DESIGN AND CONTROL OF A TENSEGRITY PRISM MANIPULATOR

A Thesis

by

HARESH KARNAN

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Robert Skelton
Co-Chair of Committee,	Manoranjan Majji
Committee Member,	Raktim Bhattacharya
Head of Department,	Rodney Bowersox

August 2018

Major Subject: Aerospace Engineering

ABSTRACT

Tensegrity, a word coined by Buckminster Fuller in the 1960s, is a portmanteau of words *tension* and *integrity*. A tensegrity structure is a prestressable network of bars and strings that can be acted by external forces at its nodes. Tensegrity structures were initially of interest to Kenneth Snelson who built architectural structures by interconnecting bars and strings into a stable configuration. In the engineering world, such structures became of interest to control theory specialists who treated tensegrity configurations as a robotic system, capable of dynamic motion with the freedom to control the tension in the strings. With the recent escalation in computing power and the birth of optimization algorithms for mass minimization of tensegrity, it was now possible to design an optimal, static tensegrity structure with minimal mass for a plethora of applications.

Manipulators are devices used to point, position and place objects in space without direct human contact. Over the years, several 2D and 3D manipulators have taken over the robotics industry and have been employed in the manufacturing and space research sector for various applications. This thesis focuses on a three dimensional tensegrity spatial manipulator, based on the 3 strut tensegrity prism topology. *InTense* as it will be referred hereafter in this text is a tensegrity based spatial manipulator designed to be used for pointing control applications in space.

In this research, a tensegrity based manipulator/joint design is explored. An experimental setup of the final design is built and the tensegrity shape controller is derived and implemented to point the manipulator towards a given vector. Results and Inferences from the theoretical and experimental study are documented.

DEDICATION

To YOU - for taking your precious time to read my 2 years work in Tensegrity Robotics.

ACKNOWLEDGMENTS

I am highly indebted to my advisor Dr. Robert E. Skelton for consistently exciting my learning modes and making the learning process worth its effort. His course on Design and Control of Flexible Structures has been greatly useful in influencing my thinking and carrying out this research. I am grateful for the countless hours of valuable one to one discussions, financial support and career advices from him.

I would also like to thank my co advisor Dr. Manoranjan Majji for amplifying my strengths in experimental robotics. His teachings on Kalman Filter theory and flexible structures is greatly appreciated.

Thanks to Dr.Raktim Bhattacharya for having kindly agreed to serve on my committee.

Thanks to Dr.Suman Chakravorty for his course on Estimation theory.

Thanks to Tyler Bryant, Muhao Chen and Dr. Edwin Peraza Hernandes for their constant support and encouragement. I would also like to thank Raman for motivating and helping me climb relatively steeper learning curves in tensegrity theory.

I would also like to thank my parents Mr.Karnan and Mrs.Sumathy, and my grandparents Sakunthala and Late. Mr.Gopalakrishnan for encouraging me to pursue graduate studies in USA.

Thanks to Chandana Amanchi, Madhuri Gontala, Nithin Reddy and Mythreyi Vodnala, for their constant support and encouragement.

I would also like to thank Sumithra akka, Prabhu Sivalingam and my two amazing cousins Sahana and Nivi for their care and affection towards me. Thanks, akka.

Special thanks to RMI NITT, the club without which the experimental aspect of my skillsets would have been nonexistent.

I also dedicate this thesis to my love, Gayatri Sivaraman, who stood by me at difficult times. Special thanks to Sanjana Rajendran for her motivational comments.

CONTRIBUTORS AND FUNDING SOURCES

Funding Sources

This graduate study was supported by a Graduate Research Assistantship from the Aerospace Engineering Department of Texas A&M University, College Station, TX.

TABLE OF CONTENTS

Page

ABSTRACT ii

DEDICATION iii

ACKNOWLEDGMENTS iv

CONTRIBUTORS AND FUNDING SOURCES v

TABLE OF CONTENTS vi

LIST OF FIGURES viii

LIST OF TABLES xi

1. INTRODUCTION 1

1.1 Tensegrity Structures 1
1.1.1 Classes of Tensegrity Structures 2
1.1.2 Why Tensegrity? 3
1.1.3 Self-Similar Tensegrity 4
1.2 Parallel and Serial Manipulators 4
1.3 Manipulators in space 6
1.3.1 The Canfield Joint 6
1.3.2 The Stewart Platform 7
1.3.3 Drawbacks with standard manipulators 8
1.4 Tensegrity solutions for designing manipulators 10
1.4.1 The Tensegrity Prism Manipulator 10
1.4.2 The 3D T-D bar as a Tensegrity Manipulator 14
1.5 Conclusion 16

2. DYNAMICS OF CONSTRAINED TENSEGRITY STRUCTURES 17

2.1 A rigid bar in 3D space 17
2.2 Dynamics of a Rigid Rod 18
2.3 Expressing the tensegrity dynamics in Matrix form 19
2.4 Reduced form of class K Dynamics 22
2.5 Why Non-minimal coordinates 24

3. CONTROL OF TENSEGRITY STRUCTURES 25

3.1 Introduction 25
3.2 Optimization to find the target position with minimum change in string length 25
3.2.1 Explanation of the constraint equations 26
3.2.2 Feasible workspace of the tensegrity prism 28
3.3 Shape control of class 1 tensegrity systems 29
3.3.1 Derivation of the control law 32
3.4 Shape control of class K tensegrity systems 35
3.4.1 Derivation of the control law 35
3.5 Pointing control of class K tensegrity structure 38
3.6 Examples 40
3.6.1 Shape control of tensegrity prism manipulator 40
3.6.2 Pointing control of tensegrity prism manipulator 45
3.7 Analysis of the theoretical results 46
3.7.1 Shape controller example 46
3.7.2 Pointing controller example 50

4. TESTING THE CONTROL ON A LABORATORY PROTOTYPE 52

4.1 Construction of the Robot - From Idea to Prototype 52
4.1.1 Idea 52
4.1.2 Design 55
4.1.3 Prototype 57
4.1.4 Electronics 59
4.2 Using ROS for Tensegrity Robotics Programming 62
4.3 Extended Kalman Filter 64
4.3.1 Estimating the states 67
4.4 Results of shape control on the experimental setup 70

5. CONCLUSIONS AND FUTURE WORK 73

5.1 Conclusion 73
5.2 Directions for future research 73

REFERENCES 74

LIST OF FIGURES

FIGURE Page

1.1 A simple Tensegrity structure. 1

1.2 A Class 2 Tensegrity Dbar. 2

1.3 T-Bar self similar iteration. 4

1.4 Serial and Parallel manipulators. 5

1.5 DaVinci the robotic arm, used in robot assisted surgeries. 6

1.6 A parallel robot used as a flight simulator at SuperJet International. 7

1.7 A Serial Manipulator Robot (in red) fitting Tesla S auto parts at Tesla Factory. 8

1.8 An ABB Delta robot used in industry for sorting goods. 9

1.9 The Canfield Mechanism. 10

1.10 The Stewart-Platform Mechanism. 11

1.11 Empirical structure of Canfield mechanism labelling the 6 bars and 9 joints. 11

1.12 AMiBA 2 space radio telescope mounted on a Stewart platform. 12

1.13 Hexapod robot used in robot assisted brain surgery. 12

1.14 3 bar 9 string tensegrity prism mechanism. 13

1.15 3 bar 12 string tensegrity prism mechanism. 13

1.16 Empirical structure of the 3 bar 6 string Tensegrity prism manipulator. 15

1.17 Constant width T-D bar. 15

1.18 Empirical structure of a 3D T-Dbar manipulator mechanism. 16

2.1 A rigid bar in 3D space acted by external forces. 17

3.1 2 skew lines (rigid bars) in 3D space. 28

3.2 Bar length constraint for the prism, expressed as individual spherical surface boundaries. 29

3.3 Top view of feasible equilibrium configurations of the centroid. 30

3.4 Side view of feasible equilibrium configurations of the centroid. 31

3.5 Isometric view of feasible equilibrium configurations of the centroid. 32

3.6 3 bar 6 string prism manipulator initial configuration. 41

3.7 Side view of the initial configuration. 41

3.8 Top view of the initial configuration. 42

3.9 3 bar 6 string prism manipulator final configuration. 42

3.10 Error in the nodes being driven to zero. 44

3.11 Error in the barlengths during the control process. 45

3.12 Force densities in all 6 strings during the control process. 46

3.13 Initial configuration of the prism. 47

3.14 Final configuration of the prism. 48

3.15 Error in the barlengths during the control process. 50

3.16 Dot product of string vectors with the given target vector. 51

4.1 The 3 strut Tensegrity prism laboratory prototype. 52

4.2 The 3 strut Tensegrity prism idea handdrawn cartoon. 53

4.3 The Bourns-Moser absolute position encoder - EMS22A30-C28-LS6. 54

4.4 A SolidWorks rendering of the prism manipulator - Isometric view. 55

4.5 A SolidWorks rendering of the prism manipulator - TopView. 56

4.6 A SolidWorks rendering of the prism manipulator - Bottom view. 56

4.7 A SolidWorks drawing of the pulley used to actuate the strings. 57

4.8 A parallel-axis spur gear arrangement for measuring the azimuth. 58

4.9 LED status indicator strip. 60

4.10 Components attached at the base of the Tensegrity Robot. 61

4.11 Spectra strings routed through the hollow bars. 61

4.12 Endcap, to reduce stiction between strings and brass bar's ends....................... 62

4.13 Block diagram showing the different modules and connections in the robot.......... 65

4.14 The final experimental setup... 66

4.15 Azimuth and Elevation measurement. 67

4.16 A video collage of the control in action. Elevation : 60, Azimuth : -75................ 71

4.17 Trial 1 : Error in the top 3 node positions (in) vs Time (sec) 71

4.18 A video collage of the control in action. Elevation : 75, Azimuth : 90 72

4.19 Trial 2 : Error in the top 3 node positions (in) vs Time (sec) 72

LIST OF TABLES

TABLE Page

4.1 Table listing the specifications in rosparam server 64

1. INTRODUCTION

1.1 Tensegrity Structures

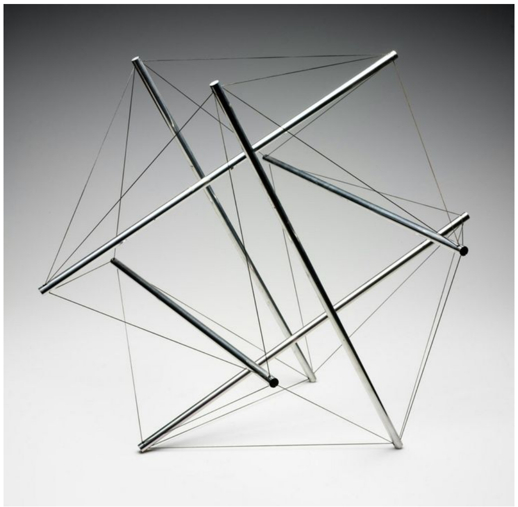

Figure 1.1: A simple Tensegrity structure.

A Tensegrity structure is a network of prestressable, axially loaded compressive and tensile members. The term *Tensegrity* was coined by Buckminster Fuller and his student Kenneth Snelson as a conjucture of the two words *tension* and *integrity*. Tensegrity structures are composed of bars (compressive members) and strings (tensile members). Though bars can also support tensile loads, we restrict only strings to take tensile loads by replacing bars in tension with a string, to maintain consistency. A tensegrity configuration is said to be stable if there exists a set of tensile and compressive forces at the strings and bars respectively to hold the structure at equilibrium. This statement holds true even when the tensegrity structure is subject to external forces such as external load, gravity, viscous fluid forces, etc, or even if the structure is only composed of bars.

We classify between forces existing from within the structure and outside. The internal forces are of two types, tensile forces in the strings and compressive forces in the bars. External forces are all the other forces acting on the elements of a tensegrity structure(gravity, viscous fluid forces, joint forces, friction). Tensegrity structures have been of great importance in architecture and engineering in the history. Kenneth Snelson engineered several tensegrity structures in the past between 1960s and early 2000s. Some of his architectural work has been listed at his website. Parallely in the engineering world, control theoreticians realized the potential of tensegrity to create dynamic structures in robotics and other engineering applications where minimizing mass and control energy of a structure was the design priority. The tensions and the rest length of the strings of a tensegrity can be changed to metamorphose the structure from one shape to another, thus creating a dynamically moving structure.

1.1.1 Classes of Tensegrity Structures

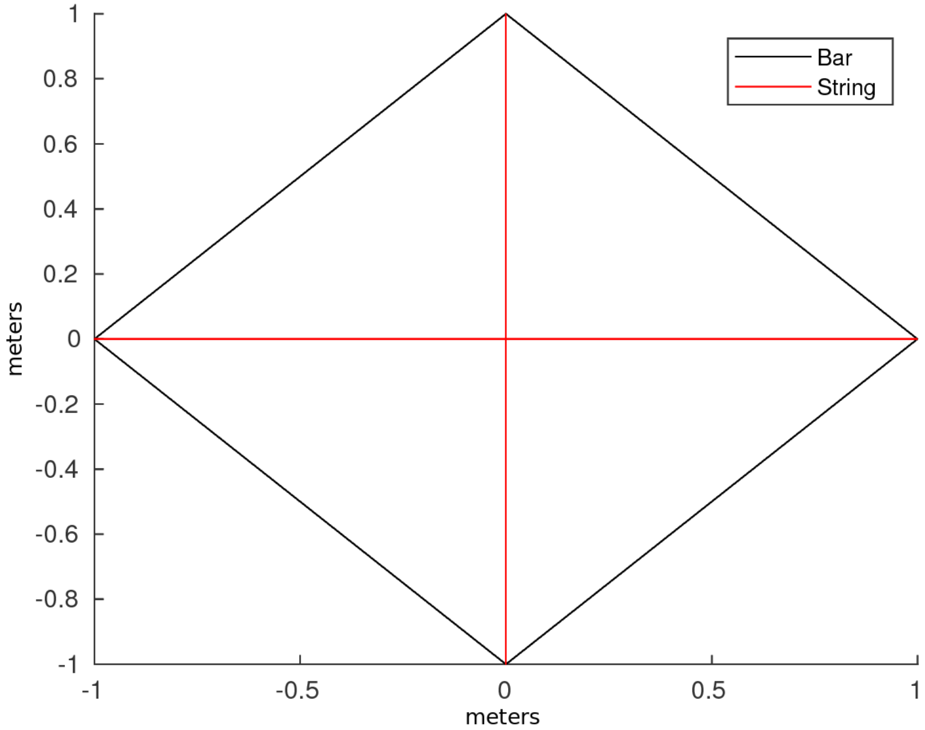

Figure 1.2: A Class 2 Tensegrity Dbar.

We define the term ’class’ as number of bars connected to a single node. If there exists no bar connected to another bar at any node, we call it a class 1 tensegrity structure. An example of a class 1 tensegrity structure is shown in Fig. 1.1. If 2 bars are connected to each other at one node, as shown in Fig. 1.2, it is called a class 2 tensegrity structure. Similarly, if k bars are connected to each other at the same node, it is called a class k tensegrity structure. Tensegrity structures can also be constrained in their motion at particular nodes : For example, constraining the planar Dbar to be pinned to the ground at one node. Bars are assumed to be connected to other bars using massless spherical joints.

1.1.2 Why Tensegrity?

Tensegrity structures are well known for their minimal mass and minimal control energy properties. In his book titled 'Tensegrity Systems' [1], Dr.Robert Skelton elaborates on this topic, introducing tensegrity structures that are mass efficient in compression, tension and torsional loads. In section 1.3 of his book, a graph explaining the several equilibrium surfaces of tensegrity is discussed. Tensegrity structures offer several equilibrium points throughout its configuration span, as shown in the paper by Dr.Skelton and his students [2]. When controlling the shape of tensegrity, the control only provides energy to steer between equilibrium points to reach the target configuration, so the total control energy needed to perform the maneuver is much less compared to traditional robot mechanisms that do not posses similar equilibrium properties as a tensegrity structure.

Tensegrity structures are also efficient in taking compressive and tensile loads. The book[1] introduces several mass efficient static designs for compressive, cantilevered and torsional loads.

One important advantage of Tensegrity structures explored in this thesis is the deployability property. Some Tensegrity structures can be compressed and stowed into a compact package, and later deployed when necessary. An example of a tensegrity structure, showing the deployability property is the ADAM collapsible truss structure [3]. This property opens applications for tensegrity robots in space, where there is a requirement for minimal mass and minimal control energy while also providing deployability : Pointing solar arrays and telescopes, deployable space antennas, growable and deployable space habitat systems, minimal mass towers in both terrestrial and extra terrestrial environments, etc.

1.1.3 Self-Similar Tensegrity

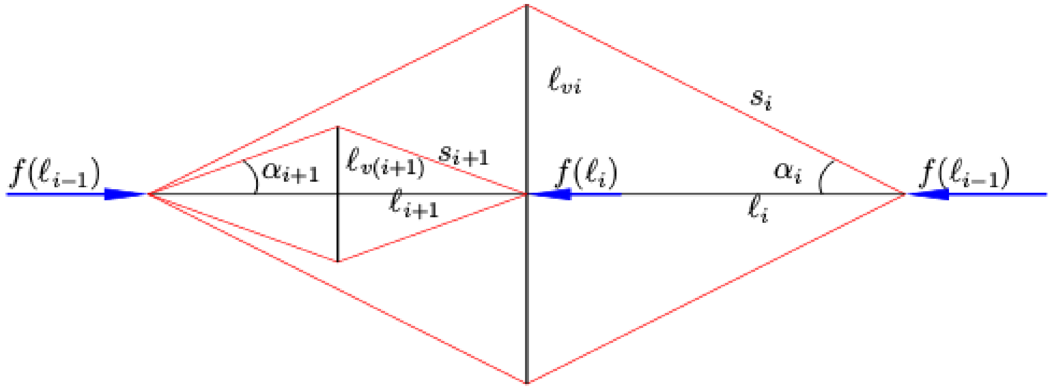

Figure 1.3: T-Bar self similar iteration.

Tensegrity structures like the Dbar and Tbar are efficient in saving mass for the given external compressive load. This is possible by replacing bars in the single Dbar or Tbar units with Dbars and Tbars. This can be done several times until we reach a minima for mass savings. This process of iteratively replacing bars or strings in a structure with Tensegrity structures is called a self-similar iteration. An example of Tbar self similar iteration is shown in Fig. 1.3.

1.2 Parallel and Serial Manipulators

A parallel manipulator is a mechanical system that consists of closed loop chain of linkages to actuate or support a platform / end effector. The main difference between a parallel and a serial manipulator lies in the fact that the end effector in a parallel mechanism is directly connected to the base through several independent linkages forming a loop. In a serial manipulator, the end effector (platform) does not belong to such a loop.

Serial manipulator drawbacks

Most robot applications require accuracy and precision. Serial manipulators achieve this by using high precision encoders at each link, connected to the actuators. A serial manipulator's

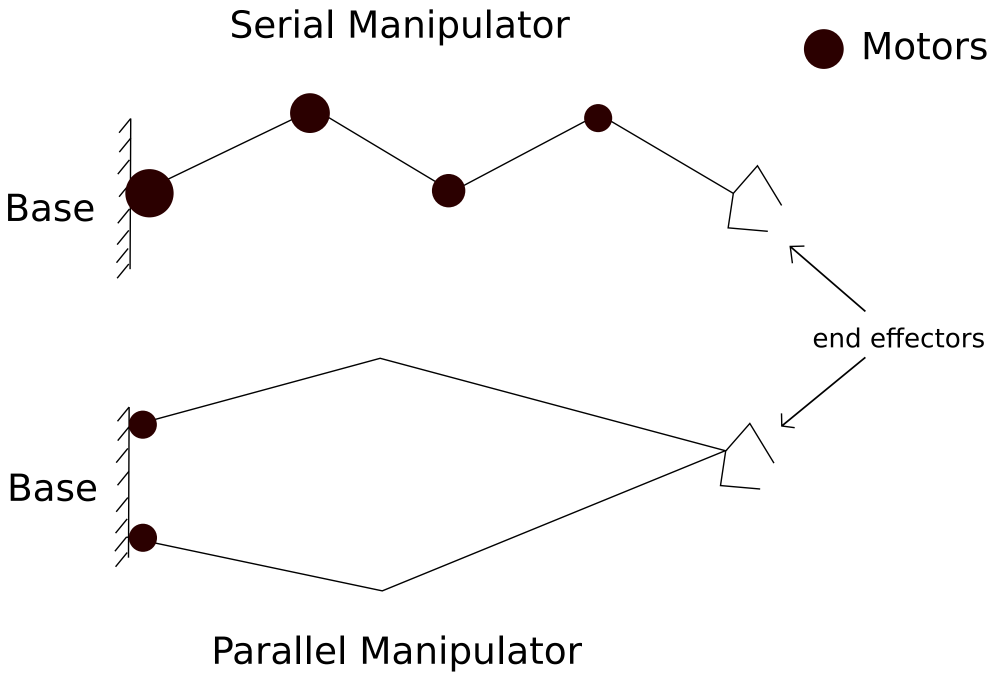

Figure 1.4: Serial and Parallel manipulators.

mechanism may not be the best way to build a robot arm as each motor supports the weight of the link as well as the motors connected to it. This issue is illustrated in Fig. 1.4. A parallel linkage overcomes this issue by placing motors at the base, from which the linkages are connected. This ensures the motors do not bear the weight of other motors as well. This property also helps design light weight parallel manipulators and as a result, parallel manipulators have faster end effector velocities than serial manipulators.

Also, each actuator in a serial manipulator has an associated resolution that affects its local precision. This loss in precision gets accumulated at each linkage reaching the end effector, thus lowering its overall precision as the number of linkages are increased. This is easily overcome in a parallel manipulator as it forms a closed loop chain.

