Czech Technical University in Prague
Faculty of Mechanical Engineering

Department of Mechanics, Biomechanics and Mechatronics

MASTER'S THESIS

Reconfiguration of Tensegrity Structures

Polohování Tensegritických Struktur

Author:	Bc. Juraj Lieskovský
Supervisor:	Ing. Petr Beneš, Ph.D.
Year:	2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Lieskovský** Jméno: **Juraj** Osobní číslo: **473729**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav mechaniky, biomechaniky a mechatroniky**

Studijní program: **Aplikované vědy ve strojním inženýrství**

Specializace: **Mechatronika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Polohování tensegritických struktur

Název diplomové práce anglicky:

Reconfiguration of tensegrity structures

Pokyny pro vypracování:

1. Seznamte se s problematikou simulace tensegritických struktur a systémů s uzavřenou kinematickou smyčkou obecně.
2. Navrhněte způsob simulace kinematické vazby 'kladka - poddajné lano'.
3. Sestavte simulační model polohovatelného/rekonfigurovatelného tensegritického manipulátoru.
4. Ověřte polohovatelnost sestaveného modelu.

Seznam doporučené literatury:

[1] Balon, A.: Optimalizace a řízení mechatronické tensegrity pro robotiku, diplomová práce, ČVUT v Praze, 2019
[2] Skelton, R. E., de Oliveira, M.: Tensegrity Systems, Springer-Verlag US, 2009
[3] Chandana, P., Valero-Cuevas, F. J., Lipson, H.: Design and control of tensegrity robots for locomotion, IEEE Transactions on Robotics, vol. 22, no. 5, pp. 944-957, 2006
[4] Kaňka, T., Šika, Z.: Modular Tensegrity Manipulator For Replacement Of Spatial Serial Robots, 29th Worshop of Applied Mechanics Book of Papers, pp. 18-21, ČVUT v Praze, 2021

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Petr Beneš, Ph.D. odbor mechaniky a mechatroniky FS

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **22.04.2022** Termín odevzdání diplomové práce: **15.08.2022**

Platnost zadání diplomové práce: ______________

Ing. Petr Beneš, Ph.D. podpis vedoucí(ho) práce	prof. Ing. Michael Valášek, DrSc. podpis vedoucí(ho) ústavu/katedry	doc. Ing. Miroslav Španiel, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací.
Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

.

Datum převzetí zadání Podpis studenta

Annotation sheet

Title: Reconfiguration of Tensegrity Structures

Název: Polohování Tensegritických Struktur

Author: Bc. Juraj Lieskovský

Academic year: 2022

Study program: Applied Sciences in Mechanical Engineering

Department: Department of Mechanics, Biomechanics and Mechatronics

Supervisor: Ing. Petr Beneš, Ph.D.

Scope of work: Pages: 72
Figures: 19
Attachments: 1 (CD)

Keywords: tensegrity, reconfiguration, rigid-body dynamics, multi-body dynamics, manipulator

Klíčová slova: tensegrita, tensegritická struktura, polohování, dynamika tuhých těles, manipulátor

Abstract: This thesis concerns itself with developing an approach to creating more accurate models of reconfigurable tensegrity structures by modeling the interactions between pulleys and axially compliant cables. This is done within the confines of rigid-body dynamics as described by Roy Feathersone which should allow for a straight forward integration into numerous multi-body dynamics libraries. Also a method of solving what would constitute the inverse dynamics problem for these structures is proposed, which can be for example used in conjunction with computed torques controllers.

Abstrakt: Práce se zabývá problematikou vytváření přesnějších modelů polohovatelných tensegritických struktur, pomocí modelování interakce mezi kladkami a poddajnými lany. Toho je dosaženo v rámci dynamiky tuhých těles, jak je popsána Royem Featherstonem, což by mělo dovolovat snadnou implementaci do mnoha knihoven pro simulaci soustav těles. Také je navrhnuta metoda pro řešení problému, který představuje problém inverzní dynamiky těchto struktur. Tu lze například použít ve spojení s přístupen computed torques pro jejich řízení.

I declare that I carried out this master's thesis independently, and only with the cited sources.

In Prague . . . .

