_VWJac(theta,...
  phi, beta, theta_out, Diameter,wire_top_diameter,...
  wire_bottom_diameter,hook_heights)
locations = [];     % End effector's positions are stored here
n_phi = length(phi);    % number of phi
n_beta = length(beta);  % number of beta
```

```
tilt_angle = [];
phi_point = [];
beta_point = [];
% outputs are hooks' and cross' coordinates (expressed in frame 2)
% when theta, phi and beta are all 0.
[input_hook,cross_c,output_hook]=setrobot(Diameter,hook_heights,...
    wire_top_diameter,wire_bottom_diameter);

    for i=1:1:n_phi
        phi_test=phi(i);

        for j = 1:1:n_beta
            beta_test = beta(j);

           % get transformation matrices. Configuration Direct kinematics
           % The outputs are T01, T02 and T03.
            [input_hook_cs, cross_cs, output_hook_cs, T04]=...
        Dir_Seri_Kin(theta, phi_test, beta_test, theta_out, hook_heights);
            % Define transform and rotation matrices
            T01=input_hook_cs;
            T02=cross_cs;
            T03=output_hook_cs;
            T23=T02\T03;
            T21=T02\T01;
            R23 = T23(1:3,1:3);
            R21 = T21(1:3,1:3);
            R02 = T02(1:3,1:3);

            % input_hook and output_hook's coordinates after
            % rotation expressed in frame 2
            input_hook_AR=R21*input_hook(1:3,:);
            output_hook_AR=R23*output_hook(1:3,:);
            % End effector's vector expressed in world frame
```

```
            EE_pos_vec = R02*(R23*output_hook(1:3,6)-cross_c(1:3,5));
            % cos(angle), angle is between end effector's vector
            % and [0;0;1], namely the tilt angle.
            angle_cos = [0 0 1]*uvec(EE_pos_vec);

            % case1 Jacobian
            Jacobian = CalCompleteJac_rotin(input_hook_AR,cross_c,...
                output_hook_AR,T02);
            % case2 Jacobian
%             Jacobian = CalCompleteJac_rotout(input_hook_AR,cross_c,...
            % output_hook_AR,R02,R23);
            if (rank(Jacobian) == 3)
                JT = Jacobian';
                % Modified code that Jacobian only considers FOUR wires
                JT_wave = JT(1:3,1:3);
                pinvJT_wave = pinv(JT_wave);
                nj = null(pinvJT_wave);
            % Here the 'if' statement is used to make sure that the null
            % space's values are all positive or all negative.
            % And if so, the values of theta_test and phi_test can be
            % assigned to theta_point and phi_point, which means that point
            % satisfies the tension requirement.
                if( (sum(nj(1:3) > 0) == 3) || (sum(nj(1:3) < 0) == 3))
                     tilt = acos(angle_cos);
                     if(tilt < (90*pi/180))
                        phi_point = [phi_point,phi_test];
                        beta_point = [beta_point,beta_test];
                        location = T03*[0;0;0;1];
                        locations = [locations,location(1:3)];
                        tilt_angle = [tilt_angle;tilt];
                     end
                end
            end
```

```
        end
    end
end
```

8.4.2 MATLAB code for Workspace of Finite Stiffness Wires

```
% This code is to calculate workspace of finite stiffness wires
% inputs are: configuration angles, external wrench, joint stiffness, wire
% positions on top/bottom plate,hook heights
% outputs are: points in workspace, wire tension, corresponding
% configuration angles, and tilt angles
% Zhangshi Liu, 20150721, based on Saleem's code

function [locations,lamda_set,tau_set,phi_set,beta_set,tilt_angle] = ...
    CalWSGivenExtWrench(wrench,kd,theta, phi, beta, theta_out, Diameter,...
    D_top,D_bottom,wire_top_diameter,wire_bottom_diameter,hook_heights)
kapa = diag(kd); %Transfer stiffness vector into matrix
locations = []; % positions of end effector's center
n_phi = length(phi);    %num of phi
n_beta = length(beta);  %num of beta
tilt_angle = [];    % EE center's tilt angle
phi_set = [];   % set containing effective phi values
beta_set = [];  % set containing effective beta values
 % set containing lamda that is least needed to be multiplied
 % with null vectors to make wire tensions positive
lamda_set = [];
tau_set = [];   % set containing wire tensions corresponding with lamda
h = hook_heights(1);
% outputs are hooks' and cross' coordinates
% (expressed in frame 2) when theta, phi and beta are all 0.
[input_hook,cross_c,output_hook]=setrobot(Diameter,...
                hook_heights,wire_top_diameter,wire_bottom_diameter);
```