Applications of Serial and Parallel Manipulators

Parallel manipulators are used in flight simulators, like the one shown in Fig.1.6 to train pilots in a simulated environment. A delta robot, as shown in Fig.1.8 is a parallel manipulator used in 3D printing technology, packaging industries for pick and place, manufacturing industries for fitting nuts, bolts and assisting humans in assembling heavy auto parts. Serial manipulators have also found similar applications as parallel manipulators. Fig. 1.7 shows a serial manipulator attaching auto parts for a Tesla model S at the Tesla Factory. Serial manipulators are currently gaining popularity in hospitals to assist in surgeries. DaVinci robot, shown in Fig. 1.5 is one such robot, well known among surgeons for performing robot assisted surgeries.

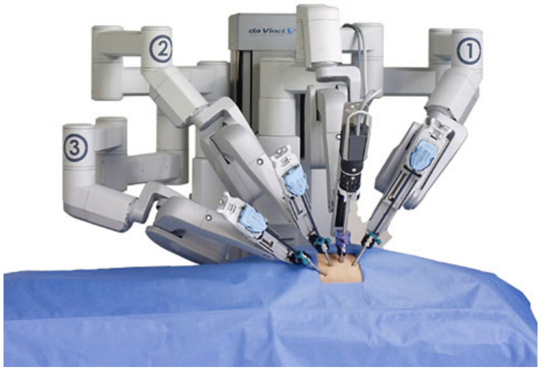

Figure 1.5: DaVinci the robotic arm, used in robot assisted surgeries.

1.3 Manipulators in space

In space, as pointed out earlier, there are a plethora of applications where robots are required for dynamic manipulation. Currently, the two most used manipulator designs are the Canfield joint shown in Fig.1.9 and the Stewart platform, illustrated in Fig. 1.10. These two manipulator mechanisms fall under the parallel manipulator category and are extensively used in various space probes and satellite missions [4] around the world.

1.3.1 The Canfield Joint

The canfield mechanism, invented by Dr.Stephen Canfield is a 3 degrees of freedom manipulator. It is a pointing mechanism that provides full hemispherical motion from its mounted base. It is mainly used in spacecrafts for pointing solar arrays towards the sun to improve the charging profile and for pointing thrusters while firing to reorient or accelerate the spacecraft. The canfield

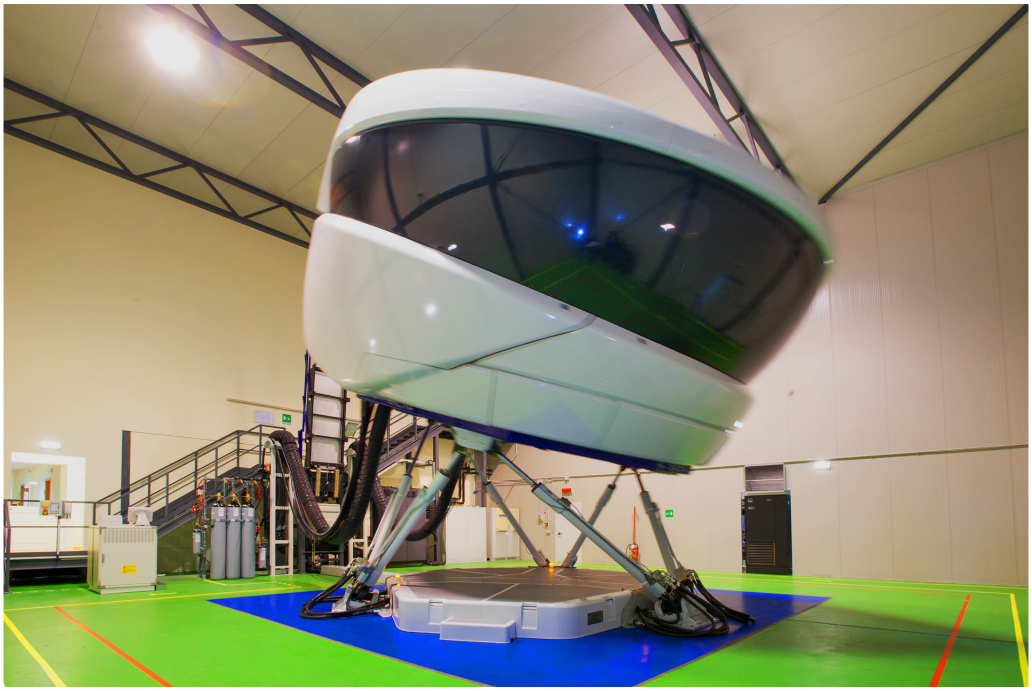

Figure 1.6: A parallel robot used as a flight simulator at SuperJet International.

mechanism has 9 joints and 6 bars overall in its empirical structure as shown in Fig. 1.11.

1.3.2 The Stewart Platform

Degrees of Freedom

The stewart platform is a 6 degrees of freedom spatial manipulator. It consists of 6 bars and 18 joints. Fig. 1.10 shows a rendering of a stewart platform mechanism. The 6 bars that also act as linear actuators are attached to the base using a 2DOF joint. each bar has 3 DOF which leads to 18 Degrees of freedom excluding the platform. When considering the top platform, the distances between any 2 joints on the platform is constant, and that introduces 6 constraints. Also, the top platform does not allow any 2 adjacent nodes to move out of plane, which introduces additional 6 degrees of freedom. In total, that leads to 6 degrees of freedom for the stewart platform.

Applications

It has been used as a reliable mechanism in pointing applications of ground antennas and telescopes towards the sky. A stewart platform is also used in the automobile industry to perform

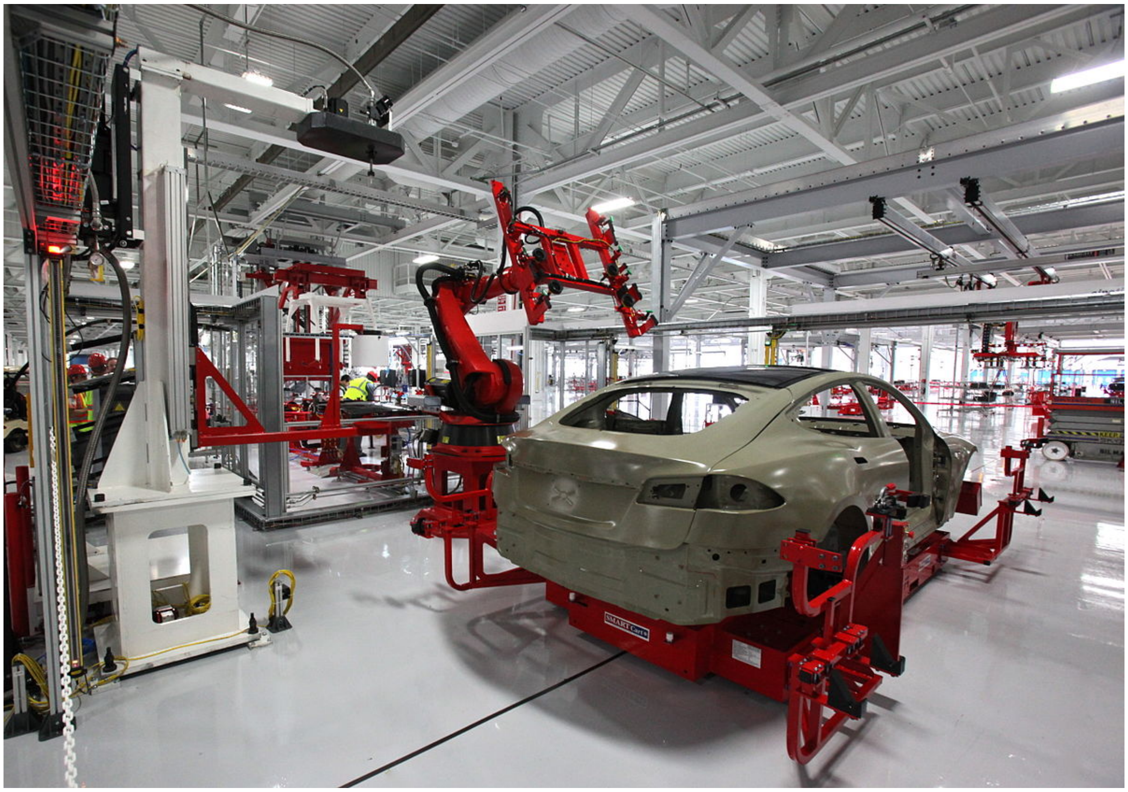

Figure 1.7: A Serial Manipulator Robot (in red) fitting Tesla S auto parts at Tesla Factory.

vibration tests on discrete vehicle components. The AMiBA space radio telescope (Fig. 1.12) in Mauna Loa, Hawaii, USA, uses a stewart platform to change its pointing orientation. The hexapod platform, known for its reliability, accuracy and precision is also used as a surgical robot[1] as shown in Fig. 1.13.

1.3.3 Drawbacks with standard manipulators

The two most used manipulators Stewart Platform(SP) and Canfield Joint(CJ) utilize more number of rigid bars(6 in CJ and SP) than necessary to produce as many degrees of freedom(3 in CJ and 6 in SP). Each bar contributes to increased mass of the overall mechanism. More the number of bars, more will be the number of joints needed to connect the bars to either the base, platform or to other bars. It is detrimental to have too many joints in a robot. Joints are prone to failures due to vibration and sudden impulses and such mechanisms are not immune to joint

[1] `http://www.hexapods.net/special_hexapods.php/`

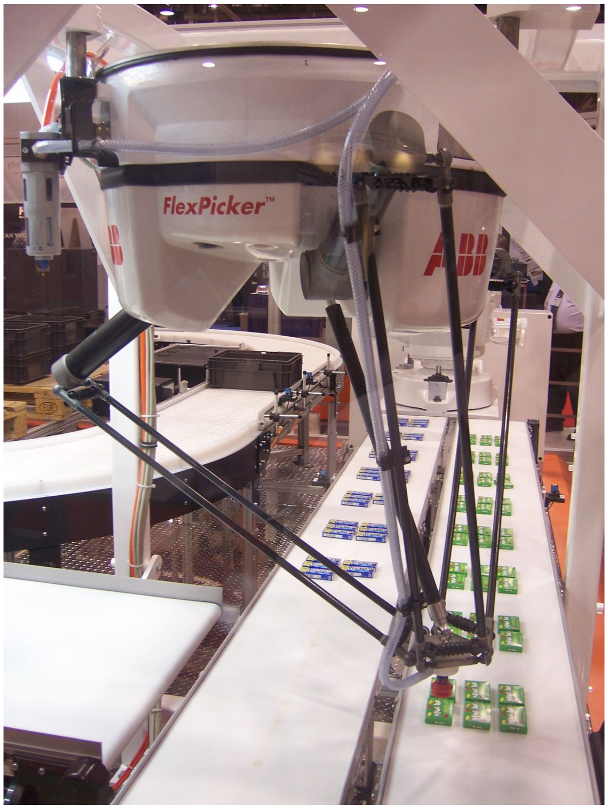

Figure 1.8: An ABB Delta robot used in industry for sorting goods.

failures. Also, joints at every segment leading to the end effector contribute to backlash error. It was shown in [5] that joint backlash as a result of wear due to severe stress imposed on the transmission system degrades the robot performance.

Actuator failure is another reason that can lead to catastrophic results. Neither of the two joints have inherent provision for introducing redundant actuators in their mechanisms.

Also, the canfield mechanism is not designed to withstand torsional load. The 3 degree of freedom joints that connect its top bars to the bottom bars could fail for twisting motion that can occur due to unforeseen torsional loads on the platform.

Parallel manipulators are known to suffer from kinematic singularities. The paper [6] focuses on a kinematic treatment for singularities arising in closed loop kinematic chains, a structural trait of the parallel manipulator class. This paper shows that when a condition arises in which the jacobian matrix that defines the kinematic structure of the closed loop chain relating the motor

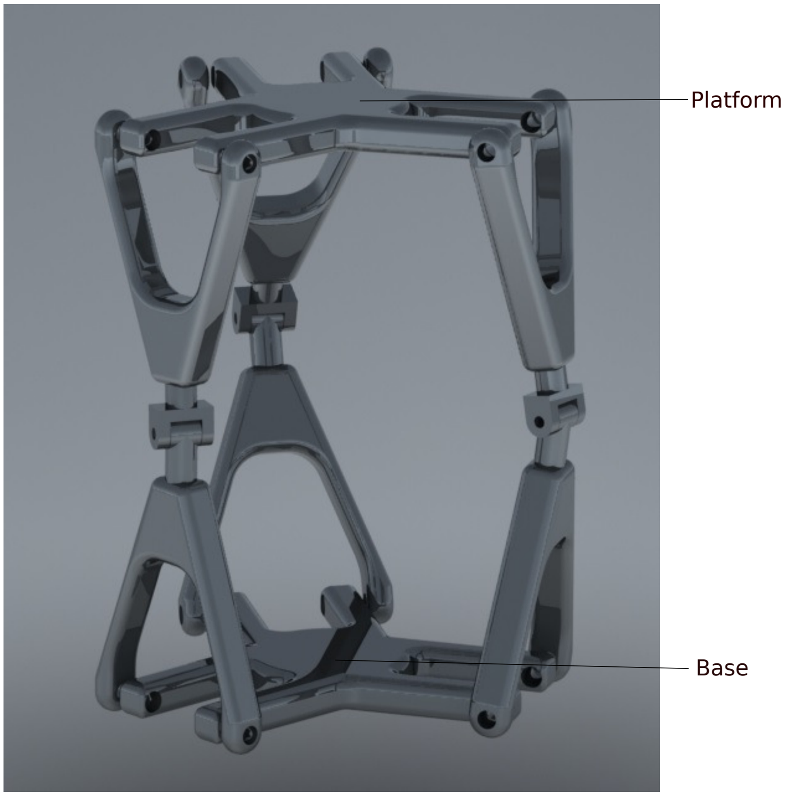

Figure 1.9: The Canfield Mechanism.

inputs to the output (cartesian coordinates) of the gripper becomes singular, then a singularity arises and the mechanism is rendered useless / the mechanism loses complete controllability of the end effector. This behaviour is common among parallel manipulators, and that is why, parallel manipulator mechanisms are prone to singularities in their configuration space.

1.4 Tensegrity solutions for designing manipulators

Tensegrity structures are ideal candidates for designing minimal mass, minimal control energy manipulators. There exists tensegrity structures, like the 3 strut Tensegity prism and the 3D TD bar mechanisms that can serve as potential candidates for a Tensegrity based manipulator design.

1.4.1 The Tensegrity Prism Manipulator

The Tensegrity prism, shown in Fig. 1.14 is a potential tensegrity manipulator mechanism that reduces the number of joints used while also providing capability for adding redundancy in

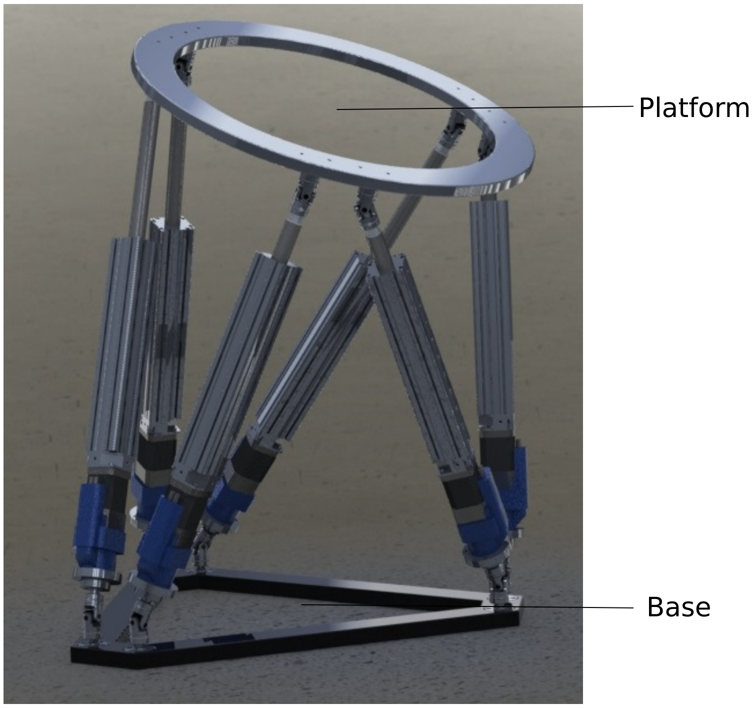

Figure 1.10: The Stewart-Platform Mechanism.

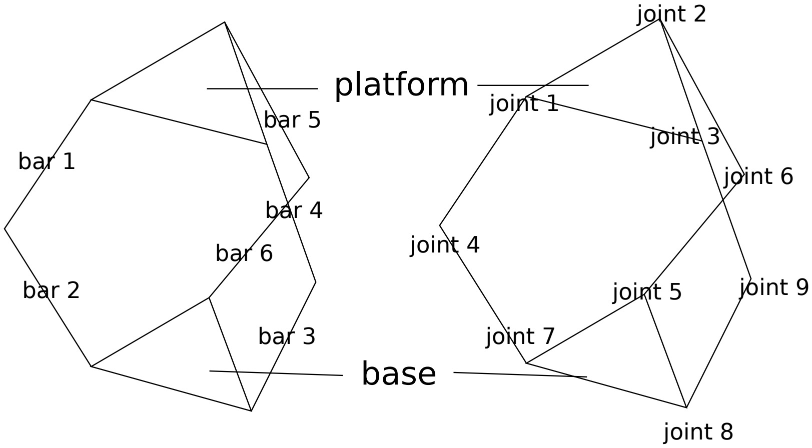

Figure 1.11: Empirical structure of Canfield mechanism labelling the 6 bars and 9 joints.

actuation. One can add redundant strings by connecting nodes that were not connected directly with a string, as shown in Fig. 1.15 - in this structure, 3 more strings are introduced between adjacent top and bottom nodes leading to a total of 12 strings.

A minimal tensegrity prism can have p bars in a single unit. In Chapter 3 of his book, [1],

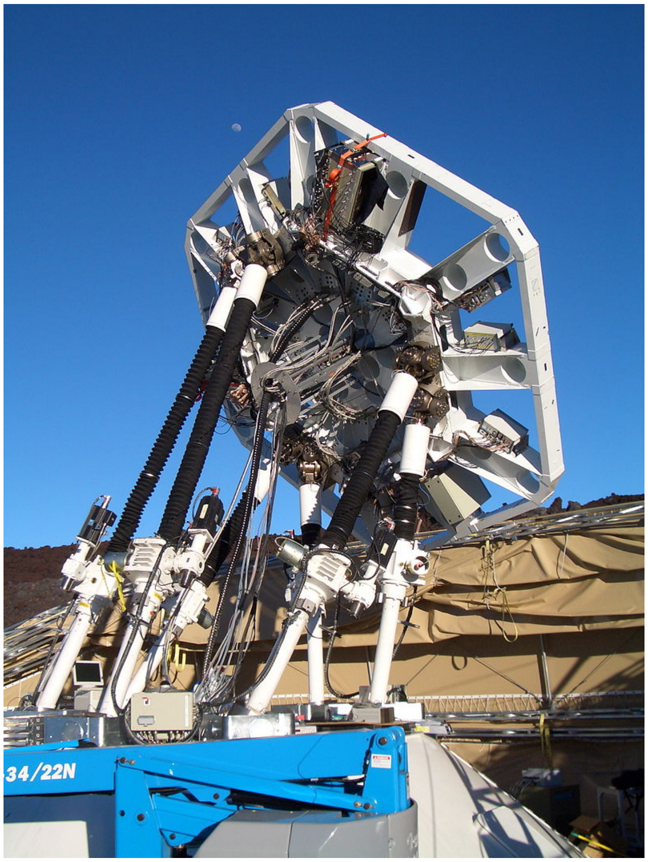

Figure 1.12: AMiBA 2 space radio telescope mounted on a Stewart platform.

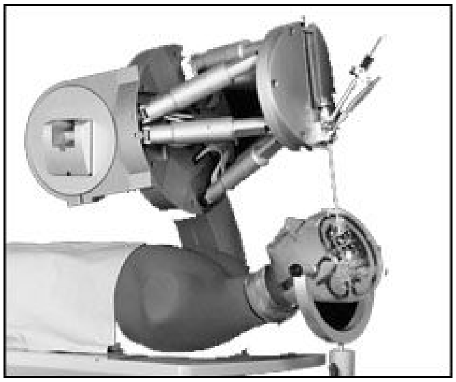

Figure 1.13: Hexapod robot used in robot assisted brain surgery.

Dr.Skelton explains the idea of building a tensegrity tower using stacks of tensegrity prisms in each layer. There exists an optimal number of such units to produce a minimal mass tower that

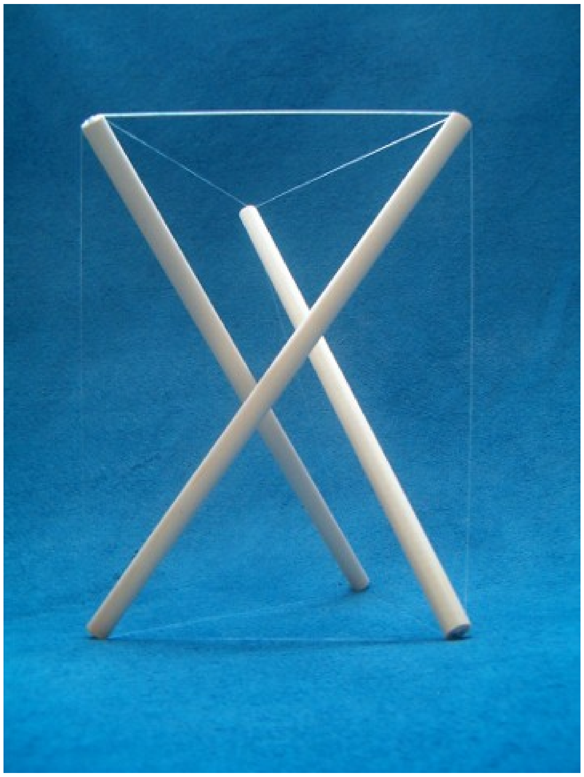

Figure 1.14: 3 bar 9 string tensegrity prism mechanism.