Bc. Juraj Lieskovský

I would like to thank my friends and family for their support and Ing. Petr Beneš, Ph.D. for his guidance.

This thesis was written in connection with a basic research project 20-21893S "Mechatronic tensegrities for energy efficient light robots" funded by the Czech Science Foundation. It was also supported by the project SGS19/156/OHK2/3T/12 Mechatronics and adaptronics 2019.

Contents

List of Figures

List of Tables

List of Listings

Introduction

The term tensegrity, coined by Buckminister Fuller, is a conjunction of the two words tension and integrity [1]. Structures described by this term have distinct compressive and tensile parts to which we will refer to as struts and cables. Traditionally struts are connected to each other by cables never coming directly into contact (figure 0.1). In [2] such systems are classified as class 1 tensegrity structures with higher classes being distinguished by the number of struts in contact with each other. For example a tensegrity structure with at most two struts in contact can be referred to as a class 2 tensegrity structure (figure 0.2b). While they were originally intended for use in architecture [1], the lightweight nature of tensegrities has motivated the exploration of their potential for locomotion [3] and manipulation [4].

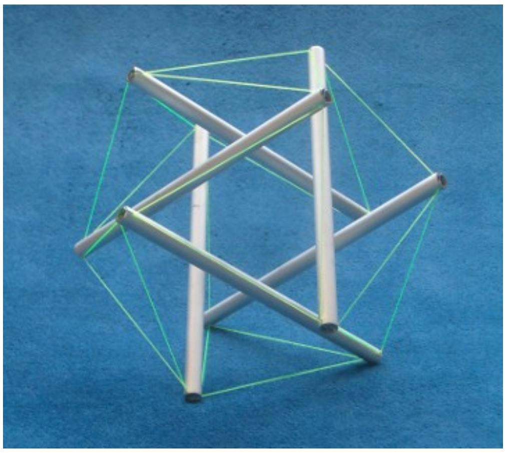
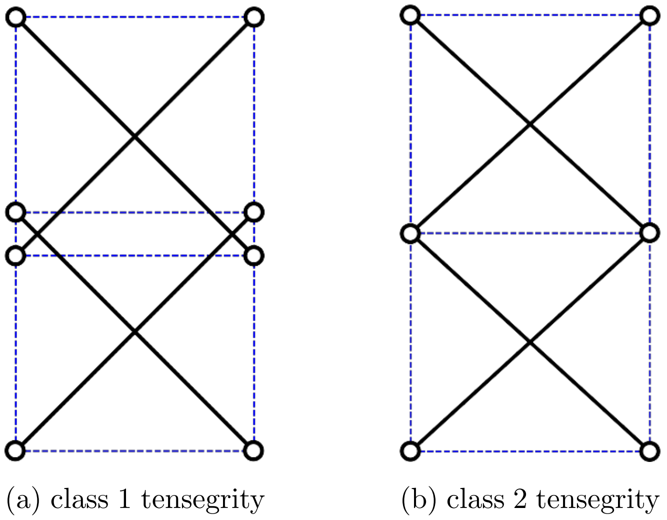


Figure 0.1: tensegrity structure - Jan Marcus
http://www.tensegriteit.nl/e-well-known.html



(a) class 1 tensegrity

(b) class 2 tensegrity

Figure 0.2: tensegrity classes shown on 2D tensegrity structures [5]

Almost universally, models of tensegrity structures are significantly simplified with all cables attached to the end of a strut intersecting in a single point [5], [4], [3], [6]. While this is acceptable for basic forms of analysis in physical realizations multiple cables cannot be anchored to the same exact point (see figure 0.1) resulting in inaccuracies of the model. This problem is accentuated if the structure's shape is modified by winding cables onto pulleys as will be done in this thesis. Each of these pulleys also has to be mounted in a block which rotates around an axis skew to that of the pulley's rotation in order to track the movement of the cable's anchor point (such assembly in can be seen in figure 0.3) introducing additional bodies.