```
% The following two for loops scan each point corresponding to phi and beta
    for i=1:1:n_phi
        phi_test=phi(i);
        for j = 1:1:n_beta
            beta_test = beta(j);

           % get transformation matrices. The outputs are T01, T02 and T03.
            % T04 is the transformation matrix that is used when the
            % 4th input is applied on top plate.
            [input_hook_cs, cross_cs, output_hook_cs, T04]=...
        Dir_Seri_Kin(theta, phi_test, beta_test, theta_out, hook_heights);

            % The input parameters are expressed in frame 2,
            % output is Jacobian in world frame.
            % Here the coordinates of input and output hooks which are
            % acquired when theta,phi and beta are 0
            % need to be transformed using T.
    % Jacobian = Cal_Jacobian_inEEFrame(T21*input_hook,cross_c,
    % T23*output_hook,theta_test, phi_test,beta); %Jacobian is 4 by 3

            T01=input_hook_cs;
            T02=cross_cs;
            T03=output_hook_cs;
            T23=T02\T03;
            T21=T02\T01;
            R23 = T23(1:3,1:3);
            R21 = T21(1:3,1:3);
            R02 = T02(1:3,1:3);
    % input_hook and output_hook after rotation expressed in frame 2
            input_hook_AR=R21*input_hook(1:3,:);
            output_hook_AR=R23*output_hook(1:3,:);
       J = CalCompleteJac_rotin(input_hook_AR,cross_c,output_hook_AR,T02);
%             yita = 0.5 % step size for delta_x in the second method
```

```
            if (rank(J) == 3)
               JT = J';
               JT_wave = JT(1:3,1:3);
               pinvJT_wave = pinv(JT_wave);
               nj=null(pinvJT_wave);
            % Here the two if statement is used to make sure that the null
            % space's values are all positive or all negative.
            % And if so, the values of theta_test and phi_test can be
            % assigned to theta_point and phi_point, which means that point
            % satisfies the tension requirement.
               if( (sum(nj(1:3) > 0) == 3) || (sum(nj(1:3) < 0) == 3))
                   % one point in workspace with infinite stiffness
                   location_temp1 = T03*[0;0;0;1];
                   angle_cos = [0 0 1]*uvec(location_temp1(1:3)-[0;0;h]);
                   tilt = acos(angle_cos);
                   if(tilt < (90*pi/180))
                      location_temp = location_temp1;
                      C = J*(kapa\J');    %C is the compliance matrix
                   % V columns are eig vectors and D's diag are eig values
                      [V,D] = eig(C);
                      D_vec = diag(D);% Transfer matrix D into vector D_vec
 % select the max eig value. Here use abs to make all eig values positive.
                      [val,num] = max(abs(D_vec));
 % Define the external wrench in the direction of
 % eig vector corresponding to max eig value
                      wrench_ext = wrench*uvec(V(:,num));
% Given certain external moment, compute delta_x--representing
% the alteration of ANGLE.
                      delta_x = C*(wrench_ext);
                      %% First method to calculate workspace
                       location = CalStiffnessWorkspace1(location_temp,...
                           delta_x,theta,hook_heights);
```

```
                        %% Second method to calculate workspace
                        % Here use "CalStiffnessWorkspace2" function to get
                        % new workspace with finite stiffness, outputs are
                       % EE center's position and corresponding phi and beta
%[location,phi_val,beta_val] = CalStiffnessWorkspace2(location_temp,theta,...
% theta_out,delta_x,yita,hook_heights,input_hook,cross_c,...
% output_hook,kapa,wrench);