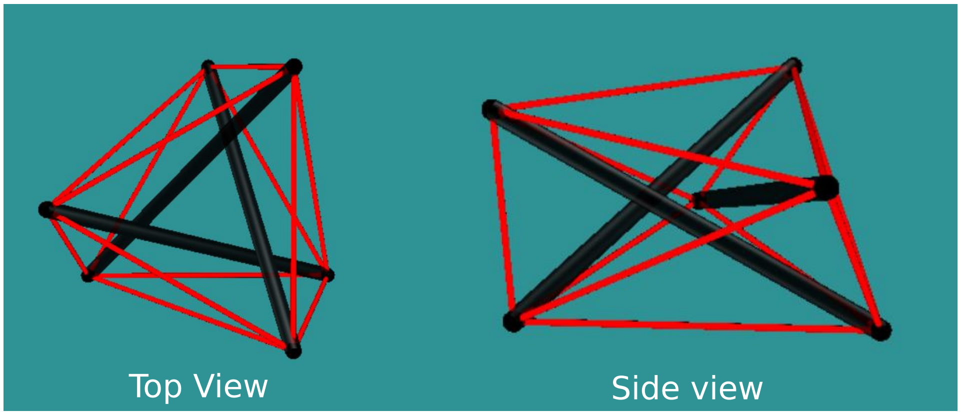

Figure 1.15: 3 bar 12 string tensegrity prism mechanism.

could take torsional load. A tensegrity prism is an ideal structure to withstand torsional loads, unlike the canfield mechanism.

A regular 3 bar prism has its top and bottom planes parallel. While the radius of the top and bottom triangles could be different, a stable equilibrium configuration is possible when the angle between the two triangles is exactly equal to 30 degrees. At this configuration, the force densities in the top 3 strings and bottom 3 strings are equal. Also, the vertical 3 strings all have the same force densities. This is true when there is no external force applied on the structure. However, if we let the force densities in the string take any value and apply external forces, we can achieve a plethora of different possible equilibirium configurations with the 3 bar structure.

The 3 bar structure has 0 translational degrees of freedom and 3 rotational degrees of freedom per bar. We restrict the rotation about its long axis for each bar, so in total, the prism manipulator has 6 degrees of freedom.

A tensegrity prism structure can provide redundancy in actuation by adding 3 extra strings, as in a 12 string tensegrity prism. However, to act as a manipulator, we will fix the bottom 3 nodes of the prism to the ground using 2 DOF joints each and remove the bottom 3 strings. The top triangle acts as the platform for manipulation. Moreover, unlike a rigid platform as in a canfield or stewart platform, the tensegrity prism has a triangular platform that can change its area. A tensegrity prism as a manipulator leads to two different configurations : The 3 bar 6 string tensegrity prism manipulator and the 3 bar 9 string tensegrity manipulator. One can also add more bars in this prism design, leading to several other configurations. In this thesis, we will only consider the minimalistic 3 bar 6 string tensegrity manipulator design and its control. An illustration of the mechanism is provided in Fig. 1.16.

1.4.2 The 3D T-D bar as a Tensegrity Manipulator

The T-Dbar tensegrity structure was introduced in Dr.Skelton's book [1]. It comprises of a Tbar of complexity n, with the last iterations replaced with a Dbar. A planar version of this structure is shown in Fig. 1.17. A 3D version of a complexity 1 TD bar is shown in Fig.1.18. In this tensegrity manipulator, the platform is a rigid surface unlike the prism manipulator. Since this manipulator is controlled using a tendon actuator, it is different from the canfield joint, but the empirical structure remains the same.

The 3D TD bar is also a tensegrity based manipulator design but does not minimize the number

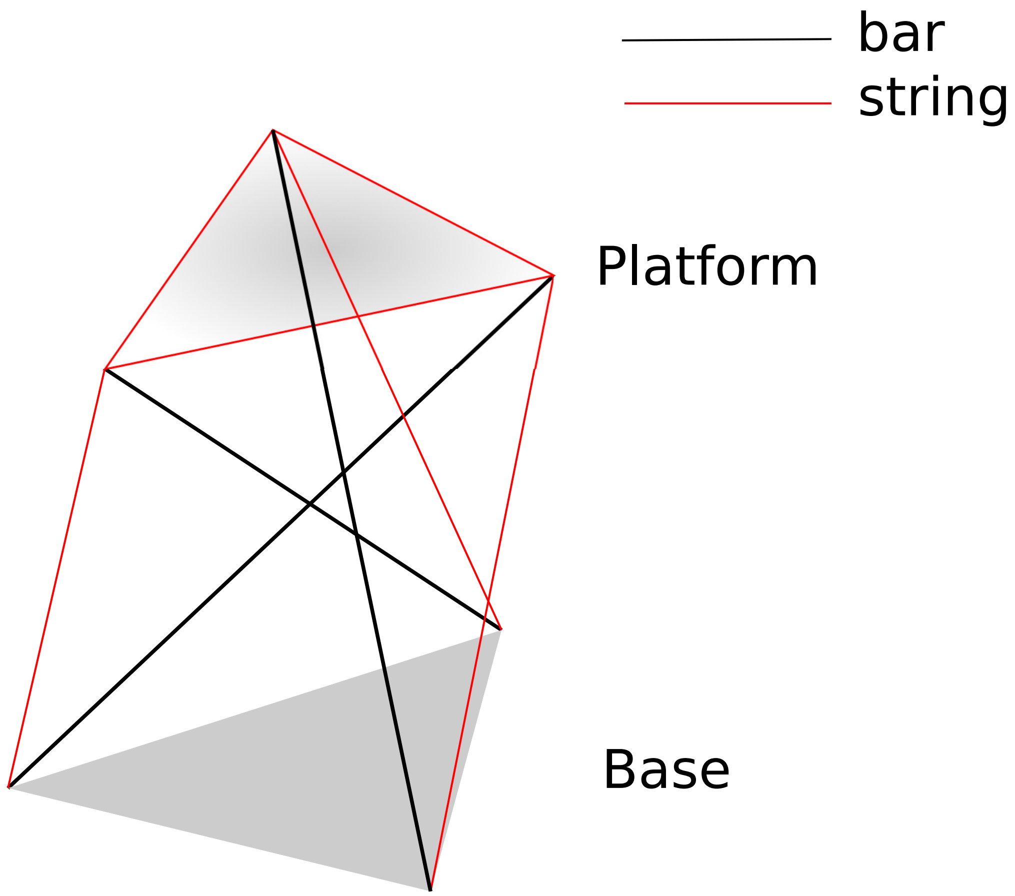

Figure 1.16: Empirical structure of the 3 bar 6 string Tensegrity prism manipulator.

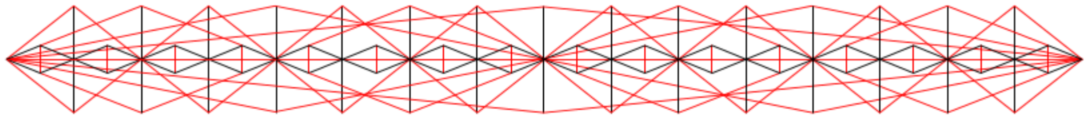

Figure 1.17: Constant width T-D bar.

of joints used. This is because it has a total of 9 spherical joints in its mechanism which is the same number of joints as a canfield carpal wrist mechanism. Moreover, it uses the same number of bars as a canfield or a stewart platform.

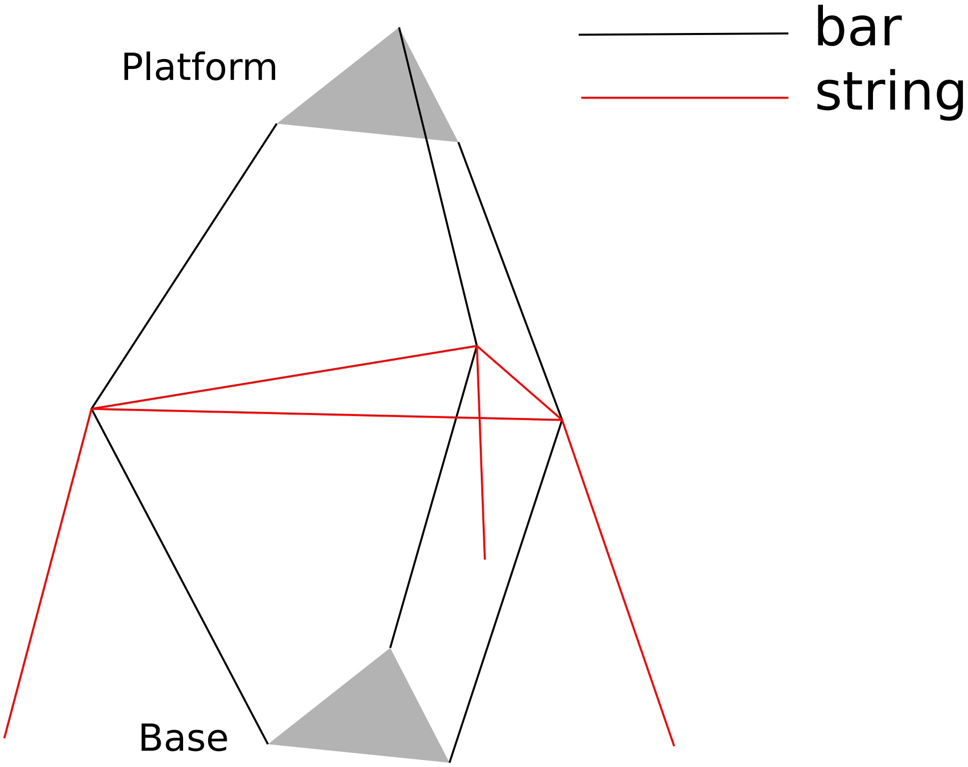

Figure 1.18: Empirical structure of a 3D T-Dbar manipulator mechanism.

1.5 Conclusion

From the above points, it is evident that either the 3 bar 6 string or a 3 bar 9 string tensegrity prism can best serve as an alternative mechanism for the canfield or stewart spatial manipulators. The following sections will focus on capturing the dynamics of this tensegrity structure and its attitude control.

2. DYNAMICS OF CONSTRAINED TENSEGRITY STRUCTURES

2.1 A rigid bar in 3D space

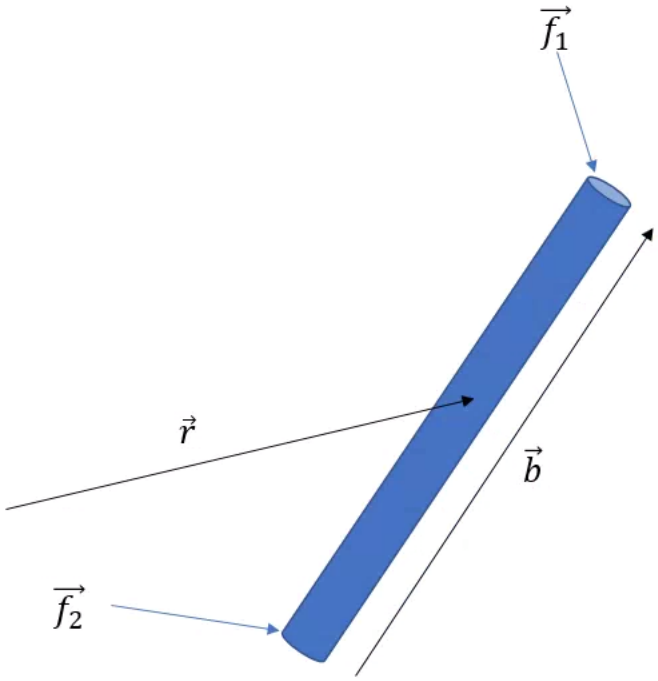

Figure 2.1: A rigid bar in 3D space acted by external forces.

Fig. 2.1 shows a rigid bar in 3D space acted by external forces $\vec{f_1}$ and $\vec{f_2}$ at its two ends. A rigid body is a solid whose individual particles donot move with respect to each other, or the movement is too small to be neglected. Such a rigid body has 6 degrees of freedom : 3 translational degrees and 3 rotational degrees of freedom.

2.2 Dynamics of a Rigid Rod

From Dr.Skelton's approach to derive the dynamics of a rigid rod in non minimal coordinates [7], we know that the angular momentum vector of a rigid rod in 3D space is given by

$$\vec{h} = \frac{m}{12}\vec{b} \times \dot{\vec{b}}$$

in inertial coordinates, $\bar{E} = [\vec{e_1}\vec{e_2}\vec{e_3}]$, $\vec{h} = \bar{E}h^E, \vec{b} = \bar{E}b^E$

dropping the superscript E, we get,

$$h = \tilde{b}\dot{b}\frac{m}{12}$$

From angular momentum and Newton's second law, $\dot{\vec{h}} = \vec{\tau}$ and $\vec{\tau} = \frac{1}{2}\vec{b} \times (\vec{f_2} - \vec{f_1}) = \frac{1}{2}\tilde{b}(f_2 - f_1)$ so, finally,

$$\dot{h} = \frac{m}{12}(\dot{\tilde{b}}\dot{b} + \tilde{b}\ddot{b}) = \frac{m}{12}\tilde{b}\ddot{b} = \frac{1}{2}\tilde{b}(f_2 - f_1) \tag{2.1}$$

And,

$$m\ddot{r} = f_1 + f_2 \tag{2.2}$$

$$b^T b = l^2, b^T\dot{b} = 0, b^T\ddot{b} = -\dot{b^T}\dot{b} \tag{2.3}$$

Adding the rod length constraints 2.3 to the bar dynamics equation 2.1, we get

$$\begin{bmatrix} \tilde{b} \\ b^T \end{bmatrix} \ddot{b} = \begin{bmatrix} \frac{12}{m}\frac{1}{2}\tilde{b}(f_2 - f_1) \\ -\dot{b^T}\dot{b} \end{bmatrix} \tag{2.4}$$

where

$$\begin{bmatrix} \tilde{b} \\ b^T \end{bmatrix}^+ = \frac{1}{l^2}\begin{bmatrix} -\tilde{b} & b \end{bmatrix} \tag{2.5}$$

A unique solution to x in the $Ax = b$ problem exists when A has full column rank. From the structure of the A matrix, we can see that A will always have a full column rank as its columns are independent. Thus, we get a unique solution for $\ddot{b}$,

$$\ddot{b} = \frac{6}{ml^2}(l^2 I - bb^T)(f_2 - f_1) - b\frac{\dot{b}^T\dot{b}}{l^2} \tag{2.6}$$

and $\ddot{r}$ as we know,

$$m\ddot{r} = f_1 + f_2 \tag{2.7}$$

2.3 Expressing the tensegrity dynamics in Matrix form

To simplify the equations for n_b number of bars and n_s strings, we can stack the vector form of the above dynamics equations into a matrix form. We introduce matrices B, S, N, C_b and $C_s, C_r, M, K, W, \Omega, P, D$ that denote the following:

B	Bar Matrix containing bar vectors stacked up in columns.
S	String Matrix containing string vectors stacked up in columns.
N	Node Matrix containing the node positions of the tensegrity structure
C_b	Connectivity Matrix of the bars.
C_s	Connectivity Matrix of the strings.
C_r	Connectivity Matrix of the center of mass of bars.
M	Mass matrix.
K	Stiffness matrix.
W	External force matrix.
Ω	Lagrange multiplier.
P	Constraint matrix.
D	Constrained node position matrix.

The B and S matrices are respectively,

$$B = \begin{bmatrix} \mathrm{b}_1 & \mathrm{b}_2 & \cdots & \mathrm{b}_{nb} \end{bmatrix}$$

$$S = \begin{bmatrix} \mathrm{s}_1 & \mathrm{s}_2 & \cdots & \mathrm{s}_{ns} \end{bmatrix}$$

The N matrix is the $2n_b$ nodes stacked in columns. Each row contains the x,y and z coordinates of the nodes respectively.

$$N = \begin{bmatrix} n_1 & n_2 & n_3 & \cdots & n_{2n_b} \end{bmatrix}$$

For the connectivity matrices C_b and C_s, the (i, j) th element of the matrix is defined as

$$[C_b]_{ij} = \begin{cases} -1, \text{if } b_i \text{ is directed away from the node } n_j \\ 1, \text{if } b_i \text{ is directed toward } n_j \\ 0, \text{if } b_i \text{ does not touch } n_j \end{cases}$$

$$[C_s]_{ij} = \begin{cases} -1, \text{if } s_i \text{ is directed away from the node } n_j \\ 1, \text{if } s_i \text{ is directed toward } n_j \\ 0, \text{if } s_i \text{ does not touch } n_j \end{cases}$$

Also, the W,T,F, C_r matrices are :

$$W = \begin{bmatrix} w_1 & w_2 & w_3 \cdots w_{2n_b} \end{bmatrix}$$

$$T = \begin{bmatrix} t_1 & t_2 & t_3 \cdots t_{ns} \end{bmatrix}$$

$$F = \begin{bmatrix} f_1 & f_2 & f_3 \cdots f_{2n_b} \end{bmatrix} = W - TC_s = W - S\hat{\gamma}C_s = W - NC_s^T\hat{\gamma}C_s$$

$$C_r = \frac{1}{2}\begin{bmatrix} I & I \end{bmatrix} \quad , \quad C_b = \begin{bmatrix} -I & I \end{bmatrix}$$

$$\begin{bmatrix} C_b^T & C_r^T \end{bmatrix}^{-1} = \begin{bmatrix} \frac{1}{2}C_b \\ 2C_r \end{bmatrix} \tag{2.8}$$

We can write equations 2.6 and 2.7 as following in matrix form :

$$\ddot{B}\hat{m}\frac{1}{12} + B(\lfloor \dot{B}^T\dot{B} \rfloor \hat{m}\widehat{l^{-2}}\frac{1}{12} + \lfloor B^T F C_b^T \rfloor \widehat{l^{-2}}\frac{1}{2}) = FC_b^T\frac{1}{2} \tag{2.9}$$

$$\ddot{R}\hat{m} = 2FC_r^T \tag{2.10}$$

Stacking the two equations 2.9 and 2.10 in matrices and using the property 2.8, we get :

$$\begin{bmatrix} \ddot{B} & \ddot{R} \end{bmatrix} \begin{bmatrix} \hat{m}C_b\frac{1}{12} \\ \hat{m}C_r \end{bmatrix} + \begin{bmatrix} B & R \end{bmatrix} \begin{bmatrix} -\hat{\lambda}C_b \\ 0 \end{bmatrix} = F \tag{2.11}$$

where,

$$-\hat{\lambda} = \lfloor \dot{B^T}\dot{B} \rfloor \hat{m}\widehat{l^{-2}}\frac{1}{12} + \lfloor B^T F C_b^T \rfloor \widehat{l^{-2}}\frac{1}{2} \tag{2.12}$$

by using the coordinate transformation, as in [7],

$$N = \begin{bmatrix} B & R \end{bmatrix} \begin{bmatrix} C_b^T & C_r^T \end{bmatrix}^{-1}$$

and using the relation between external forces and force densities in the string

$$F = W - S\hat{\gamma}C_s$$

we get the dynamics of a class 1 tensegrity structure,

$$\ddot{N}M + NK = W \tag{2.13}$$

where

$$M = C_b^T \hat{m} C_b \frac{1}{12} + C_r^T \hat{m} C_r$$

$$K = C_S^T \hat{\gamma} C_s - C_b^T \hat{\lambda} C_b$$

Similary, for class K tensegrity structures, We introduce the linear constraint equality as in [8], given by

$$NP = D \tag{2.14}$$

and the dynamics equation 2.13 becomes

$$\ddot{N}M + NK = W + \Omega P^T \tag{2.15}$$

It is to be noted that, as mentioned in the paper by Dr.Skelton and Joono Cheong [8], we express the class K system as a class 1 system by introducing virtual nodes at the connection points between bars. This is done to modify the class 1 dynamics to account for constrained systems instead of reinventing the wheel.

2.4 Reduced form of class K Dynamics

The above class K dynamics equation 2.15 along with the constraint equation 2.14 can model any class K tensegrity configuration with linear constraints in the nodes. The linear constraints include a node pinned to a point or two bars attached to one node. It should be noted that the matrix dynamic equation contains the dynamics of all the nodes. This is not necessary for nodes that are pinned to the ground (no acceleration at pinned nodes) or virtual nodes present at the bar to bar connections(acceleration at virtual node will be equal to the acceleration at real node). Hence, it is advisable to reduce the form of the dynamics equation by introducing a coordinate transformation that captures only the dynamics of the nodes that are in motion. Hence, we introduce a new coordinate transformation $\eta = NU$,

$$\eta = \begin{bmatrix} NU_1 & NU_2 \end{bmatrix}$$

where, U_1 and U_2 are derived from the singular value decomposition of P matrix.

$$P = U\Sigma V^T = \begin{bmatrix} U_1 & U_2 \end{bmatrix} \begin{bmatrix} \Sigma_1 \\ 0 \end{bmatrix} V^T$$

The structure of the singular value matrix is so because the P matrix does not have redundant columns - We simply donot repeat the same constraints as another constraint while constructing the P matrix. Hence, as derived by Dr.Skelton and Joono Cheong in [8], we get the final reduced form of the dynamics equation for class K tensegrity to be :

$$NKM^{-1}U_1 - \Omega P^T M^{-1} U_1 = WM^{-1}U_1 \tag{2.16}$$

$$\ddot{\eta}_2 \tilde{M} + \eta_2 \tilde{K} = \tilde{W} \tag{2.17}$$

where,

$$\tilde{M} = U_2^T M U_2$$

$$\tilde{K} = U_2^T K U_2$$

$$\tilde{W} = WU_2 - \eta_1 U_1^T K U_2$$

These are the reduced form of the class K tensegrity dynamics equation that we will revisit for use in the shape control law. The first equation 2.16 is used to solve for the lagrange multiplier Ω using algebraic manipulations as detailed in [8]. It is to be noted that the lagrange multiplier term Ω is present in both the K term and in the equation itself. Once we calculate the Ω, it could be plugged in $\tilde{W}$ and $\tilde{K}$ for further simplification.