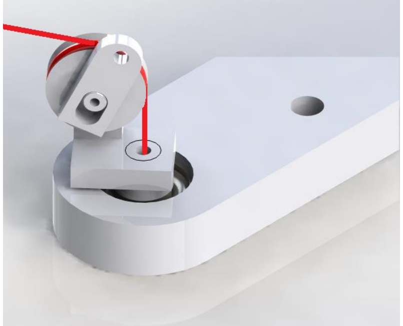


Figure 0.3: pulley assembly of a cable-robot demonstrator [7]

The resulting structure can be viewed as a system of rigid bodies with certain peculiarities. Firstly, kinematically it is a closed loop system. Secondly, for the purposes of modal analysis is it essential that its cables are modelled as compliant. Lastly, if we were to disregard the cables' compliance the system would have more actuators than degrees of freedom. Most libraries for multi-body dynamics (such as simscape multibody [8]) are not equipped to model and control systems with these characteristics. For this reason the open-source library RigidBodyDynamics.jl (RBD.jl) [9] was chosen for the development of necessary approaches as it is also quite well documented and the julia programming language [10], in which it is written, is conductive to fast prototyping. Additionally RBD.jl, similarly to many other libraries (e.g. [11], [12]), references Roy Featherstone's algorithms his most complete work being [13].

Objectives of this thesis can be stated as follows. Gain an understanding of how equations of motion of a closed-loop system can be algorithmically constructed. Use this understanding to develop a method for modelling the interactions between pulleys and compliant cables. Create a model of a tensegrity manipulator including the aforementioned interactions. Develop an approach to acquiring a solution to what would be the inverse dynamics problem if the manipulator's cables were rigid.

We will divide this thesis into five chapters, first of which will review Roy Featherstone's approaches to constructing the equations of motion of rigid body systems detailed in [13] with a focus on systems with kinematic loops. In chapter two we will overview forms of constrained optimization problems and how they can be used for determining the optimal input when controlling rigid body systems. In chapter three we will build upon concepts from the previous chapters, developing approaches to modelling cable-driven manipulators with compliant cables and solving what in their case approximates to the inverse dynamics problem. Chapter four will deviate from the theme of rigid body dynamics by focusing on cubic Hermite splines which will be used later in sections 5.3 and 5.4. Finally in chapter five approaches described in chapter three will be applied to modelling and controlling tensegrity structures.

Chapter 1

Rigid Body Dynamics

1.1 Topology of a Rigid Body System

In its most basic form, the topology of a system of rigid bodies can be represented by a graph where rigid bodies (vertices) are connected by joints (edges), each joint j defining motion constraints between a predecessor $p(j)$ and successor $s(j)$ frame of reference embedded in the two bodies the joint connects.

A frame of reference, commonly represented by an origin point and a right-handed orthogonal unitary basis, is a purely kinematic device that serves as a standard relative to which motion can be measured. For a dynamical account of motion an inertial frame defined in [14] as a "reference-frame with a time-scale, relative to which the motion of a body not subject to forces is always rectilinear and uniform, accelerations are always proportional to and in the direction of applied forces, and applied forces are always met with equal and opposite reactions" must be present in the system. We will define a single inertial frame for the system, referred to as the world frame, which will be embedded in a body referred to as the systems base.

An open-loop system can be described using a kinematic tree starting at the base where each body i is connected to its parent $\lambda(i)$ by joint i. It is also connected to it's children $j \in \mu(i)$ by joints j. For each joint i we may identify its subtree set of bodies $\nu(i)$, that is the set of bodies in the subtree that is supported by joint i. For bodies i we may identify their support set of joints $\kappa(i)$ that are between it and the base of the system.

A closed-loop systems can then be described as an open-loop system with additional loop-joints connecting bodies of otherwise independent branches of the kinematic tree.

1.2 EoM closed-loop system

Using spatial vectors detailed in chapter two of [13] which centers around the use of 6D motion and force vectors we may write equations of motion for a system of N_B rigid bodies conected by N_J joints in the general form

$$\boldsymbol{I}_i \boldsymbol{a}_i + \boldsymbol{v}_i \times^* \boldsymbol{I}_i \boldsymbol{v}_i = {}^i\boldsymbol{X}_0^* \boldsymbol{f}_i^x + e_{ij} {}^i\boldsymbol{X}_{s(j)}^* \boldsymbol{f}_j \qquad \substack{i=1,\dots,N_B \\ j=1,\dots,N_J} \tag{1.2.1}$$

where

$\boldsymbol{I}_i$ spatial inertia of body i

$\boldsymbol{a}_i$ acceleration of body i

$\boldsymbol{v}_i$ velocity of body i

${}^B\boldsymbol{X}_A^*$ transformation of a force vector from frame A to frame B