                        %% Record results either 1st or 2nd method
                        center_c = T02*cross_c(:,5);
                        EE = location - center_c(1:3);
                        EE_uvec = uvec(EE(1:3));
                        tilt_angle_temp = acos(EE_uvec'*[0;0;1]);
                        [lamda, tau] = CalLamdaTau(J,nj,wrench_ext);
                        lamda_set = [lamda_set,lamda];
                        tau_set = [tau_set,tau];
                        locations = real([locations,location]);
                    tilt_angle = real([tilt_angle;acos(EE_uvec'*[0;0;1])]);
                    end
                end
            end
        end
    end
end


% This function is to set the cross, input_hook and output_hook's points' coordinates
% Zhangshi Liu, 2015/05/26, based on Saleem's code
function [input_hook,cross_c,output_hook]=setrobot(Diameter,hook_heights,...
    wire_top_diameter,wire_bottom_diameter)

%rename the parameters
d=Diameter;
```

```
D_top=wire_top_diameter;
D_bottom=wire_bottom_diameter;
h1=hook_heights(1);

%% Initialize input_hook, cross and output_hook
% columns correspond to xyz coordinates of each hook point and three top
% wire connection points. origin at 3 with positive x towards 4(input hook)
% positive z
% towards 1 and 5 (input hook). input hook's wires are at variable diameter

%The last three colunms(7,8,9,10) refer to xyz coordinates of the wire points.
%The 6th point is the origin of input_hook, used to calculate rf

% Remember, all the coordinates should be expressed in frame 2, the cross_c
% frame.

input_hook=[-d/2, -d/2, 0, d/2, d/2, 0, D_bottom/2,...
    -sin(pi/6)*D_bottom/2, -sin(pi/6)*D_bottom/2;...
              0,    0,  0,  0,   0,  0,  0,   ...
              cos(pi/6)*D_bottom/2, -cos(pi/6)*D_bottom/2;...
              0,  -h1, -h1, -h1, 0, 0, -h1,     -h1,           -h1;...
              1,    1,  1,  1,   1,  1,  1,     1,              1];
h2=hook_heights(2);

%The last three colunms(7,8,9) refer to xyz coordinates of the wire points.
%The 6th point is the origin of output_hook


output_hook=[0,   0,  0,   0,    0,  0, D_top/2, -sin(pi/6)*D_top/2,...
            -sin(pi/6)*D_top/2;...
            d/2, d/2, 0, -d/2, -d/2, 0,  0,   cos(pi/6)*D_top/2,...
            -cos(pi/6)*D_top/2;...
            h1,   0,  0,   0,    h1, h1, h1,       h1,           h1;...
```

```
             1, 1, 1, 1, 1, 1, 1, 1, 1];
cross_c=[d/2,   0,     -d/2,  0,    0;...
         0,     d/2,   0,     -d/2, 0;...
         0,     0,     0,     0,    0;...
         1,     1,     1,     1,    1];


% This code, regarded as serial robot in this case, is to calculate direct
% kinematics. Namely, calculate the cross, input and
% output hooks' coordinates' homogeneous transposes with regard to world
% frame given three rotation angles: theta, phi and beta, as well as
% hook_heights and shaft_lengths.
% Zhangshi Liu, 2015/05/26, ARMA Lab, Vanderbilt University
function [input_hook_cs, cross_cs, output_hook_cs, T04]=Dir_Seri_Kin(theta,...
    phi, beta, theta_out, hook_heights)

%% Calculate Rotation matrices

%theta is the rotation angle of input hook,
%phi is the rotation angle about x02 and beta y02
R01=rotr([0;0;1],theta);
R12=rotr([1;0;0],phi);
R02=R01*R12;
R23=rotr([0;1;0],beta);
R03=R02*R23;
R34=rotr([0;0;1],theta_out);
R04 = R03*R34;

%% Calculate homogenous transposes
h1 = hook_heights(1);
h2 = hook_heights(2);

T01=[R01,[0;0;h1];0 0 0 1];
```

```
T02=[[R02;[0 0 0]], T01*[0 0 0 1]'];
T03=[[R03;[0 0 0]], T02*[h2*sin(beta) 0 h2*cos(beta) 1]'];
T04=[[R04;[0 0 0]], T03*[0 0 0 1]'];