2.5 Why Non-minimal coordinates

The reason Non-minimal coordinates was chosen over minimal coordinates to derive the control is because it is easy to specify the target in terms of inertial coordinates x,y and z instead of angles in radians. Also, the control variable, as discussed in the later part of this thesis is force density in the strings, which appears linearly in the control. So, though the dynamics is nonlinear, the control variable appears linearly. We make use of this observation and derive the control law in the following section.

3. CONTROL OF TENSEGRITY STRUCTURES

3.1 Introduction

This section mainly focuses on the shape control law and pointing control of tensegrity systems. Shape control of both class 1 and class K tensegrity systems are derived and analyzed. In the later sections, pointing control of class K tensegrity system is derived and analyzed. It will be showed that both shape control and pointing control laws derived are effective in solving the control problem for a tensegrity prism configuration as a pointing device. The main objective of the control laws derived here are as follows :

Given a 3 bar tensegrity prism manipulator, find the control law to actuate the strings such that the top plate of the prism moves from an initial configuration to being orthogonal to a given vector $\vec{v}$. This research focusses on two ways to point the prism towards the given vector :

1. Solve an optimization problem to find the target nodes of the prism and then let the controller move the nodes to achieve the target configuration.

2. Controller moves the nodes directly to satisfy the pointing operation, without finding the nodes prior.

For easy reference, we name the two methods 'shape control' and 'pointing control' respectively. The two methods have its own advantages and disadvantages, which will be mentioned towards the end in the examples section.

3.2 Optimization to find the target position with minimum change in string length

In shape control of tensegrity, an optimization problem is solved that calculates the target node positions of the top 3 nodes in the tensegrity prism manipulator to satisfy some necessary constraints. The calculated node positions are fed to the controller that then drives the necessary nodes towards the goal. We begin by formulating the problem statement :

Given a 3 bar prism, what is the final target position from the initial given configuration, shown in Fig.3.6 (azimuth and elevation 90 degrees, initial height of the prism being 6.18 inches) and also satisfying the constraints that the rigid bars donot touch each other, while minimizing the change in string length needed to change the shape.

The problem could be written down in mathematical statements as follows :

$$min \quad J = tr(C_s N^T N C_s^T - C_s N_0^T N_0 C_s^T) \tag{3.1}$$

Subject to the following constraints :

$$N(C_s^T \hat{\gamma} C_s - C_b^T \hat{\lambda} C_b) = W, \qquad \begin{bmatrix} \gamma \\ \lambda \end{bmatrix} \geq 0 \tag{3.2}$$

$$\lfloor C_b N^T N C_b^T \rfloor = \hat{l}^2 \tag{3.3}$$

$$\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} N \geq 0 \tag{3.4}$$

$$v^T L N C_s^T R = 0 \tag{3.5}$$

$$NP = D \tag{3.6}$$

$$min(b_1, b_2), min(b_2, b_3), min(b_3, b_1) \geq d + \epsilon \tag{3.7}$$

where, N_0 is the initial configuration of the prism, C_s, C_b are the connectivity matrices of the bars and strings in the prism structure, $\hat{\lambda}, \hat{\gamma}$ are the force densities (force per unit length) in the bars and strings, $b_1, b_2, b_3, d, \epsilon$ are the vectrix of 3 bar vectors in inertial coordinates, diameter of each bar and minimum necessary distance between two bars respectively. tr() is the trace function that returns the trace of a matrix. The L,R,P,D matrices are discussed in further detail in the next sections and also in [8].

3.2.1 Explanation of the constraint equations

The function min() in Eq.3.7 is defined as the shortest distance between two bars. Consider 2 rigid bars, represented by skew lines b_1 and b_2 from nodes n_1 and n_2, with two points p_1 and p_2

being the points closest to each other between the two lines. A schematic is shown in Fig. 3.1. The two points p_1 and p_2 can be expressed in terms of n_1, n_2, b_1, b_2 as :

$$p_1 = n_1 + t_1 b_1$$

$$p_2 = n_2 + t_2 b_2$$

where t_1 and t_2 are unknown variables that define the position of the point in the line. Similarly, we know that the vector passing through the two points p_1 and p_2 is related to the perpendicular to the two vectors $\vec{b_1}$ and $\vec{b_2}$ by the relation :

$$p_1 - p_2 = t_3(\vec{b_1} \times \vec{b_2})$$

Expressing the above problem in terms of $Ax = b$, we get the following expression :

$$\begin{bmatrix} -b_1, & b_2, & \vec{b_1} \times \vec{b_2} \end{bmatrix} \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = \begin{bmatrix} n_1 - n_2 \end{bmatrix}$$

This gives us the two points p_1 and p_2 and thus, we can calculate the shortest distance between any 2 bars.

The Eqn. 3.2 ensures that the target node is a possible equilibrium configuration for the manipulator to exist in. This constraint is essential because a geometrically possible configuration for the 3 bars maynot necessarily be an equilibrium configuration to the prism. Eqn. 3.3 ensures the bar length of all the rods are satisfied. The top 3 nodes can only move on the surface of the three hemispheres, as shown in Fig.3.2, whose radius is defined by the length of each bar l. The Eqn. 3.4 ensures the optimization only searches for a possible configuration above the ground from where the nodes n_1, n_2 and n_3 are pinned to. Eqn. 3.5 ensures the final target configuration is one in which the top 3 strings of the prism are orthogonal to the given vector $\vec{v}$. Finally, Eqn. 3.6 is the constraint that contains information about the bottom 3 nodes pinned to their respective points in

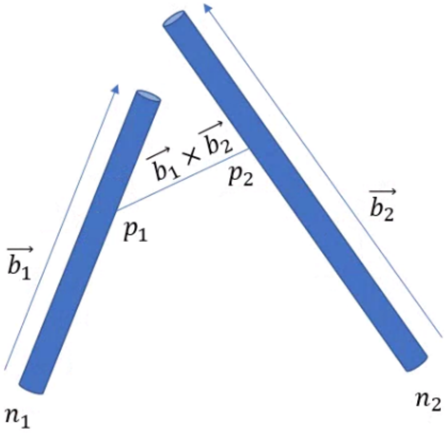

Figure 3.1: 2 skew lines (rigid bars) in 3D space.

the ground. In the examples considered in this document, we pin the bottom 3 nodes $n_{1\to3}$ on the vertices of an equilateral triangle of edge length 7.05 inches.

The above optimization problem yields the target positions of the top 3 nodes $n_{4\to6}$. It is unknown yet if this problem has an analytical solution to yield the results. To get a numerical solution to this problem, $fmincon$ function from the optimization toolbox in matlab can be used.

3.2.2 Feasible workspace of the tensegrity prism

It was shown in Dr.Skelton's book [1] that if the top and bottom plate of the prism is parallel, irrespective of the radii of the top and bottom triangles, the angle between the two triangles is always $30°$ for no external force acting on the system. If we **donot** consider the constraint that the faces are parallel, then there exists several equilibrium configurations for the prism. We define each configuration of the prism by the elevation and azimuth angles each bar takes with respect to the inertial axes x,y and z. To visualize the feasible workspace, we plot discrete values of azimuth ranging from $-30°$ to $60°$ and elevation ranging from $0°$ to $90°$ both having an increment of $10°$ steps, for all 3 bars. For the 10^6 geometrically possible points, equilibrium test is performed for no external load and the centroid of the top plate is plotted, if it is in a feasible equilibrium configuration. A snapshot of the top, side and isometric views of the workspace is shown in

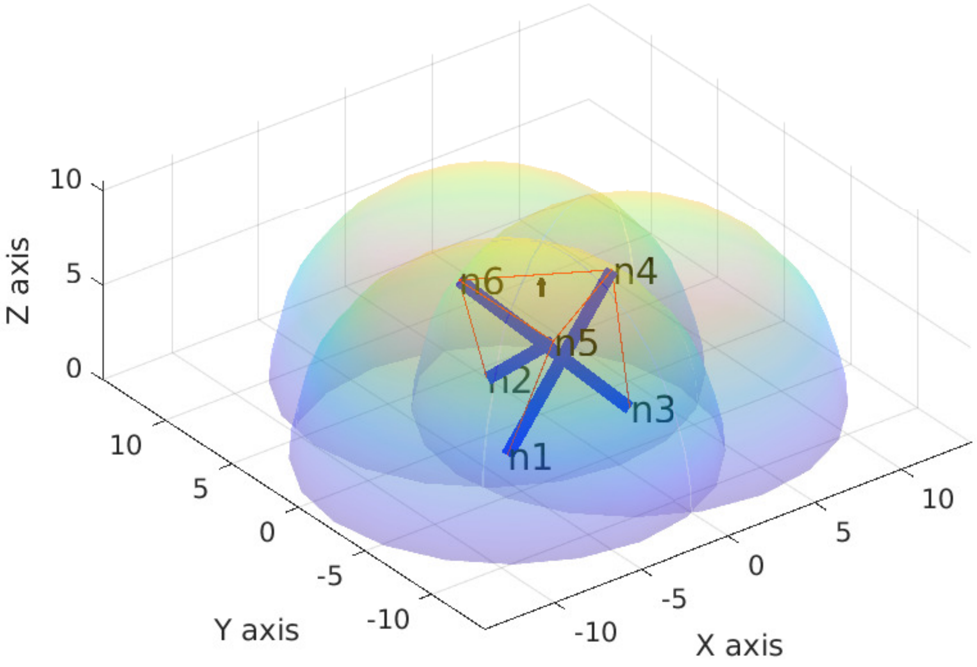

Figure 3.2: Bar length constraint for the prism, expressed as individual spherical surface boundaries.

Figures 3.3,3.4 and 3.5. These plots show that there are several possible dextrous equilibrium configurations for the tensegrity prism that can be traversed between to change the shape of the prism from one configuration to other.

3.3 Shape control of class 1 tensegrity systems

It is worth recollecting that a class 1 tensegrity structure is a network of prestressable bars and strings that have no bar to bar connection or one which has no constraints on any of the nodes. Such a system can be modelled by the dynamics equation :

$$\ddot{N}M + NK = W$$

Suppose we want some or all of the nodes of the tensegrity structure to move from its present

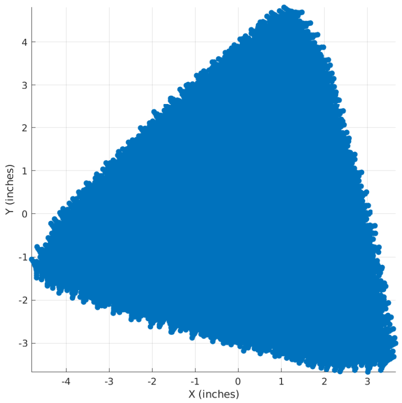

Figure 3.3: Top view of feasible equilibrium configurations of the centroid.

configuration to a target configuration, we write the objective as follows :

We define the error between the initial and the final configuration to be

$$E = LNR - N_t \tag{3.8}$$

Where the L matrix contains 0s and 1s. The structure is an identity matrix if all 3 axes x,y and z need to be controlled. We simply remove the rows corresponding to the axes that need not be controlled. For example, if y and z axes need to be controlled, the structure of the L matrix would be

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Similarly, the R matrix contains information about the nodes that need to be controlled. The structure is an identity matrix if all the nodes need to be controlled. If some nodes need not be controlled, then we simply remove those columns from the identity formulation. For example, out

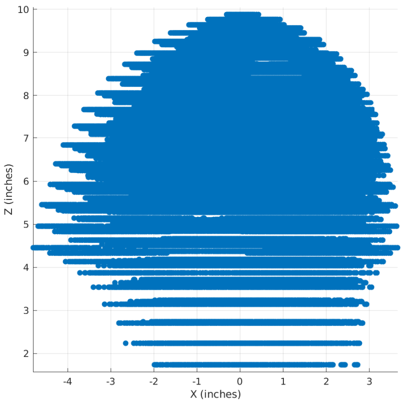

Figure 3.4: Side view of feasible equilibrium configurations of the centroid.

of 4 nodes in a tensegrity structure, we need to control the 1st, 2nd and 4th node, the R matrix would be :

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The N_t matrix contains information of the target positions for each coordinate of the nodes that need to be placed at the target.

The nature of the target configuration is not restricted to only position information, but velocity $\dot{N}_t$ and acceleration $\ddot{N}_t$ as well. However if one is only interested in position, we set the velocity and acceleration terms $\dot{N}_t$ and $\ddot{N}_t$ respectively to 0.

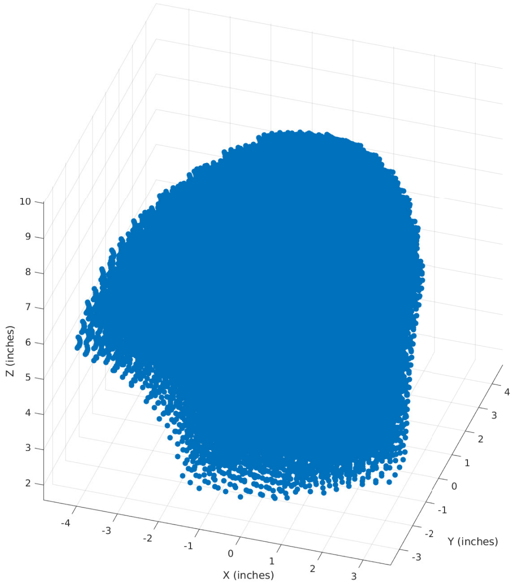

Figure 3.5: Isometric view of feasible equilibrium configurations of the centroid.

3.3.1 Derivation of the control law

To derive the control law, we formulate the second order error dynamics equation, given as :

$$\ddot{E} + A\dot{E} + BE = 0 \tag{3.9}$$

The A and B matrices are the coefficient matrices that dictate the controller damping and controller frequency of the second order system. It is not necessary for these two matrices to be diagonal, but for simplicity, we assume A and B to be diagonal matrices.

We substitute the error equation 3.9 with,

$$E = LNR - N_t$$

$$\dot{E} = L\dot{N}R - \dot{N}_t$$

$$\ddot{E} = L\ddot{N}R - \ddot{N}_t$$

We pick the velocity and acceleration terms $\dot{N}_t$ and $\ddot{N}_t$ to be 0. We get,

$$L\ddot{N}R + AL\dot{N}R + B(LNR - N_t) = 0$$

we then substitute the class 1 dynamics equation $\ddot{N} = (W - NK)M^{-1}$ into the previous equation and we get,

$$L(WM^{-1} - NKM^{-1})R + AL\dot{N}R + B(LNR - N_t) = 0$$

the control, which is the force densities in the string γ is in the K matrix of the equation, so we isolate it in one side and write the following equation :

$$LNKM^{-1}R = LWM^{-1}R + AL\dot{N}R + B(LNR - N_t)$$

We substitute H for the RHS of the previous equation.

$$LNKM^{-1}R = H$$

Expanding K and equating the i^{th} column of the matrix expression, we get :

$$LN(C_s^T\hat{\gamma}C_s - C_b^T\hat{\lambda}C_b)M^{-1}Re_i = He_i$$

We use the following property that the elements of a diagonal matrix and a vector multiplied to it can be interchanged :

$$\begin{bmatrix} a & 0 & 0 & \cdots \\ 0 & b & 0 & \cdots \\ 0 & 0 & c & \cdots \\ \cdots & & & \\ \cdots & & & \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ . \\ . \\ . \end{bmatrix} = \begin{bmatrix} x & 0 & 0 & \cdots \\ 0 & y & 0 & \cdots \\ 0 & 0 & z & \cdots \\ \cdots & & & \\ \cdots & & & \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ . \\ . \\ . \end{bmatrix}$$

we get,

$$LN(C_s^T(C_sM^{-1}Re_i)^\wedge\gamma - C_b^T(C_bM^{-1}Re_i)^\wedge\lambda) = He_i$$

Since we know λ is a linear function of γ,

$$-\hat{\lambda} = \lfloor \dot{B}^T\dot{B} \rfloor \hat{m} \hat{l^{-2}} \frac{1}{12} + \lfloor B^T(W - S\hat{\gamma}C_s)C_b^T \rfloor \hat{l^{-2}} \frac{1}{2}$$

we can express it in the form $-\lambda = \Lambda\gamma + \tau$ where Λ and τ are :

$$\Lambda = \begin{bmatrix} -\frac{1}{2l_i^2} b_i^T S(C_sC_b^Te_i)^\wedge \\ . \\ . \\ . \\ i = 1, ..., n_b \end{bmatrix}$$

$$\tau = \begin{bmatrix} \frac{1}{2l_i^2} b_i^T W C_b^T e_i + \frac{m_i}{12l_i^2}||\dot{b_i}||^2 \\ . \\ . \\ . \\ i = 1, , ..., n_b \end{bmatrix}$$

Hence, the control equation is now :

$$LN(C_s^T(C_sM^{-1}Re_i)^\wedge + C_b^T(C_bM^{-1}Re_i)^\wedge\Lambda)\gamma = He_i - LNC_b^T(C_bM^{-1}Re_i)\tau$$

The above equation is of the form $\Gamma\gamma = \mu$ where,

$$\Gamma = LN(C_s^T(C_sM^{-1}Re_i)^\wedge + C_b^T(C_bM^{-1}Re_i)^\wedge\Lambda)$$

$$\mu = He_i - LNC_b^T(C_bM^{-1}Re_i)\tau$$

The above linear algebra problem can be solved for $\gamma \geq 0$ by finding the least squares solution of the following problem using $lsqlin$ function in matlab :

$$min \quad \gamma \qquad ST \quad ||\Gamma\gamma - \mu||_2 \qquad \gamma \geq 0$$

3.4 Shape control of class K tensegrity systems

3.4.1 Derivation of the control law

A class K tensegrity structure has constrained nodes. It could be that 2 or more bars are connected at nodes, or also that some or all nodes are pinned to points. Both these conditions can be captured in the constraint equation :

$$NP = D \tag{3.10}$$

The definitions of the P and D matrices are as mentioned in [8]. P matrix contains the constraining relation between nodes and also of individual pinned nodes. D matrix contains information on where the nodes are pinned. The dynamics of class K tensegrity systems, derived earlier , is :

$$\ddot{N}M + NK = W + \Omega P^T \tag{3.11}$$

However , the dynamics also includes the constraint equation 3.10. So, it is necessary to reduce the order of the dynamics to consider only the nodes that are in motion. This is given by the reduced form of the dynamics equations :

$$NKM^{-1}U_1 - \Omega P^T M^{-1}U_1 = WM^{-1}U_1 \tag{3.12}$$

and ,

$$\ddot{\eta_2}\tilde{M} + \eta_2\tilde{K} = \tilde{W} \tag{3.13}$$

As mentioned earlier, the equation 3.12 is used to solve for the lagrange multiplier Ω which is then plugged in to equation 3.13. We know that

$$N = \eta U^T, \quad \dot{N} = \dot{\eta_2}U_2^T, \quad \ddot{N} = \ddot{\eta_2}U_2^T$$

substituting this in equation 3.9, we get,

$$L\ddot{\eta_2}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = 0$$

substituting the reduced form of the dynamics equation $\ddot{\eta_2} = (\tilde{W} - \eta_2\tilde{K})\tilde{M}^{-1}$ in the above equation, we get,

$$L(\tilde{W} - \eta_2\tilde{K})\tilde{M}^{-1}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = 0$$

$$L\tilde{W}\tilde{M}^{-1}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = L\eta_2\tilde{K}\tilde{M}^{-1}U_2^T R$$

$$L(WU_2 - \eta_1 U_1^T KU_2)\tilde{M}^{-1}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = L\eta_2 U_2^T KU_2\tilde{M}^{-1}U_2^T R$$

$$LWU_2\tilde{M}^{-1}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = L\eta_2 U_2^T KU_2\tilde{M}^{-1}U_2^T R + L\eta_1 U_1^T KU_2\tilde{M}^{-1}U_2^T R$$

$$LWU_2\tilde{M}^{-1}U_2^T R + AL\dot{\eta_2}U_2^T R + B(L\eta U^T R - N_t) = L\eta U^T KU_2\tilde{M}^{-1}U_2^T R \tag{3.14}$$

We define the matrix H as the LHS of the previous equation.

$$H = LWU_2\tilde{M}^{-1}U_2^TR + AL\dot{\eta}_2U_2^TR + B(L\eta U^TR - N_t)$$

Now, taking the i^{th} column and equating LHS and RHS of equation 3.14, we get

$$L\eta U^TKU_2\tilde{M}^{-1}U_2^TRe_i = He_i$$

$$L\eta U^T(C_s^T\hat{\gamma}C_s - C_b^T\hat{\lambda}C_b)U_2\tilde{M}^{-1}U_2^TRe_i = He_i$$

$$L\eta U^T(C_s^T(C_sU_2\tilde{M}^{-1}U_2^TRe_i)^\wedge\gamma - C_b^T(C_bU_2\tilde{M}^{-1}U_2^TRe_i)^\wedge\lambda) = He_i$$

In the previous expression, the $()^\wedge$ operator stands for an operation that sets the off-diagonal elements to zero and retains the diagonal elements. Lets define $M_i = U_2\tilde{M}^{-1}U_2^T$ and $-\lambda = \Lambda\gamma + \tau$ The variables Λ and τ are same as those defined in class 1 shape control.

$$L\eta U^T(C_s^T(C_sM_iRe_i)^\wedge + C_b^T(C_bM_iRe_i)^\wedge\Lambda)\gamma = He_i - L\eta U^TC_b^T(C_bM_iRe_i)^\wedge\tau$$

$$\Gamma\gamma = \mu \tag{3.15}$$

where,

$$\Gamma = L\eta U^T(C_s^T(C_sM_iRe_i)^\wedge + C_b^T(C_bM_iRe_i)^\wedge\Lambda)$$

$$\mu = He_i - L\eta U^TC_b^T(C_bM_iRe_i)^\wedge\tau$$

The equation 3.15 is of the form $Ax = b$. We find a solution for x by solving the least squares problem :

$$min \quad \gamma \qquad ||\Gamma\gamma - \mu||_2 \qquad , \gamma \geq 0$$

In Matlab, we can make use of the $lsqlin$ function to solve the above equation and get a least squares numerical solution for the control variable γ. The control variable which is force densities in the string, does not have any physical significance. But, we can convert this to rest length of the string itself and control the rest length of the string. To convert force densities into rest length, we make use of hooke's law :

$$t = k||s - s_0||, \quad s \geq s_0$$

where t, k, s, s_0 are the tension, stiffness, actual length and rest length of the string. Since force density $\gamma = \frac{t}{s}$, we can write the rest length of the string in terms of other variables as :

$$||s_{0_i}|| = ||s_i||(1 - \frac{\gamma_i}{k_i})$$

where i is the index of the string which varies from 1 to 6. The stiffness of the string can be calculated using the relation $k_i = \frac{EA}{||s_i||}$ where E is the Young's Modulus and A is the cross section area of the string.