$\boldsymbol{f}_i^x$ external force (if any) acting on body i

$\boldsymbol{f}_j$ force transmited across joint j

and

$$e_{ij} = \begin{cases} 1 & \text{if } i = s(j) \\ -1 & \text{if } i = p(j) \\ 0 & \text{otherwise} \end{cases} \tag{1.2.2}$$

In equation 1.2.1 $\times^*$ signifies the cross product between a motion and force vector whereas $\times$ would be the same operator between two motion vectors. Similarly ${}^B\boldsymbol{X}_A$ transforms motion vectors as opossed to ${}^B\boldsymbol{X}_A^*$ which transforms force vectors.

Taking into account the structure of the kinematic tree where joint i connects the body i to its parent we may manipulate the equation 1.2.1 for that single body into the form

$$\boldsymbol{I}_i \boldsymbol{a}_i + \boldsymbol{v}_i \times^* \boldsymbol{I}_i \boldsymbol{v}_i = {}^i\boldsymbol{X}_0^* \boldsymbol{f}_i^x + \boldsymbol{f}_i - {}^i\boldsymbol{X}_j^* \boldsymbol{f}_j + e_{il} {}^i\boldsymbol{X}_l^* \boldsymbol{f}_l \qquad \substack{i=1,\dots,N_B \\ j\in\mu(i) \\ l=N_B+1,\dots,N_J} \tag{1.2.3}$$

where $\boldsymbol{f}_i$ is the force transmited from body $\lambda(i)$ to body i across tree-joint i, $\boldsymbol{f}_j$ the force transmited from body i to body $j \in \mu(i)$ across tree-joint j and $\boldsymbol{f}_l$ the force transmited across loop-joint l.

1.2.1 Joint Models

Force $\boldsymbol{f}_J$ transmitted across a joint can be separated into two components active force $\boldsymbol{f}_a = \boldsymbol{T}_a\boldsymbol{\tau}$ resulting from a vector of joint torques $\boldsymbol{\tau} \in \mathbb{R}^{N-N_C}$ of an external (actuators) or internal (springs, dampers, friction) origin and constraint force $\boldsymbol{f}_c = \boldsymbol{T}_c\boldsymbol{\lambda}$ removing the system's degrees of freedom, each DoF removed corresponding to an element of the vector of constraint forces variables $\boldsymbol{\lambda} \in \mathbb{R}^{N_C}$ where N is the number of DoF of an unconstrained rigid body and N_C the number of constraints imposed by a joint. We will refer to matrices $\boldsymbol{T}_a$ and $\boldsymbol{T}_c$ as the active and constraint force subspace respectively, together spanning $\mathbb{R}^N$.

$$\boldsymbol{f}_J = \boldsymbol{T}_a\boldsymbol{\tau} + \boldsymbol{T}_c\boldsymbol{\lambda} \tag{1.2.4}$$

Joudrain's principle of power states: *"The constraint force delivers zero power along every direction of velocity freedom that is compatible with the motion constraints"* [13]. Let $\boldsymbol{v}_J$ be the joint velocity, that is the body's relative velocity to the predecessor of the joint, we may write Jourdain's principle of power in the algebraic form

$$\boldsymbol{f}_c \cdot \boldsymbol{v}_J = 0 \tag{1.2.5}$$

As it must hold for all values of $\boldsymbol{\lambda}$ it is apparent that

$$\boldsymbol{T}_c^T\boldsymbol{v}_J \overset{!}{=} \mathbf{0} \tag{1.2.6}$$

for motion constraints to be satisfied. Equation 1.2.6 constitutes the joint's implicit motion constraints which are enforced by parameters $\boldsymbol{\lambda}$ also referred to as Lagrangian multipliers.