%% Here the input_hook_cs' origin is coincident with cross_cs' origin.
% the output_hook_cs' origin is on the surface of output hook.
input_hook_cs=T01;
cross_cs=T02;
output_hook_cs=T03;
end


function location = CalStiffnessWorkspace1(location_temp,delta_x,theta,hook_heights)

    h = hook_heights(1);
    delta_angle = -norm(delta_x);
    delta_axis = delta_x/delta_angle;
    u = delta_axis(1);
    v = delta_axis(2);
    w = delta_axis(3);
    delta_R = [u^2+(v^2+w^2)*cos(delta_angle), ...
        u*v*(1-cos(delta_angle))-w*sin(delta_angle), ...
        u*w*(1-cos(delta_angle))+v*sin(delta_angle);
              u*v*(1-cos(delta_angle))+w*sin(delta_angle), ...
              v^2+(u^2+w^2)*cos(delta_angle), ...
              v*w*(1-cos(delta_angle))-u*sin(delta_angle);
              u*w*(1-cos(delta_angle))-v*sin(delta_angle), ...
              v*w*(1-cos(delta_angle))+u*sin(delta_angle), ...
              w^2+(u^2+v^2)*cos(delta_angle)];
    delta_angle_2 = norm(delta_x);
    delta_axis_2 = delta_x/delta_angle;
    u_2 = delta_axis_2(1);
    v_2 = delta_axis_2(2);
```

```
    w_2 = delta_axis_2(3);
    delta_R_2 = [u_2^2+(v_2^2+w_2^2)*cos(delta_angle_2), ...
        u_2*v_2*(1-cos(delta_angle_2))-w_2*sin(delta_angle_2), ...
        u_2*w_2*(1-cos(delta_angle_2))+v_2*sin(delta_angle_2);
                u_2*v_2*(1-cos(delta_angle_2))+w_2*sin(delta_angle_2), ...
                v_2^2+(u_2^2+w_2^2)*cos(delta_angle_2), ...
                v_2*w_2*(1-cos(delta_angle_2))-u_2*sin(delta_angle_2);
                u_2*w_2*(1-cos(delta_angle_2))-v_2*sin(delta_angle_2), ...
                v_2*w_2*(1-cos(delta_angle_2))+u_2*sin(delta_angle_2), ...
                w_2^2+(u_2^2+v_2^2)*cos(delta_angle_2)];

    angle_cos = [0 0 1]*uvec(location_temp(1:3)-[0;0;h]);
    location_1 = delta_R*(location_temp(1:3)-[0;0;h]);
    angle_cos_1 = [0 0 1]*uvec(location_1);
    location_2 = delta_R_2*(location_temp(1:3)-[0;0;h]);
    angle_cos_2 = [0 0 1]*uvec(location_2);

    [value,index] = max([angle_cos,angle_cos_1,angle_cos_2]);
    if(index == 1)
         location = location_temp(1:3);
    elseif(index == 2)
         location = location_1 + [0;0;h];
    elseif(index == 3)
         location = location_2 + [0;0;h];
    end
end


% This function is to calculate workspace with finite stiffness step by step
function [location,phi,beta] = CalStiffnessWorkspace2(location_temp,theta,theta_out,delta_x