3.5 Pointing control of class K tensegrity structure

In the previous control technique, an optimization problem was formulated that calculates the target position of the top nodes such that it points towards the given vector, while also satisfying other constraints. In this approach, the control problem itself is formulated to satisfy the orthogonality condition by introducing the constraint in the control equation. The error equation is formulated as :

$$E = v^T LNC_s^T R$$

Similarly, the first and second derivative of the error equation with respect to time would be, respectively,

$$\dot{E} = v^T L\dot{N}C_s^T R$$

$$\ddot{E} = v^T L\ddot{N}C_s^T R$$

Plugging the values of $E, \dot{E}, \ddot{E}$ and $N = \eta U^T, \dot{N} = \dot{\eta}_2 U_2^T, \ddot{N} = \ddot{\eta}_2 U_2^T$ in Eq. 3.9,

$$v^T L\ddot{\eta}_2 U_2^T C_s^T R + Av^T L\dot{\eta}_2 U_2^T C_s^T R + Bv^T L\eta U^T C_s^T R = 0 \tag{3.16}$$

Substituting the reduced order dynamics equation for class K tensegrity systems in the previous equation,

$$v^T L(\tilde{W} - \eta_2\tilde{K})\tilde{M}^{-1}U_2^T C_s^T R + Av^T L\dot{\eta}_2 U_2^T C_s^T R + Bv^T L\eta U^T C_s^T R = 0 \tag{3.17}$$

Substituting the values of $\tilde{M}, \tilde{K}$ and $\tilde{W}$, as defined in the previous section,

$$v^T L(WU_2 - \eta_1 U_1^T KU_2 - \eta_2 U_2^T KU_2)\tilde{M}^{-1}U_2^T C_s^T R + Av^T L\dot{\eta}_2 U_2^T C_s^T R + Bv^T L\eta U^T C_s^T R = 0 \tag{3.18}$$

$$\begin{aligned} v^T LWU_2\tilde{M}^{-1}U_2^T C_s^T R + Av^T L\dot{\eta}_2 U_2^T C_s^T R + Bv^T L\eta U^T C_s^T R = \\ v^T L(\eta_1 U_1^T K + \eta_2 U_2^T K)U_2\tilde{M}^{-1}U_2^T C_s^T R \end{aligned} \tag{3.19}$$

Further simplifying and assuming $M_i \leftarrow U_2\tilde{M}^{-1}U_2^T$ and

$$H \leftarrow v^T LWM_i C_s^T R + Av^T L\dot{\eta}_2 U_2^T C_s^T R + Bv^T L\eta U^T C_s^T R \tag{3.20}$$

$$v^T L\eta U^T(C_s^T\hat{\gamma}C_s - C_b^T\hat{\lambda}C_b)M_i C_s^T R = H \tag{3.21}$$

Equating the i^{th} columns of the LHS and RHS in the previous expression :

$$v^T L\eta U^T (C_s^T \hat{\gamma} C_s - C_b^T \hat{\lambda} C_b) M_i C_s^T Re_i = He_i \tag{3.22}$$

$$v^T L\eta U^T (C_s^T (C_s M_i C_s^T Re_i)^\wedge \gamma - C_b^T (C_b M_i C_s^T Re_i)^\wedge \lambda) = He_i \tag{3.23}$$

substituting $\lambda = -\Lambda\gamma + \tau$, where Λ and τ are same variables as defined in the previous section :

$$v^T L\eta U^T (C_s^T (C_s M_i C_s^T Re_i)^\wedge + C_b^T (C_b M_i C_s^T Re_i)^\wedge \Lambda)\gamma = He_i - v^T L\eta U^T C_b (C_b M_i C_s^T Re_i)^\wedge \tau \tag{3.24}$$

The above equation is of the form $\Gamma\gamma = \mu$ where,

$$\Gamma = v^T L\eta U^T (C_s^T (C_s M_i C_s^T Re_i)^\wedge + C_b^T (C_b M_i C_s^T Re_i)^\wedge \Lambda)$$

$$\mu = He_i - v^T L\eta U^T C_b (C_b M_i C_s^T Re_i)^\wedge \tau$$

and by solving numerically the linear algebra problem, similar to the previous section, the control γ that moves the structure to point the top plate towards a given vector can be calculated.

3.6 Examples

3.6.1 Shape control of tensegrity prism manipulator

The example of a 3 bar 6 string prism as a manipulator has been considered in this example. The prism, shown in Fig. 3.6 contains 3 rigid bars (in blue) of length 10 inches and diameter 0.5 inches each, pinned to the ground at one end with a 2 DOF joint. The other unpinned end of each bar is free to move along the surface of a hemisphere above the ground, dictated by a radius equal to the length of the bar. Let the configuration shown in Fig. 3.6 be the initial configuration of the prism. The configuration of the prism shown in Fig. 3.9 is the final configuration (position) of the prism. It should be noted that the initial configuration has an azimuth and elevation of $90°$

each, whereas the final configuration has an azimuth of $90°$ and elevation of $75°$. This could be calculated by connecting the centroids of the top and bottom triangular faces of the prism with a line and measuring the angles it makes with the inertial axes.

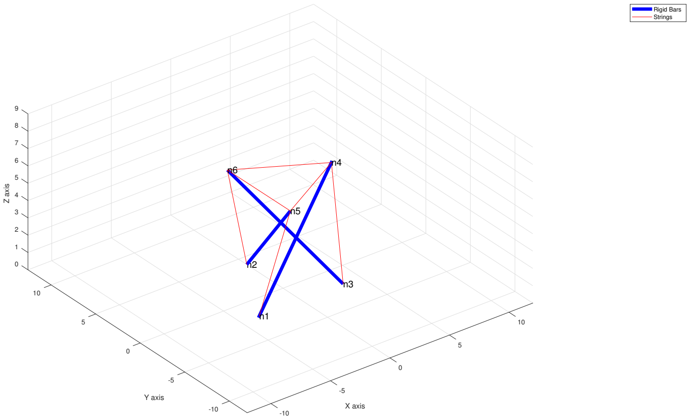

Figure 3.6: 3 bar 6 string prism manipulator initial configuration.

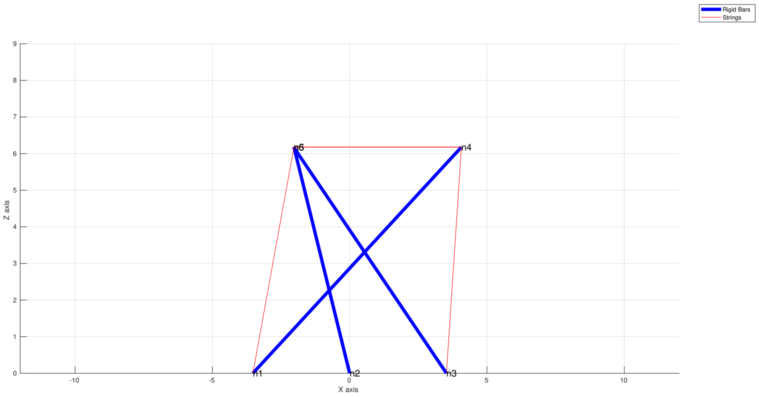

Figure 3.7: Side view of the initial configuration.

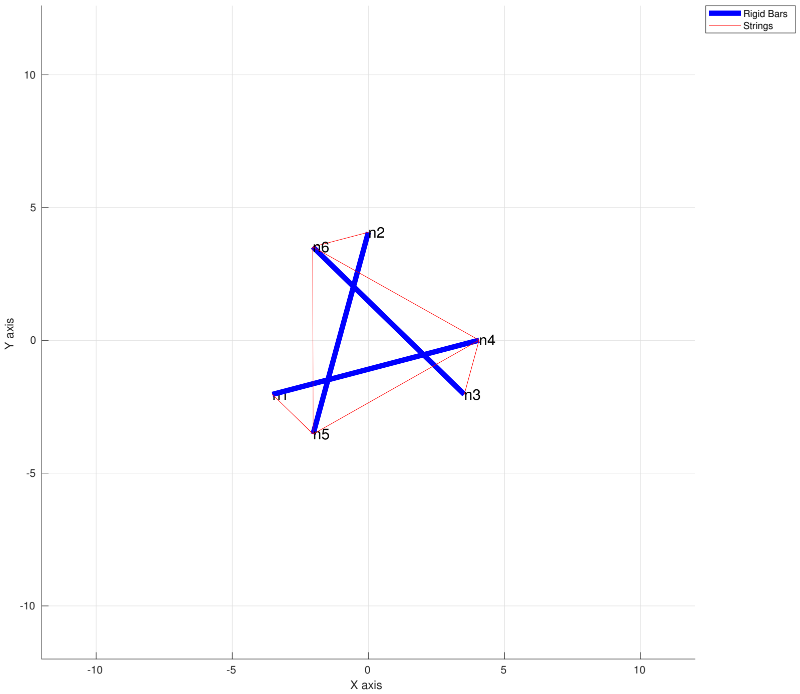

Figure 3.8: Top view of the initial configuration.

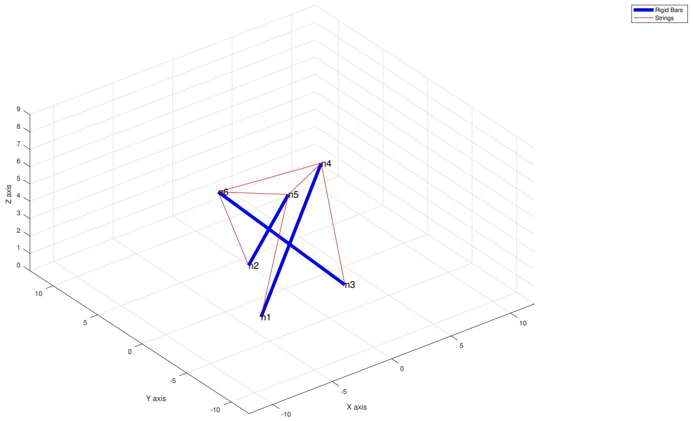

Figure 3.9: 3 bar 6 string prism manipulator final configuration.

In this example, the final target configuration of the nodes n_4, n_5, n_6 are calculated by solving the optimization problem as discussed in the previous section. For a target vector, defined by an azimuth of $90°$ and elevation of $75°$, the target node positions $N_t = [n_4, n_5, n_6]$ as calculated by the optimization is :

$$N_t = \begin{bmatrix} 3.94 & -1.88 & -2.94 \\ 1.18 & -2.90 & 3.55 \\ 5.83 & 6.92 & 5.19 \end{bmatrix}$$

The L and R matrices are :

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

P and D matrices for the constraint are :

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$D = \begin{bmatrix} 3.52 & 0 & 3.52 \\ -2.04 & 4.07 & -2.04 \\ 0 & 0 & 0 \end{bmatrix}$$

The A and B controller coefficient matrices are assumed to be $3I$ and $2I$ respectively. This is chosen such that the controller behaves like a second order overdamped system. We assume no external force is acting on the structure. We run the above discussed Class K tensegrity shape control algorithm to move the prism from its initial configuration at 3.6 to its final configuration at 3.9. To prevent bar lengths from accumulating integration errors, we use the Class K bar length correction algorithm as mentioned in [9]. The results of this simulation is shown in Figures 3.10, 3.11 and 3.12.

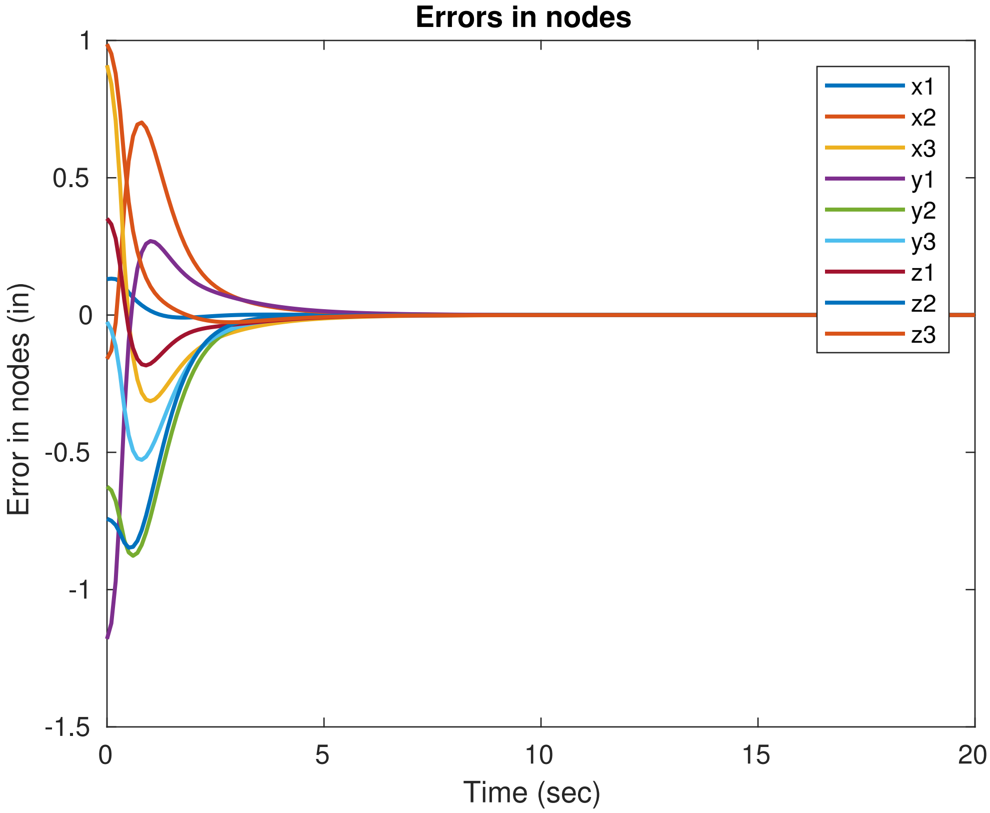

Figure 3.10: Error in the nodes being driven to zero.

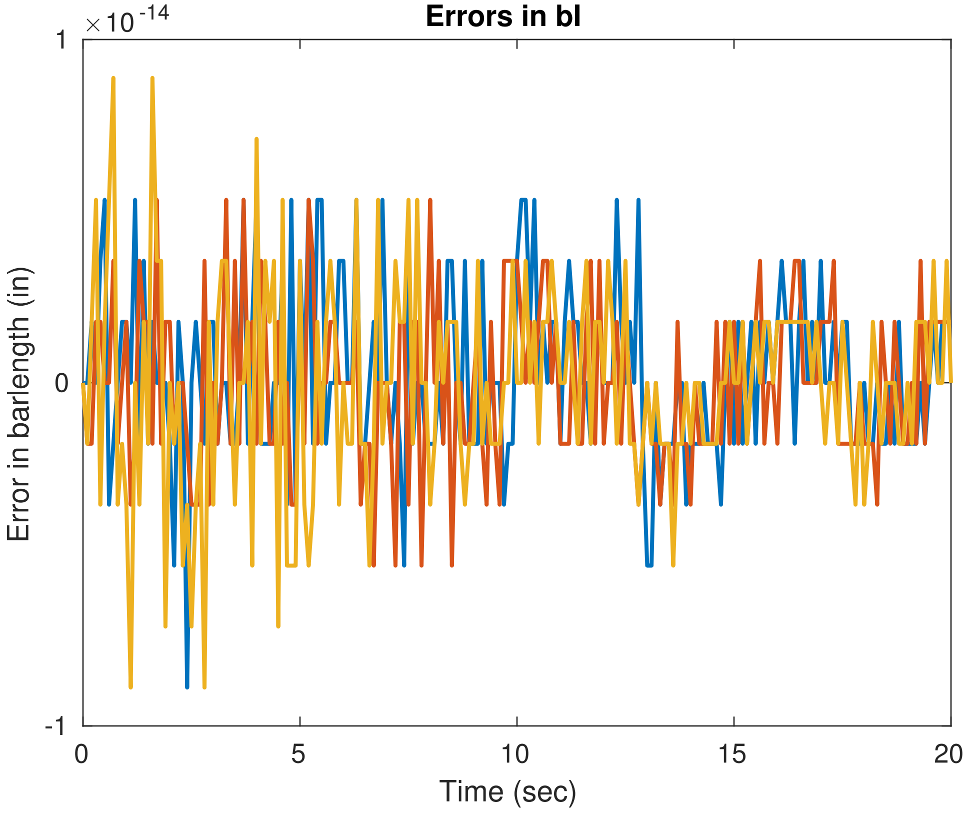

Figure 3.11: Error in the barlengths during the control process.

It is worth mentioning that the above algorithm works best with the barlength correction algorithm running in all 4 iterations of the RK4 integrator, as well as the post integration step. Otherwise, there would be significant bar length errors encountered at each step of integration.

3.6.2 Pointing control of tensegrity prism manipulator

An example of the pointing controller that positions the prism orthogonal to the given vector, without prior knowledge of the target position of the nodes is presented here. The C_b, C_s,P,D,L,R,A,B matrices are similar to the previous example for the shape controller. We begin the controller with same initial configuration of the prism as the previous example, with a target vector defined by an azimuth and elevation of 90 and 75 degrees respectively. The controller successfully points the top plate of the prism orthogonal to the vector. The initial and final configurations are shown in

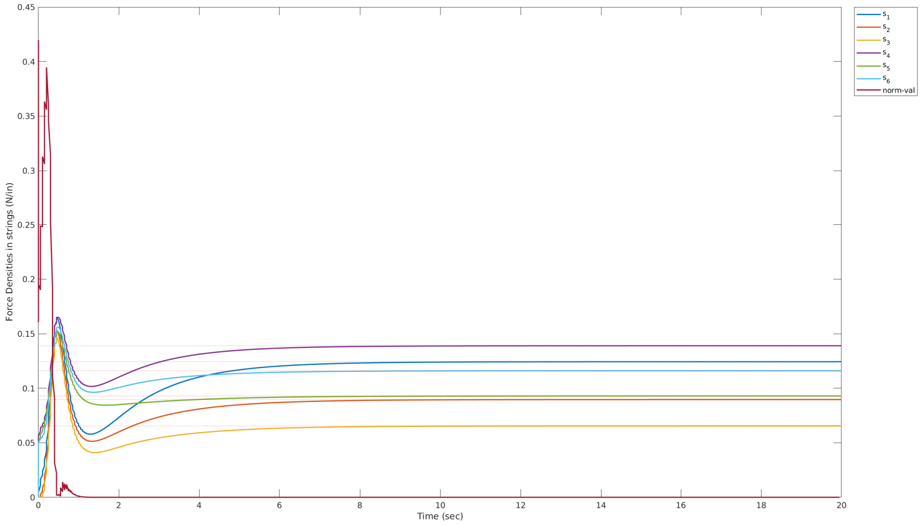

Figure 3.12: Force densities in all 6 strings during the control process.

Figures. 3.13 and 3.14.