Motion can also be constrained explicitly by expressing $\boldsymbol{v}_J$ as a function of joint parameters $\boldsymbol{q} \in \mathbb{R}^{N-N_C}$ and their derivatives in time.

$$\boldsymbol{v}_J = \boldsymbol{S}\dot{\boldsymbol{q}} \tag{1.2.7}$$

where $\boldsymbol{S}$ is the motion subspace. Using $\boldsymbol{q}$ we may also determine the joint's transformation ${}^s\boldsymbol{X}_p$ and given the second derivative in time its acceleration

$$\boldsymbol{a}_J = \boldsymbol{S}\ddot{\boldsymbol{q}} + \boldsymbol{c}_J\ , \quad \boldsymbol{c}_J = \dot{\boldsymbol{S}}\dot{\boldsymbol{q}} \tag{1.2.8}$$

where $\boldsymbol{c}_J$ is the acceleration bias term.

Having defined the joint velocity using the motion subspace we may substitute it into the algebraic form of the Jourdain's principle of virtual power along with the constraint force defined through the use of the constraint force subspace to attain

$$\boldsymbol{S}^T \boldsymbol{T}_c = \boldsymbol{0} \tag{1.2.9}$$

as it must hold for all values of $\dot{\boldsymbol{q}}$ and $\boldsymbol{\lambda}$. For practical reasons we may choose $\boldsymbol{T}_a$ such that

$$\boldsymbol{S}^T \boldsymbol{T}_a = \boldsymbol{1} \tag{1.2.10}$$

these become apparent if we then premultiply the equation 1.2.4 by $\boldsymbol{S}^T$ resulting in

$$\boldsymbol{S}^T \boldsymbol{f}_J = \boldsymbol{\tau} \tag{1.2.11}$$

Similarly to how joint acceleration $\boldsymbol{a}_J$ was obtained we may differentiate equation 1.2.6 in time resulting in

$$\boldsymbol{T}_c^T \boldsymbol{a}_J = -\dot{\boldsymbol{T}}_c^T \boldsymbol{v}_J \tag{1.2.12}$$

Together $\boldsymbol{T}_a$, $\boldsymbol{T}_c$, $\dot{\boldsymbol{T}}_c$ constitute quantities that must be defined for every joint in order to apply implicit motion constraints whereas $\boldsymbol{S}$, $\boldsymbol{c}_J$ and ${}^s\boldsymbol{X}_p$ are necessary for explicit motion constraints to be applied. As a side note all of the aforementioned quantities are typically expressed in the coordinates of the successor frame of the joint.

1.2.2 Applying Motion Constraints

Using explicit motion constraints velocity of the i-th body $\boldsymbol{v}_i$ can be expressed recursively as

$$\boldsymbol{v}_i = {}^i\boldsymbol{X}_{\lambda(i)} \boldsymbol{v}_{\lambda(i)} + {}^i\boldsymbol{X}_{s(i)} \boldsymbol{v}_{J_i} \ , \quad \boldsymbol{v}_{J_i} = \boldsymbol{S}_i \dot{\boldsymbol{q}}_i \tag{1.2.13}$$

and its acceleration $\boldsymbol{a}_i$ as the sum of an acceleration $\boldsymbol{a}_i^{ap}$ and velocity $\boldsymbol{a}_i^{vp}$ product

$$\boldsymbol{a}_i = \boldsymbol{a}_i^{ap} + \boldsymbol{a}_i^{vp} \tag{1.2.14}$$

both of which can also be expressed recursively

$$\boldsymbol{a}_i^{ap} = {}^i\boldsymbol{X}_{\lambda(i)}\boldsymbol{a}_{\lambda(i)}^{ap} + \boldsymbol{S}_i\ddot{\boldsymbol{q}}_i \qquad (\boldsymbol{a}_0^{ap} = \boldsymbol{0}) \tag{1.2.15}$$

$$\boldsymbol{a}_i^{vp} = {}^i\boldsymbol{X}_{\lambda(i)}\boldsymbol{a}_{\lambda(i)}^{vp} + \boldsymbol{c}_{J_i} + \boldsymbol{v}_i \times \boldsymbol{v}_{J_i} \qquad (\boldsymbol{a}_0^{vp} = -\boldsymbol{a}_g) \tag{1.2.16}$$

In addition the acceleration product can be written as

$$\boldsymbol{a}_i^{ap} = {}^i\boldsymbol{X}_{s(j)}\boldsymbol{S}_j\ddot{\boldsymbol{q}}_j \qquad j \in \kappa(i) \tag{1.2.17}$$