h = hook_heights(1);
EE_pos = location_temp(1:3);
```

```
% This variable is used to accumulate yita*delta of each step and the while loop wil
sum = 0;
    while(sum <= norm(delta_x))
        %% We have to calculate new Jacobian and phi/beta for each loop.
        % Get new Jacobian based on new phi and beta
        %use inverse kinematics to find current phi and beta corresponding to curren
        [phi,beta] = InvKin2DOF(EE_pos,hook_heights);
        [input_hook_cs, cross_cs, output_hook_cs, T04] = ...
            Dir_Seri_Kin(theta, phi, beta, theta_out, hook_heights);
        T01=input_hook_cs;
        T02=cross_cs;
        T03=output_hook_cs;
        T23=T02\T03;
        T21=T02\T01;
        R23 = T23(1:3,1:3);
        R21 = T21(1:3,1:3);
        R02 = T02(1:3,1:3);
        % input_hook and output_hook after rotation expressed in frame 2
        input_hook_AR=R21*input_hook(1:3,:);
        output_hook_AR=R23*output_hook(1:3,:);
        J = CalCompleteJac_rotin(input_hook_AR,cross_c,output_hook_AR,T02);
        %% Use new Jacobin to update sum
        C = J*(kapa\J');    % C is the compliance matrix of current config
        [V,D] = eig(C); % Calculate eigenvalues
        D_vec = diag(D);    % Transform the eigenvalue matrix into a vector
        [val,num] = max(abs(D_vec));    % Find the greatest eigenvalue
        wrench_ext = wrench*uvec(V(:,num)); % Find the eigenvector
        delta_x_new = C*(wrench_ext);   % Get new delta_x
        delta_step = delta_x_new*yita;
        delta_step_angle = -norm(delta_step)
        delta_step_axis = delta_step/delta_step_angle;
        u = delta_step_axis(1);
        v = delta_step_axis(2);
```

```
w = delta_step_axis(3);
% This is the rotation matrix about an axis in space, the angle
% refers to the norm of delta_step, and the direction [u,v,w] is
% the unit vector of delta_step
delta_step_R = [u^2+(v^2+w^2)*cos(delta_step_angle),...
    u*v*(1-cos(delta_step_angle))-w*sin(delta_step_angle),...
    u*w*(1-cos(delta_step_angle))+v*sin(delta_step_angle);
           u*v*(1-cos(delta_step_angle))+w*sin(delta_step_angle),...
                v^2+(u^2+w^2)*cos(delta_step_angle),...
            v*w*(1-cos(delta_step_angle))-u*sin(delta_step_angle);
            u*w*(1-cos(delta_step_angle))-v*sin(delta_step_angle),...
            v*w*(1-cos(delta_step_angle))+u*sin(delta_step_angle),...
            w^2+(u^2+v^2)*cos(delta_step_angle)];

delta_step_angle_2 = norm(delta_step);
delta_step_axis_2 = delta_step/delta_step_angle;
u_2 = delta_step_axis_2(1);
v_2 = delta_step_axis_2(2);
w_2 = delta_step_axis_2(3);

% This is the rotation matrix about an axis in space, the angle
% refers to the norm of delta_step, and the direction [u,v,w] is
% the unit vector of delta_step
delta_step_R_2 = [u_2^2+(v_2^2+w_2^2)*cos(delta_step_angle_2),...
  u_2*v_2*(1-cos(delta_step_angle_2))-w_2*sin(delta_step_angle_2),...
  u_2*w_2*(1-cos(delta_step_angle_2))+v_2*sin(delta_step_angle_2);
  u_2*v_2*(1-cos(delta_step_angle_2))+w_2*sin(delta_step_angle_2),...
  v_2^2+(u_2^2+w_2^2)*cos(delta_step_angle_2),...
  v_2*w_2*(1-cos(delta_step_angle_2))-u_2*sin(delta_step_angle_2);
  u_2*w_2*(1-cos(delta_step_angle_2))-v_2*sin(delta_step_angle_2),...
  v_2*w_2*(1-cos(delta_step_angle_2))+u_2*sin(delta_step_angle_2),...
  w_2^2+(u_2^2+v_2^2)*cos(delta_step_angle_2)];
```

```
        angle_step_cos = [0 0 1]*uvec(EE_pos-[0;0;h]);
        location_step_1 = delta_step_R*(EE_pos-[0;0;h]);
        angle_cos_1 = [0 0 1]*uvec(location_step_1);
        location_step_2 = delta_step_R_2*(EE_pos-[0;0;h]);
        angle_cos_2 = [0 0 1]*uvec(location_step_2);

        [value,index] = max([angle_cos_1,angle_cos_2]);
        if(index == 1)
             EE_pos = location_step_1 + [0;0;h];
        elseif(index == 2)
             EE_pos = location_step_2 + [0;0;h];
        end

        sum = sum + norm(delta_x_new*yita);
    end
location = EE_pos;
end
```