3.7 Analysis of the theoretical results

3.7.1 Shape controller example

The Fig.3.12 shows the trajectories of force densities in all the 6 strings of the prism, in solid lines. The dotted lines in the figure are the minimum possible equilibrium force densities of the prism at its final target configuration. We calcuate this by solving a LP problem as follows :

The equilibirum configuration satisfies the following equilibrium equation for tensegrity systems :

$$NK = W$$

where $K = C_s^T \hat{\gamma} C_s - C_b^T \hat{\lambda} C_b$. The above equation can be expressed in $Ax = b$ form by performing the following algebraic manipulation :

Equating the i^{th} column of LHS and RHS,

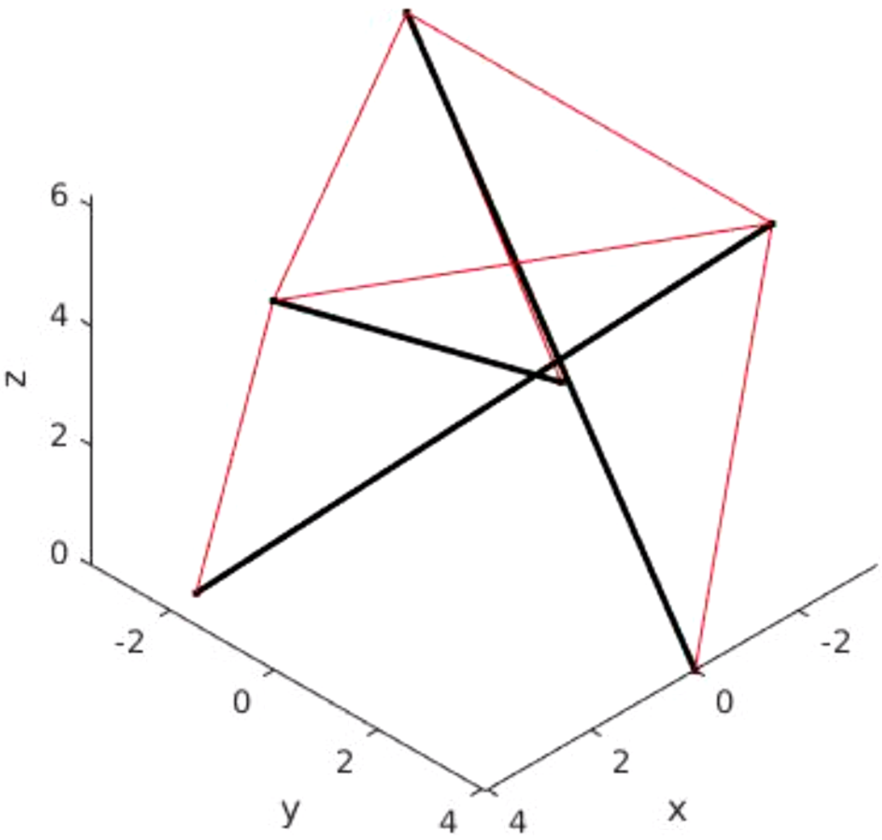

Figure 3.13: Initial configuration of the prism.

$$NKe_i = We_i$$

$$N(C_s^T \hat{\gamma} C_s ei - C_b^T \hat{\lambda} C_b e_i) = We_i$$

$$N(C_s^T \widehat{C_s e_i} \gamma - C_b^T \widehat{C_b e_i} \lambda) = We_i$$

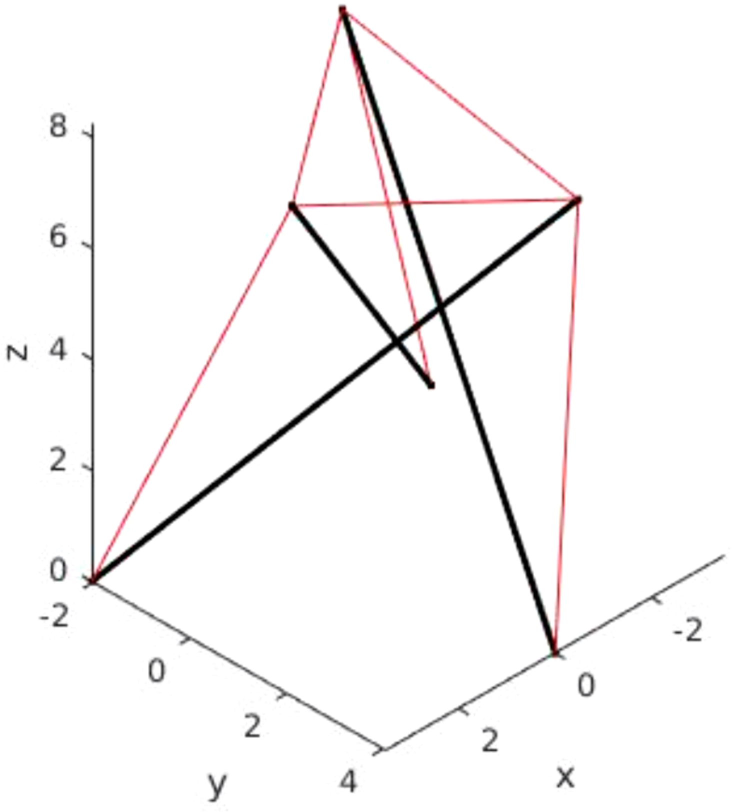

Figure 3.14: Final configuration of the prism.

$$\begin{bmatrix} N(C_s^T\widehat{C_se_1} - C_b^T\widehat{C_be_1}) \\ N(C_s^T\widehat{C_se_2} - C_b^T\widehat{C_be_2}) \\ . \\ . \\ . \\ N(C_s^T\widehat{C_se_n} - C_b^T\widehat{C_be_n}) \end{bmatrix} \begin{bmatrix} \gamma \\ \lambda \end{bmatrix} = \begin{bmatrix} We_1 \\ We_2 \\ . \\ . \\ . \\ We_n \end{bmatrix}$$

Now it is of the form $Ax = b$ where,

$$A = \begin{bmatrix} N(C_s^T\widehat{C_s e_1} - C_b^T\widehat{C_b e_1}) \\ N(C_s^T\widehat{C_s e_2} - C_b^T\widehat{C_b e_2}). \\ . \\ . \\ N(C_s^T\widehat{C_s e_n} - C_b^T\widehat{C_b e_n}) \end{bmatrix}$$

$$b = \begin{bmatrix} We_1 \\ We_2 \\ . \\ . \\ . \\ We_n \end{bmatrix}$$

$$x = \begin{bmatrix} \gamma \\ \lambda \end{bmatrix}$$

The constraint on the force densities is that it is always positive. This is because a string cannot take a compressive force and a bar should not take a tensile force. Hence,

$$\begin{bmatrix} \gamma \\ \lambda \end{bmatrix} \geq 0$$

To find the minimum value of force densities in the strings that satisfy the equilibrium equation, the problem can be expressed as a Linear Programming problem given by,

$$min \quad \Sigma_{i=1}^{i=ns}\gamma_i \qquad ST \quad Ax = b \tag{3.25}$$

Where ns is the number of strings in the structure. It can be observed that the trajectories of force densities in the controller saturates at its equilibrium configuration. This shows that the tensegrity shape control law moves the structure from one equilibrium configuration to another.

In the Fig. 3.12, the norm-val plot shows the value of the least squares norm used to calculate the control. The value of norm-val has been scaled down in magnitude by 1000 times. It can be seen that during the initial moments of the control, the error is significantly high, which means there is not control that exactly satisfies the equation 3.15. But as the time progresses, the value of the norm goes to zero. It is also worth noting that the bar length errors are in the order of 10^{-14} which could be considered negligible compared to the actual length of the rod which is 10 inches.

3.7.2 Pointing controller example

The theoretical results of the pointing controller are shown in Figures. 3.15 and 3.16.

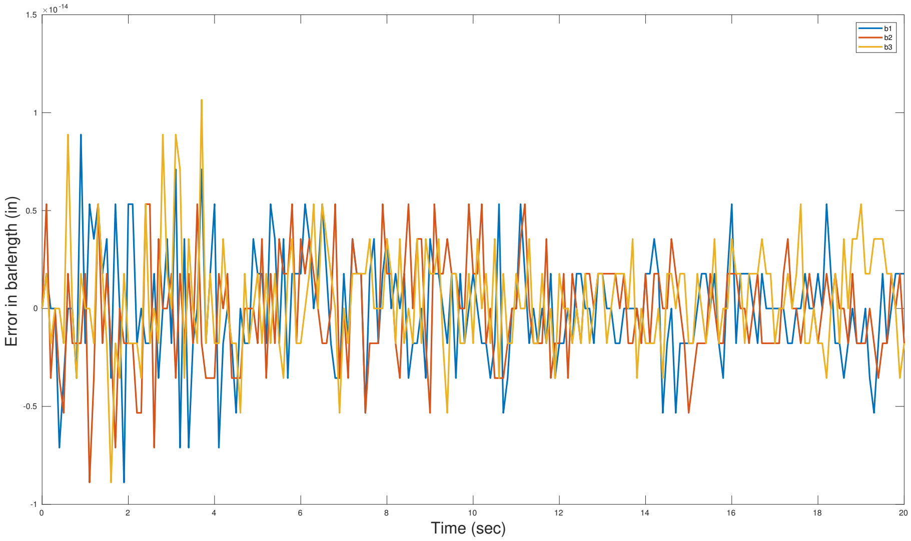

Figure 3.15: Error in the barlengths during the control process.

To avoid significant bar length errors in the integration process, the simulation needs to be run with the aid of barlength correction algorithm running in parallel at every RK iteration and once again after every integration step. With the barlength correction algorithm, the errors in barlength are of the order 10^{-14} inches which is insignificant compared to the actual lengths of the bar (10

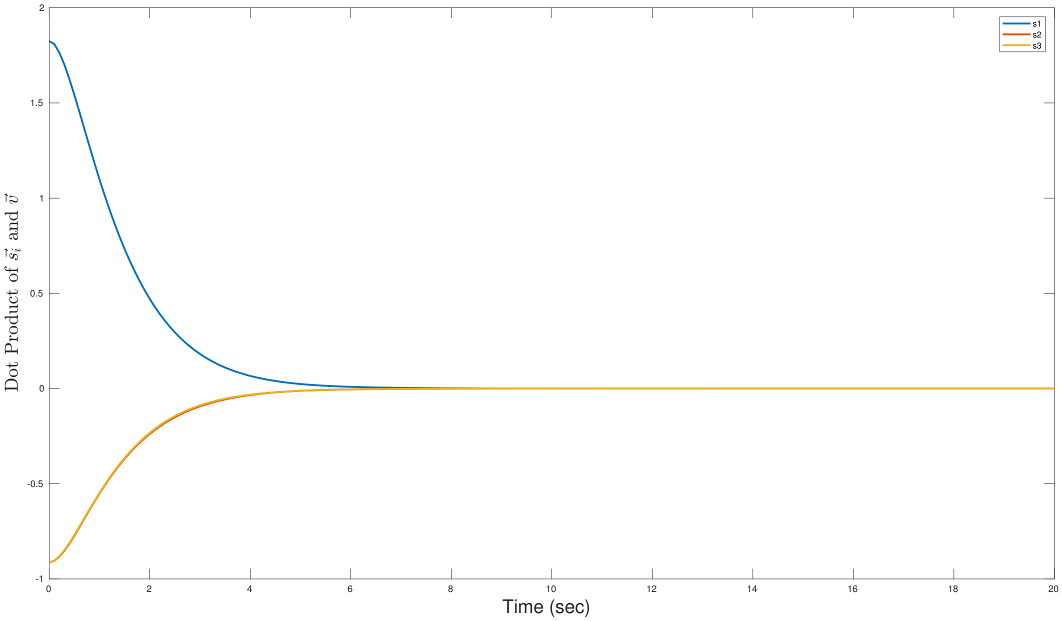

Figure 3.16: Dot product of string vectors with the given target vector.

inches). This type of controller does not take into the consideration the constraints that the z axis of the top 3 nodes must be positive and also the constraint that bars cannot collide or pass through each other. Since these reasons are valid and one typically does not want to encounter such problems in an experimental setting, controller 1 was implemented in the experimental robot.

4. TESTING THE CONTROL ON A LABORATORY PROTOTYPE

The theoretical study shows promising results in control of a 3 bar 6 string tensegrity manipulator to be used as a pointing control manipulator. To experimentally test these results, a laboratory prototype of a 3 bar 6 string tensegrity manipulator was built at the **Tensegrity Systems** Laboratory at Texas A&M University. The final prototype of *InTense*, the tensegrity based manipulator is shown in Fig. 4.1.

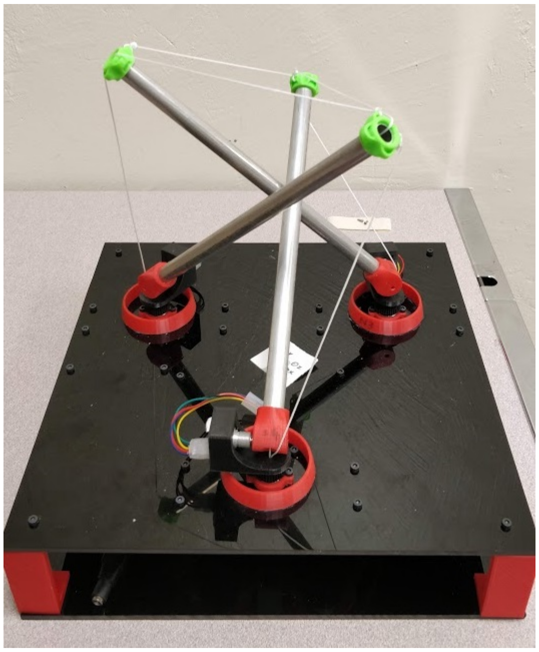

Figure 4.1: The 3 strut Tensegrity prism laboratory prototype.

4.1 Construction of the Robot - From Idea to Prototype

4.1.1 Idea

As a first step to the construction of the robot, a hand-drawn sketch, with the base, nodes, joints, bars and strings was drawn and analyzed. It was inferred from the sketch that the joint at the base needs to be a 2DOF joint to facilitate rotational motion of the bar, except on its long axis. Fig.

4.2 shows the sketch. The red lines are the strings and the orange cylinders are the bars. All the 6 strings are routed through the bars. One end of the string is rigidly attached to the base and the other end is coiled to a pulley on a stepper motor located beneath the base of the motor.

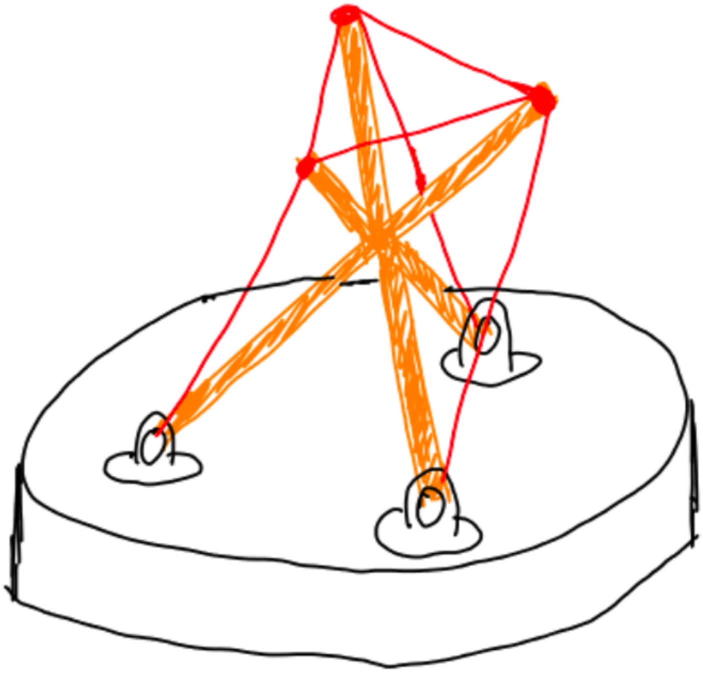

Figure 4.2: The 3 strut Tensegrity prism idea handdrawn cartoon.

Sensor and Actuator selection and placement

The shape control requires information of N and $\dot{N}$, everytime the controller is executed. There can be two possible approaches to sense the position of the nodes :

1. Use the PhaseSpace(PS) motion capture system at LASR lab to sense the position of the nodes N with respect to the PS system's inertial coordinates and perform a coordinate transformation to transform the node position to the inertial coordinates of the robot.

2. Use angular position encoders at the 2DOF joint at the base to sense the elevation and azimuth and then use a nonlinear transformation, given the length of the bar to calculate N.

The PS system requires a human operator to calibrate the system when necessary. It is comparatively bulky and heavy, and, its cost to accuracy ratio is much higher, compared to the angular position encoders. Also, the power consumption of the motion capture system is significantly higher compared to a traditional position encoder. For these reasons, we select the position encoders to sense the azimuth and elevation of each of the 3 bars in the robot. A regular position encoder requires calibration of the zero position every time it is powered, hence we pick a class of position encoders called the 'Absolute position encoders' where the zero position is precalibrated and fixed. **Moser Electronics** manufactures reliable 1024 bit, industrial grade absolute position encoders, shown in Fig. 4.3 that were selected to be used in the robot.

The force densities in the strings were converted to rest lengths, which is then fed to the motor as the control input. So, the motor needs to be precise and provide a reasonable amount of torque to pull the strings and actuate the robot. A 28-BYJ-48 stepper motor, commonly used in 2D printers was chosen as the actuator. These motors provide a high stall torque value, are precise (4076 steps per revolution) and cost much less compared to NEMA stepper motors. The steppers were placed beneath the base of the robot, with pulleys attached to its axle.

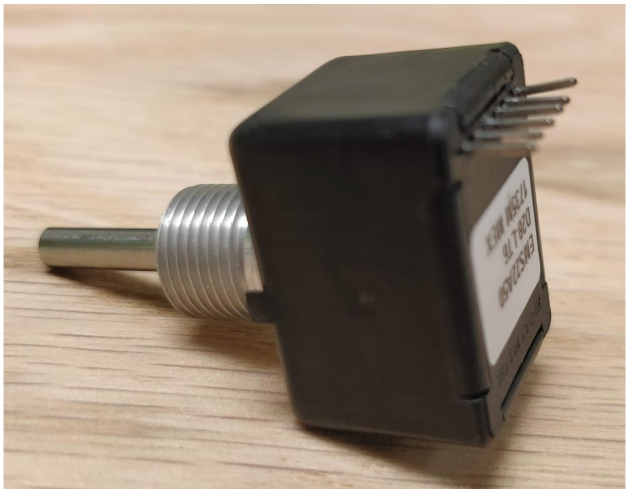

Figure 4.3: The Bourns-Moser absolute position encoder - EMS22A30-C28-LS6.

4.1.2 Design

In the design phase of construction, a SolidWorks CAD model of the robot was designed. In this phase, the hand-sketched idea is extended into a well defined mechanism, providing dimensions and specifications of the materials used for construction. In the CAD model, custom housings were designed to hold the BYJ48 stepper motor [10] and the ULN2003 stepper driver to the base. Fig. 4.4,4.6 and 4.5 shows the SolidWorks CAD model of the robot.

A CAD drawing of the pulley used to change the rest length of the strings is shown in Fig. 4.7. It was 3D printed in PLA material and used in the robot. Custom holders were designed to accomodate for the Arduino microcontroller and the Raspberry Pi Single board computers in the base of the robot.

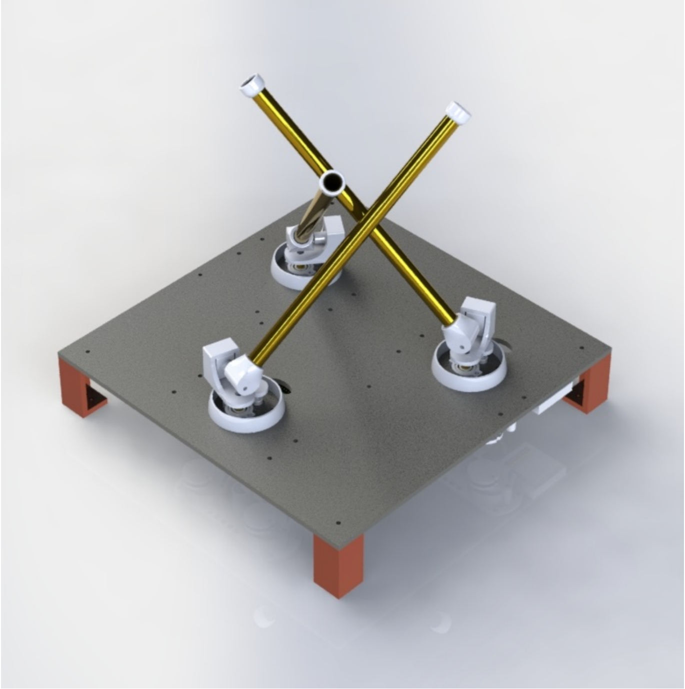

Figure 4.4: A SolidWorks rendering of the prism manipulator - Isometric view.

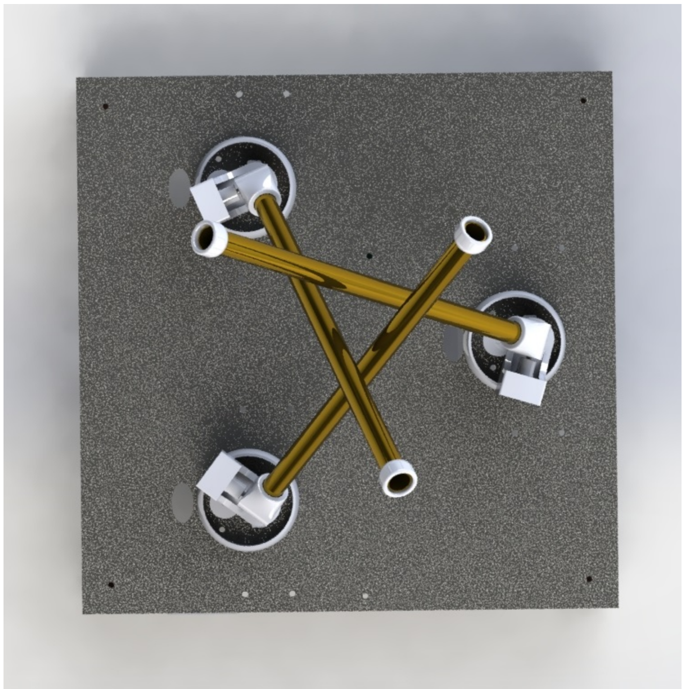

Figure 4.5: A SolidWorks rendering of the prism manipulator - TopView.

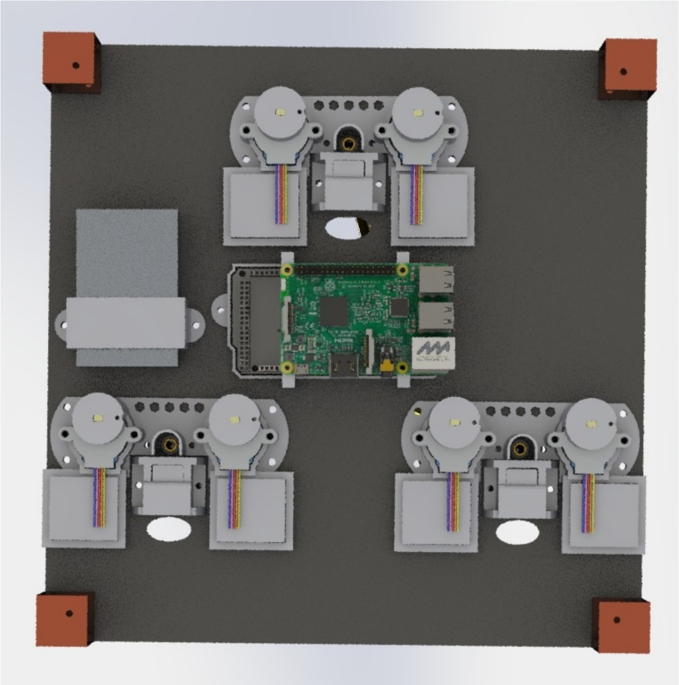

Figure 4.6: A SolidWorks rendering of the prism manipulator - Bottom view.

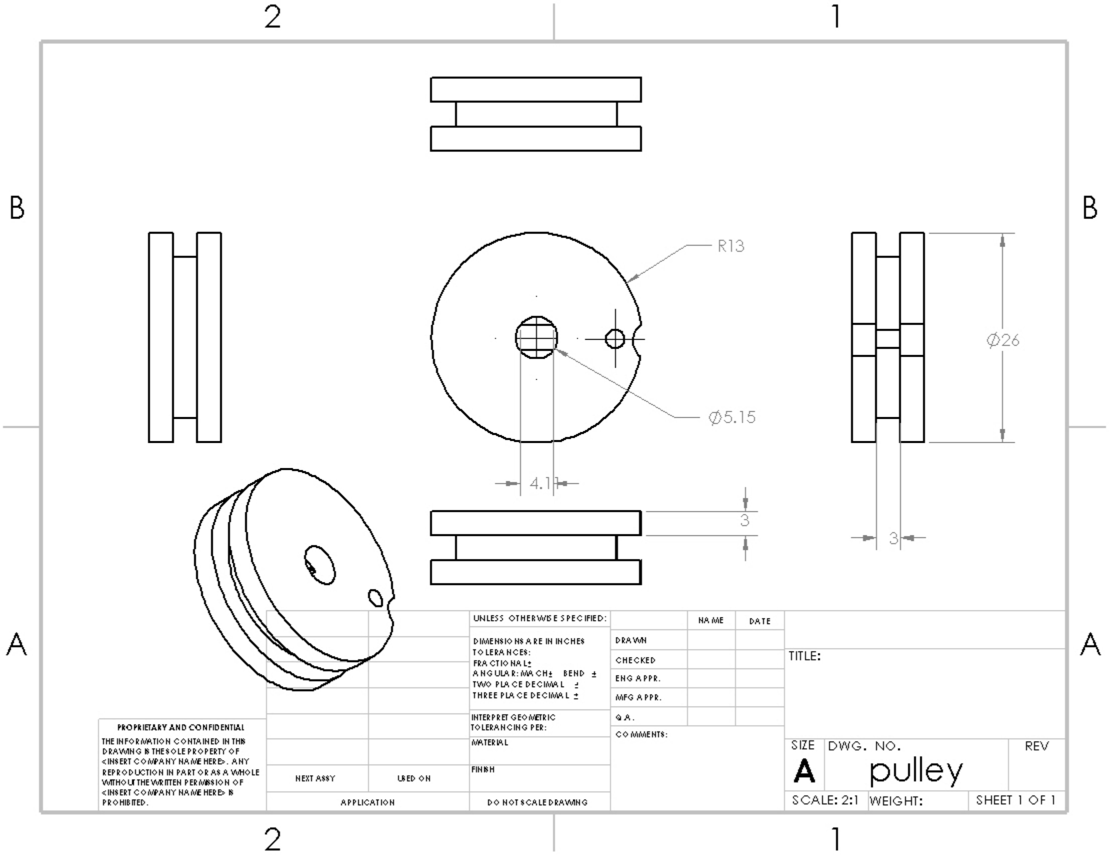

Figure 4.7: A SolidWorks drawing of the pulley used to actuate the strings.

4.1.3 Prototype

The translucent-black base sheet shown in Fig. 4.4 is composed of 0.25 inch thick acrylic material. The holes for wiring and screws were designed in SolidWorks and then precisely carved on the sheet using the acrylic laser cutting equipment at the Aerospace Engineering Building. The hollow brass rods, each 10 inch long, with an outer radius of 0.5 inches were cut using metal rod cutters and its sharp edges were filed to reduce stiction when in contact with the strings. The parts (in white), for housing the stepper motors, motor driver board, absolute position encoders and the brass rod holders were 3D printed using ’red’ ABS plastic.

To facilitate measuring the azimuth angular position of each bar from the sides of the equilateral triangle, as well as to allow for strings to be routed through the base, a parallel axis mechanism composed of spur gears were arranged, as shown in Fig. 4.8. The gears were 3D printed with 'black' ABS material. To prevent backlash errors, a helical gear was initially designed, but the resolution of the 3D printer did not allow for accurate manufacturing of the gears.

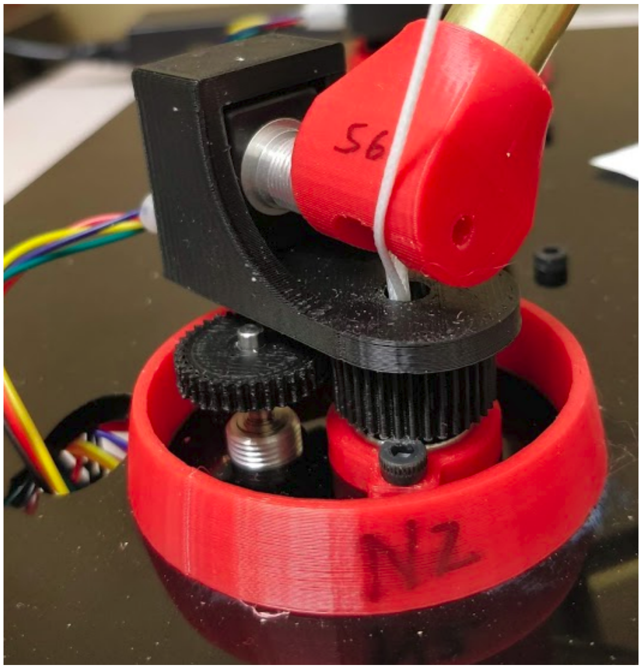

Figure 4.8: A parallel-axis spur gear arrangement for measuring the azimuth.

The pinned node of the brass bar was affixed to the 3D printed cylindrical holder that was friction fit to the axle of the Moser encoder. On the other end of the bar, a 3D printed cap was fit, as shown in Fig. 4.12 to reduce friction between the metal edges of the bars and the strings. It was noticed experimentally that this also helped reduce stiction between the brass bar's edge and strings.

Honeywell's Spectra was used as the strings. The diameter of the string is 0.033 inches. Spectra has a large value of yield strength and provides minimal elongation for the tension provided, compared to other string materials (nylon, cotton, jute) of the same cross section area [11].

4.1.4 Electronics

For the power supply, a 15A 5V regulated power supply was used. A custom general purpose board was soldered to extend the single channel power supply into multiple channels to power the following components with 5V across its terminals :

1. Raspberry Pi Model B
2. Arduino Mega Board
3. 6 ULN 2003 Stepper drivers
4. 6 28-BYJ-48 Stepper motors
5. Status Indicator LEDs
6. 6 Moser Absolute encoders

A common ground, from the power supply was fed across all the electronic components to ensure there is no voltage bias. An image of the prototype's base is shown in Fig. 4.10. The two microcontroller and microprocessor boards used in the experiment are the Arduino Mega 2560 and Raspberry Pi 3 Model B. The Arduino board was used as an aid to the Raspberry Pi for controlling the 6 stepper motors in the robot. The BYJ48 stepper motors were programmed to move in a full-step mode, so each stepper required 4 I/O channels to be commanded. Thus, 6 steppers required 24 I/O channels to be interfaced with the control. Since the Raspberry Pi does not possess enough I/O ports, a cheaper alternative was to use the Arduino Mega 2560 board as a port extension. The Raspberry Pi sends the motor commands to the Arduino Mega using its inbuilt serial interface (USART) at 115200 bps, the Arduino receives the commands and executes the inputs on the steppers. The AccelStepper library [12] was used to program the Arduino for commanding the steppers to move, with the input as inches.

The Raspberry Pi module was installed with the Robot Operating System architecture to run the sensing and actuation of the robot. An external computer that runs Ubuntu Linux was connected to the same network as the Raspberry Pi. The shape control algorithm was run on the external computer while the sensing, estimation and low level control was run on the Raspberry Pi.

A Status indicator LED strip, shown in Fig. 4.9 was custom built and interfaced with the Raspberry Pi. It contains 6 LED indicators that indicate the following alerts :

1. Low Voltage
2. Low Current
3. ShutDown in Progress
4. Controller active
5. Power on
6. Kalman Filter - Linearization update (Blink)

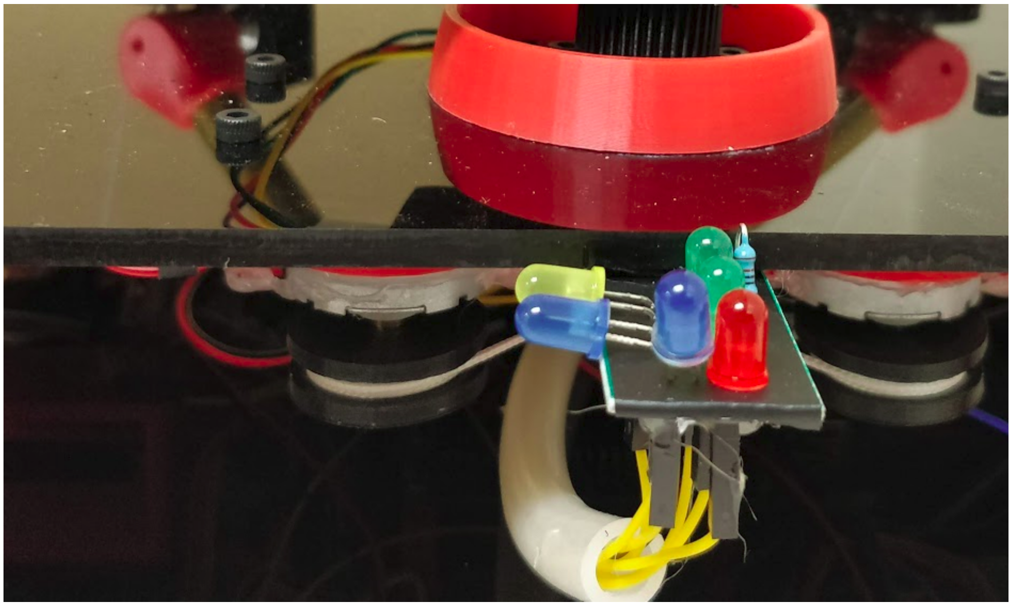

Figure 4.9: LED status indicator strip.

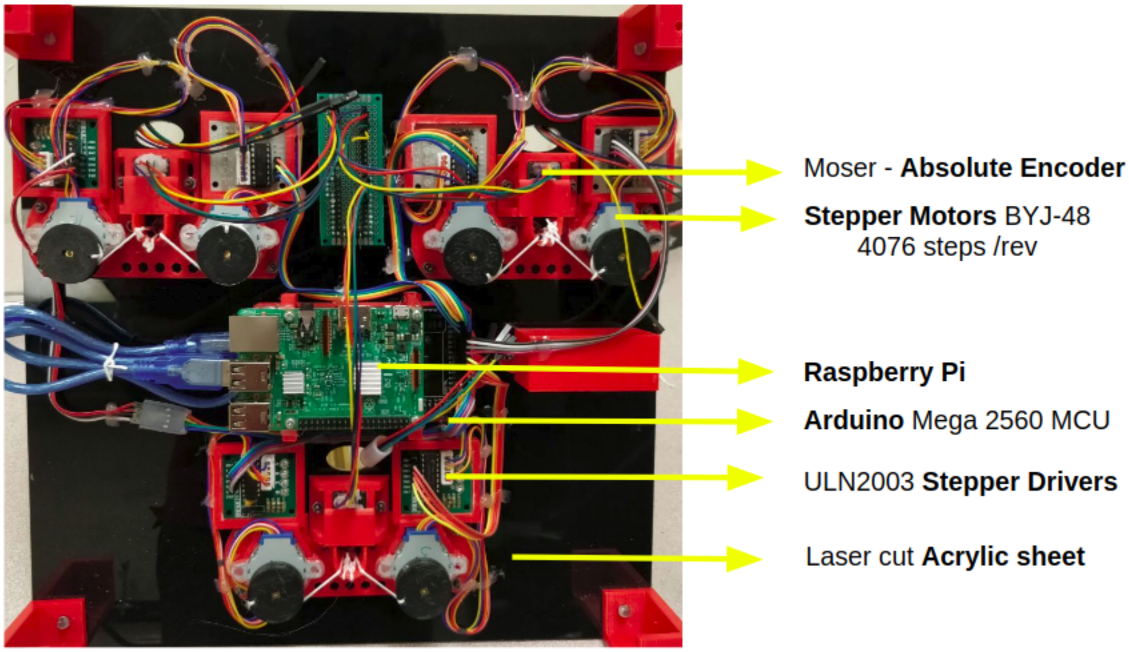
Moser - **Absolute Encoder**
Stepper Motors BYJ-48
4076 steps /rev
Raspberry Pi
Arduino Mega 2560 MCU
ULN2003 **Stepper Drivers**
Laser cut **Acrylic sheet**

Figure 4.10: Components attached at the base of the Tensegrity Robot.

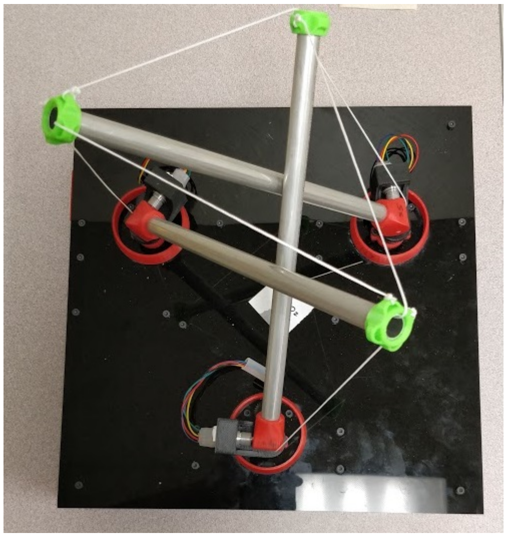

Figure 4.11: Spectra strings routed through the hollow bars.

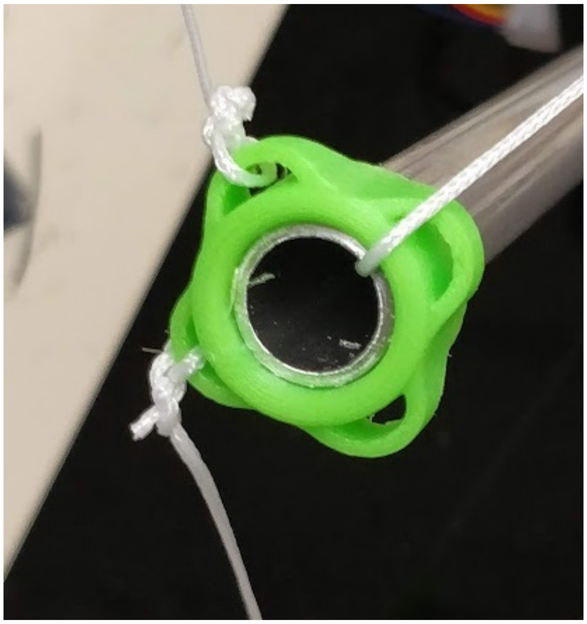

Figure 4.12: Endcap, to reduce stiction between strings and brass bar's ends.

4.2 Using ROS for Tensegrity Robotics Programming

ROS is an open source Robot Operating system that provides a flexible framework for writing robot software. It possesses a collection of tools and libraries that simplifes the task of writing robust software for robotics programming. Its modular framework makes it easy to test fractions of code and then unifying them using its inbuilt reliable communication protocols. A single fractional part of a robot module is called a **node**. It can be a sensor, an actuator or the control or estimation code. A node works on the principle of Publish-Subscribe architecture [13]. Each node can publish data to a topic or subscribe to a topic and receive data. A **roscore** is a collection of nodes and programs that are a pre-requisite of a ROS-based system. roscore coordinates all communication between the nodes. This central framework makes it easy to test and tweak the latency, publish rate, data and data type of each node.

A block diagram of the different nodes are shown in Fig. 4.13. The software architecture was divided into 5 nodes :

1. Tensegrity controller

2. Status Indicators

3. Extended Kalman Filter

4. Encoder position

5. Motor command

The **Tensegrity controller** node executes the tensegrity shape control code. It takes in values of N and $\dot{N}$ of the structure and publishes the control (force densities in the string). The **command_motor** node converts the force densities in the string into rest length of the string using the relation explained in previous chapters, pairs the command with the motor ID (1 to 6) and sends it to the Arduino via USART, which then parses the data and commands the steppers to move. **encoder_positon** node uses a custom synchronous communication protocol defined by Moser[14] to receive the 10 bit position data from the 6 Moser absolute encoders on the robot. The **EKF** node is an implementation of an Extended Kalman Filter to estimate and calculate the value of N and $\dot{N}$ matrices from the encoder positions. **STATUS INDICATORS** node controls the status indicator LEDs based on the robot state.

A rosparam server hosts the parameters of the robot, as documented in Table 4.2.

Mass of each bar (Kg)	0.06803886
Length of each bar (in)	10
Diameter of each bar (in)	0.25
Edge length of bottom triangle (in)	7.05
No. of bars	3
No. of strings	6
Fixed nodes	3
Freq. of sensing and control	200 Hz
Baud rate for Arduino USART	115200
degrees to radians	pi/180
Youngs modulus of Honeywell spectra	3.50E+09
PIN_CLK	11,16
PIN_DAT	5,3,7,8,10,12
PIN_CS	13,18

Table 4.1: Table listing the specifications in rosparam server

4.3 Extended Kalman Filter

The shape control of tensegrity requires the controller to be fed with the N and $\dot{N}$ matrices to calculate the control input γ. The absolute encoders publish the values θ and ϕ, which is the azimuth and elevation of each of the bars, as depicted in the Fig. 4.15. From the measured values of $\theta_1, \theta_2, \theta_3, \phi_1, \phi_2, \phi_3$, we can calculate the N matrix given by the expression :

$$N = \begin{bmatrix} n_1 & n_2 & n_3 & n_4 & n_5 & n_6 \end{bmatrix} \tag{4.1}$$

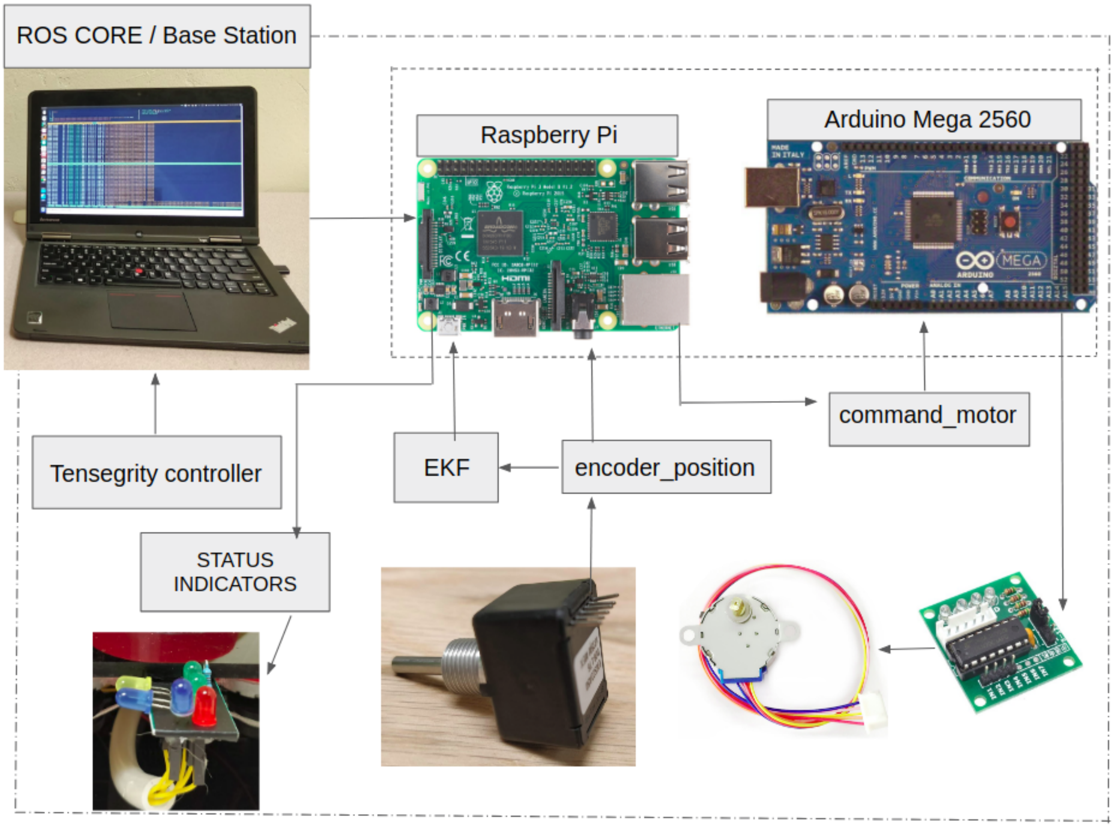

Figure 4.13: Block diagram showing the different modules and connections in the robot.

where n_1, n_2, n_3 form the vertices of an equilateral triangle of side 7.05 inches and,

$$n_4 = \begin{bmatrix} bcos(\phi_1)cos(\theta_1) \\ bcos(\phi_1)sin(\theta_1) \\ bsin(\phi_1) \end{bmatrix} + n_1$$

$$n_5 = \begin{bmatrix} bcos(\phi_2)cos(\theta_2 + 120) \\ bcos(\phi_2)sin(\theta_2 + 120) \\ bsin(\phi_2) \end{bmatrix} + n_2$$

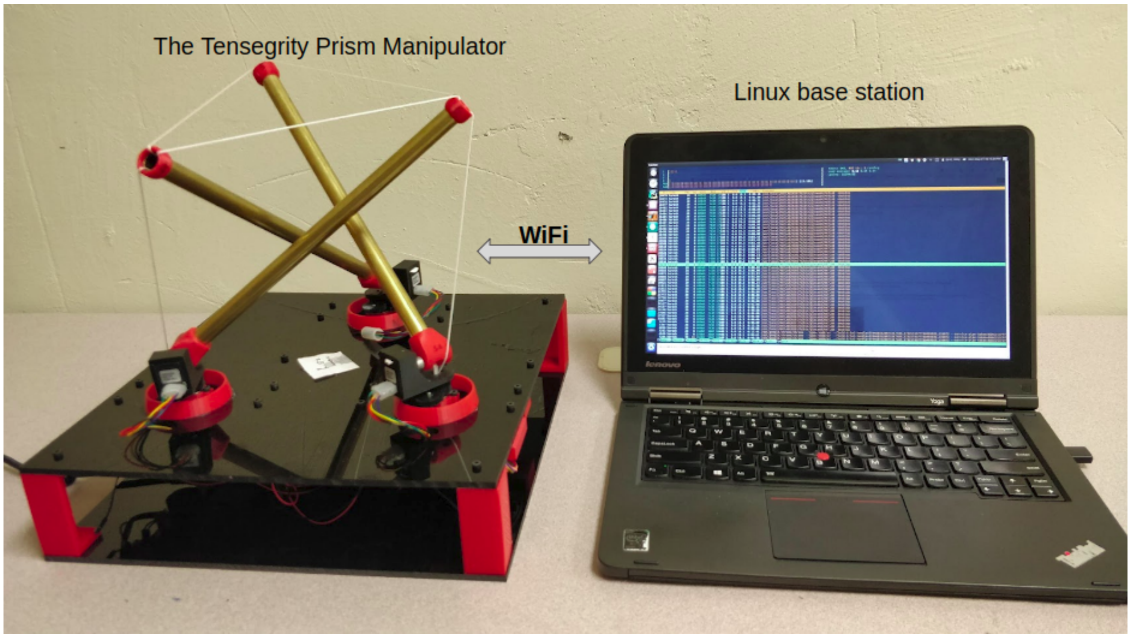

Figure 4.14: The final experimental setup.

$$n_6 = \begin{bmatrix} bcos(\phi_3)cos(\theta_3 + 240) \\ bcos(\phi_3)sin(\theta_3 + 240) \\ bsin(\phi_3) \end{bmatrix} + n_3$$

Similarly, we can calculate the $\dot{N}$ matrix by knowing the values of $\theta_1, \theta_2, \theta_3, \phi_1, \phi_2, \phi_3, \dot{\theta}_1$ and $\dot{\theta}_2, \dot{\theta}_3, \dot{\phi}_1, \dot{\phi}_2, \dot{\phi}_3$ using the following expression :

$$\dot{N} = \begin{bmatrix} \dot{n}_1 & \dot{n}_2 & \dot{n}_3 & \dot{n}_4 & \dot{n}_5 & \dot{n}_6 \end{bmatrix} \tag{4.2}$$

where $\dot{n}_1, \dot{n}_2, \dot{n}_3 = 0$ because the bottom 3 nodes are pinned, and $\dot{n}_4, \dot{n}_5, \dot{n}_6$ can be derived by differentiating the previous expressions with time :

$$\dot{n}_4 = \begin{bmatrix} -bsin(\phi_1)cos(120 + \theta_1)\dot{\phi}_1 - bcos(\phi_1)sin(120 + \theta_1)\dot{\theta}_1 \\ -bsin(\phi_1)sin(120 + \theta_1)\dot{\phi}_1 + bcos(\phi_1)cos(120 + \theta_1)\dot{\theta}_1 \\ bcos(\phi_1)\dot{\phi}_1 \end{bmatrix}$$

$$\dot{n}_5 = \begin{bmatrix} -bsin(\phi_2)cos(\theta_2)\dot{\phi_2} - bcos(\phi_2)sin(\theta_2)\dot{\theta_2} \\ -bsin(\phi_2)sin(\theta_2)\dot{\phi_2} + bcos(\phi_2)cos(\theta_2)\dot{\theta_2} \\ bcos(\phi_2)\dot{\phi_2} \end{bmatrix}$$

$$\dot{n}_6 = \begin{bmatrix} -bsin(\phi_3)cos(240+\theta_3)\dot{\phi_3} - bcos(\phi_3)sin(240+\theta_3)\dot{\theta_3} \\ -bsin(\phi_3)sin(240+\theta_3)\dot{\phi_3} + bcos(\phi_3)cos(240+\theta_3)\dot{\theta_3} \\ bcos(\phi_3)\dot{\phi_3} \end{bmatrix}$$

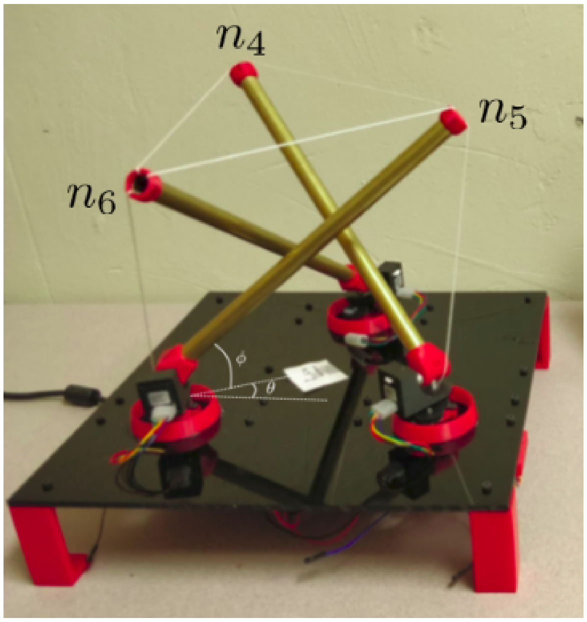

Figure 4.15: Azimuth and Elevation measurement.

The Extended Kalman filter (EKF) is a nonlinear filter which is an extension of the optimal Kalman Filter for linear systems. The EKF linearizes the nonlinear dynamics of the system about an estimate of the current state of the system and the control input. Using EKF, we estimate the values of $\theta_1, \theta_2, \theta_3, \phi_1, \phi_2, \phi_3, \dot{\theta_1}, \dot{\theta_2}, \dot{\theta_3}, \dot{\phi_1}, \dot{\phi_2}, \dot{\phi_3}$ and calculate N and $\dot{N}$ for the control.

4.3.1 Estimating the states

We define the state vector and the input vector as :

$$\chi = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \phi_1 \\ \phi_2 \\ \phi_3 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\phi}_1 \\ \dot{\phi}_2 \\ \dot{\phi}_3 \end{bmatrix}$$

$$U = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \end{bmatrix}$$

The complete Non-linear dynamics of the 3 bar tensegrity manipulator can be derived using any known dynamics approach and be written in the form :

$$\dot{\chi} = g(\chi, U)$$

We can linearize the above equation about the point χ^*, U^* and express it in the form,

$$\dot{x} = Ax + Bu$$

The point χ^* can be $\begin{bmatrix} z \\ \\ 0 \end{bmatrix}$ and U^* can be the static equilibrium tensions in the string at the current χ^* position. We then discretize the linear equation using the following equations :

$$A_k = e^{A\tau}$$

$$B_k = \int_{\tau=0}^{T} e^{A\tau} d\tau$$

where τ is the delta change in time between two iterations. On the robot, the kalman filter runs at 200 Hz frequency, so the value of τ is 0.005. The linearization process is computationally expensive, thus, the linearized model is updated every 3 seconds.

In the Extended Kalman Filter, we can predict the next state using the following prediction equations,

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1}, u_k)$$

$$P_{k|k-1} = A_k P_{k-1|k-1} A_k^T + B_k Q B_k^T$$

P is the covariance of the states. Q is the covariance of uncertainty in the actuators whose value is $0.029761 * I$.

The measurement equation is :

$$z = \begin{bmatrix} I_6 & O_6 \end{bmatrix} x + v \tag{4.3}$$

where v is a zero mean white noise with covariance R whose value, determined from the datasheet of the encoders is $10^{-6} * I$. The innovation/measurement residual equation is :

$$\tilde{y}_k = z_k - h(\hat{x}_{k|k-1})$$

The innovation covariance is :

$$S_k = H_k P_{k|k-1} H_k^T + R_k$$

The Near-optimal Kalman gain is :

$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$

The updated state estimate is :

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k \tilde{y}_k$$

The updated covariance estimate is :

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}$$

4.4 Results of shape control on the experimental setup

The shape control algorithm was run on the built experimental tensegrity model. It can be observed from the graph that the controller performs well in the experimental setup and is consistent with the theoretical simulation. A video collage of the different states of the robot is shown in Fig. 4.16. In this example, a target vector is defined in the inertial frame with an azimuth of $-75°$ and an elevation of $60°$. The graph showing the error in node positions vs time is shown in Fig. 4.17. The colors blue, green and red represent x,y and z axes respectively.

Another example, now with an azimuth of $90°$ and elevation of $75°$ is executed on the prototype. The results are shown in Fig. 4.18 and the graph of error in node positions vs time is shown in Fig. 4.19.

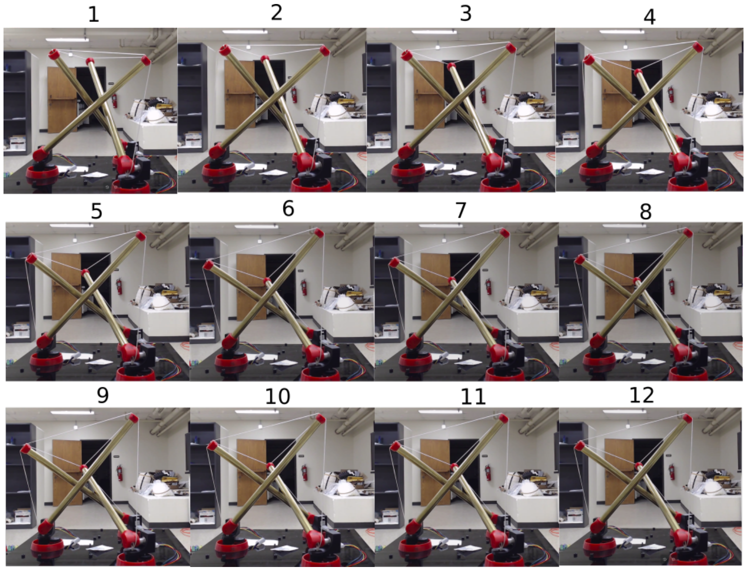

Figure 4.16: A video collage of the control in action. Elevation : 60, Azimuth : -75

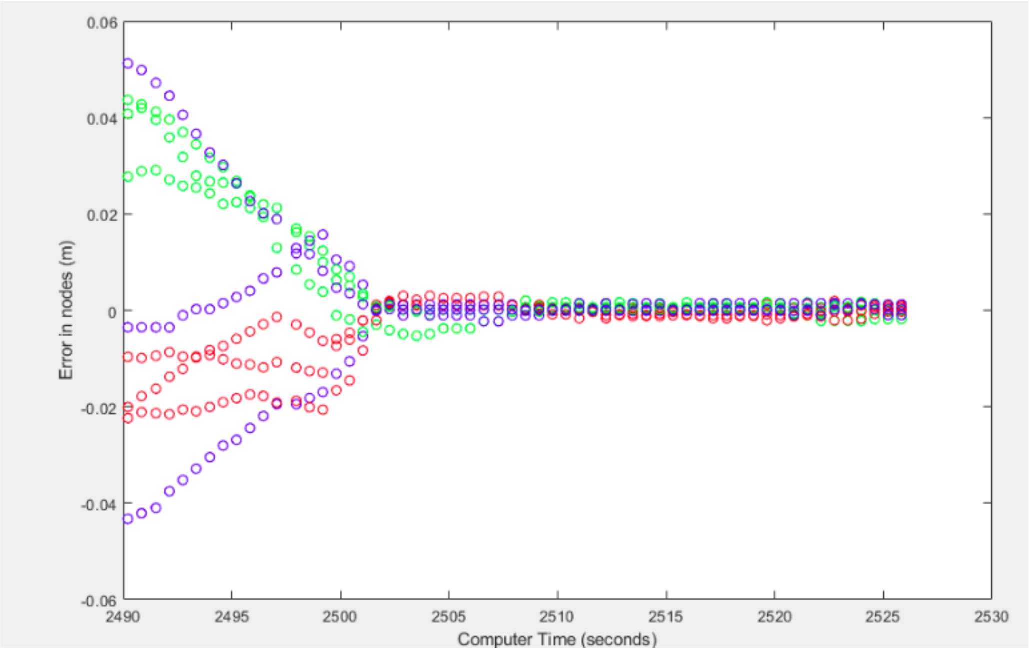

Figure 4.17: Trial 1 : Error in the top 3 node positions (in) vs Time (sec)

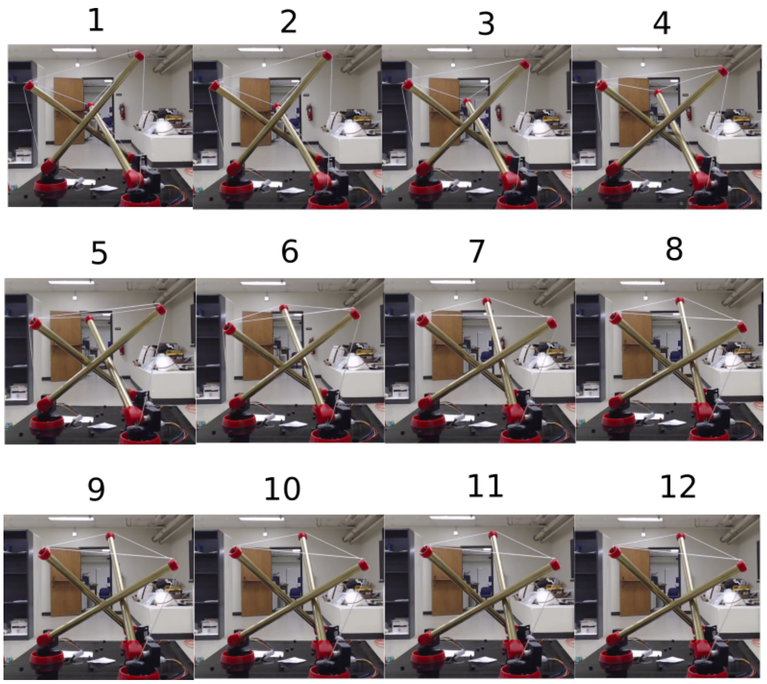

Figure 4.18: A video collage of the control in action. Elevation : 75, Azimuth : 90

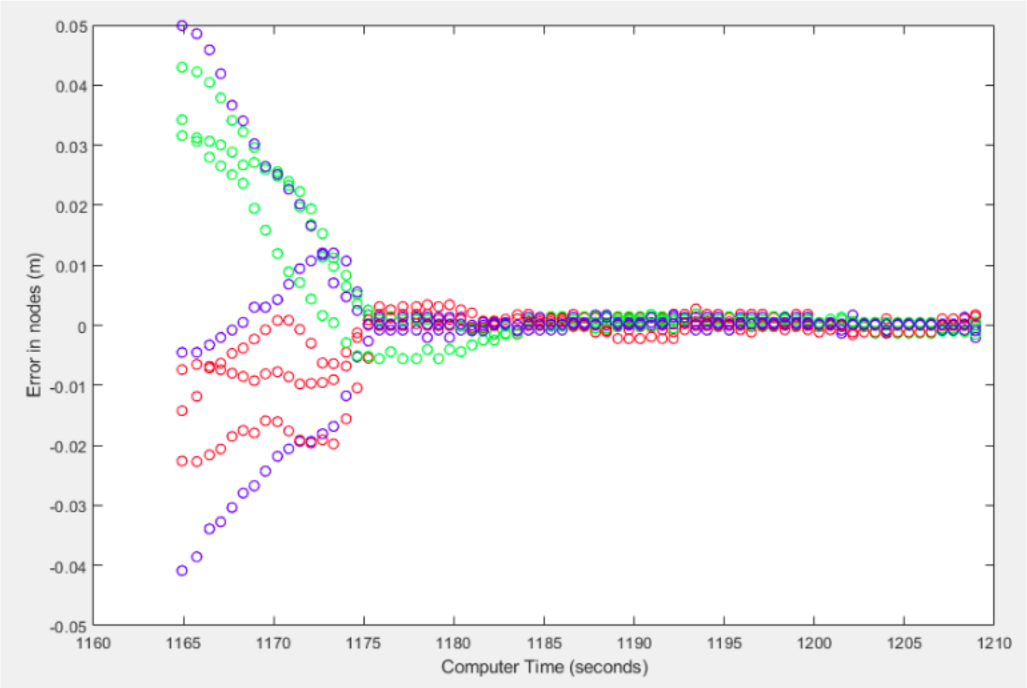

Figure 4.19: Trial 2 : Error in the top 3 node positions (in) vs Time (sec)

5. CONCLUSIONS AND FUTURE WORK

5.1 Conclusion

The previous section validates the shape control on the experimental setup. The graphs for the two examples show that the controller drives the errors in the node positions towards zero. Out of 25 experimental trials, The worst case error in the nodes were 2.077 mm and the lowest value of error was 0.1511 mm. It can be improved significantly by designing precise, metallic, stictionless and frictionless joints using the CNC machine at LASR lab.

It was also noticed in the experiment that the stiction between the bottom nodes and the strings routed through the hollow bars is significantly high. This can be overcome by using thinner strings of diameter lesser than the current string diameter of 0.033 inches.

5.2 Directions for future research

The derived shape control of tensegrity structure uses the complete nonlinear dynamics of the structure, but it may not be the most optimal controller. This approach can be compared with a traditional linear control approach by formulating the non linear dynamics in the form :

$$\dot{x} = A(x,u)x + B(x,u)u$$

where A and B matrices are state and input dependent. We can assume A and B to be independent of the state by substituting the value of the state at that instant, hence expressing the dynamics in a linear system fashion. We can then use the previously well known optimal control techniques on this system to drive the structure from an initial position to its final position, while updating the A and B matrices along the path.

Also, multiple stages of this tensegrity prism manipulator can be built to increase the workspace of the manipulator.

REFERENCES

[1] D. O. M. Skelton Robert E, *Tensegrity Systems*. Springer, 2009.

[2] M. M. Jean-Paul Pinaud and R. E. Skelton, "Path planning for the deployment of tensegrity structures," *Smart Structures and Materials*, 2003.

[3] G. Tibert, "Deployable tensegrity structures for space applications," *Doctoral thesis, Royal Institute of Technology, Department of Mechanics*, 2002.

[4] M. Majji and M. Diz, "Vision based feedback control of a carpal wrist joint," *AIAA Guidance, Navigation, and Control (GNC) Conference, Guidance, Navigation, and Control and Co-located Conferences, (AIAA 2013-4955)*, 2013.

[5] P. S. M.-C. Pan, H. Van Brussel, "Intelligent joint fault diagnosis of industrial robots," *Mechanical Systems and Signal Processing*, 1998.

[6] C. Gosselin and J. Angeles, "Singularity analysis of closed-loop kinematic chains," *IEEE Transactions on Robotics and Automation*, 1990.

[7] W. G. e. Stephen J. Elliott (auth.), H. Ulbrich, *IUTAM Symposium on Vibration Control of Nonlinear Mechanisms and Structures: Proceedings of the IUTAM Symposium held in Munich, Germany, 18âĂŞ22 July 2005*. Solid Mechanics and its Applications 130, Springer Netherlands, 1 ed., 2005.

[8] C. J. . S. Robert, "Nonminimal dynamics of general class k tensegrity systems," *International Journal of Structural Stability and Dynamics*, 2015.

[9] Y. C. Joono Cheong, Robert E. Skelton, "A numerical algorithm for tensegrity dynamics with non-minimal coordinates," *Mechanics Research Communications,*, vol. 58, pp. 46–52, 2014.

[10] "28-byj-48 stepper driver fact sheet." http://robocraft.ru/files/datasheet/28BYJ-48.pdf.

[11] "Honeywell spectra fiber." https://www.honeywell-spectra.com/?document=fiber-capability-guide&download=1.

[12] “Accelstepper library.” Web. http://www.airspayce.com/mikem/arduino/AccelStepper/.

[13] “Robot operating system.” http://wiki.ros.org/.

[14] “Moser absolute encoder datasheet.” https://www.mouser.com/datasheet/2/54/EMS22A-50229.pdf.