Now we may apply explicit motion constraints to the equations of motion by consequently substituting equations 1.2.14 and 1.2.17 into 1.2.1 the result being

$$\boldsymbol{I}_i{}^i\boldsymbol{X}_{s(j)}\boldsymbol{S}_j\ddot{\boldsymbol{q}}_j + \boldsymbol{I}_i\boldsymbol{a}_i^{vp} + \boldsymbol{v}_i \times^* \boldsymbol{I}_i\boldsymbol{v}_i = {}^i\boldsymbol{X}_0^*\boldsymbol{f}_i^x + e_{il}\,{}^i\boldsymbol{X}_{s(l)}^*\boldsymbol{f}_l \qquad \substack{i=1,\dots,N_B \\ j\in\kappa(i) \\ l=1,\dots,N_J} \tag{1.2.18}$$

where

$$\boldsymbol{f}_l = \boldsymbol{T}_{a_l}\boldsymbol{\tau}_l + \boldsymbol{T}_{c_l}\boldsymbol{\lambda}_l \tag{1.2.19}$$

Superfluous constraint forces transmitted across tree-joints can then be eliminated by pre-multiply 1.2.18 with

$$\boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_i^* \qquad k \in \nu(i) \tag{1.2.20}$$

The result can be written as

$$\boldsymbol{M}_{kj}\ddot{\boldsymbol{q}}_j + \boldsymbol{c}_k = \boldsymbol{W}_{ki}\boldsymbol{f}_i^x + \boldsymbol{\tau}_k + \boldsymbol{L}_{kl}\boldsymbol{\tau}_l + \boldsymbol{K}_{kl}\boldsymbol{\lambda}_l \qquad \substack{i=1,\dots,N_B \\ j,k\in\kappa(i) \\ l=N_B+1,\dots,N_J} \tag{1.2.21}$$

where

$$\boldsymbol{M}_{kj} = \boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_i^*\boldsymbol{I}_i{}^i\boldsymbol{X}_{s(j)}\boldsymbol{S}_j \tag{1.2.22}$$

$$\boldsymbol{c}_k = \boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_i^*(\boldsymbol{I}_i\boldsymbol{a}_i^{vp} + \boldsymbol{v}_i \times^* \boldsymbol{I}_i\boldsymbol{v}_i) \tag{1.2.23}$$

$$\boldsymbol{W}_{ki} = \boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_0^* \tag{1.2.24}$$

$$\boldsymbol{L}_{kl} = e_{kl}\boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_{s(l)}^*\boldsymbol{T}_{a_l} \tag{1.2.25}$$

$$\boldsymbol{K}_{kl} = e_{kl}\boldsymbol{S}_k^T\,{}^{s(k)}\boldsymbol{X}_{s(l)}^*\boldsymbol{T}_{c_l} \tag{1.2.26}$$

The remaining constraint forces $\boldsymbol{\lambda}_l$ must be supplemented with implicit motion constraints which take the form

$$\underbrace{e_{lk}\boldsymbol{T}_{c_l}^{T\,s(l)}\boldsymbol{X}_{s(k)}\boldsymbol{S}_k}_{\boldsymbol{K}_{lk}}\ddot{\boldsymbol{q}}_k = \underbrace{-e_{lk}\dot{\boldsymbol{T}}_{c_l}^{T\,s(l)}\boldsymbol{X}_{s(k)}\boldsymbol{S}_k\dot{\boldsymbol{q}}_k}_{\boldsymbol{k}_k} \qquad \substack{l=N_B+1,\dots,N_J \\ k\in\kappa(i)} \tag{1.2.27}$$

1.2.3 Joint Space Equations of Motion

Dropping the indices we can write equations 1.2.21 and 1.2.27 together as

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{\tau} + \boldsymbol{L}\,\boldsymbol{\tau}^l + \boldsymbol{K}^T\boldsymbol{\lambda} \tag{1.2.28}$$

$$\boldsymbol{K}\ddot{\boldsymbol{q}} = \boldsymbol{k} \tag{1.2.29}$$

where

$\boldsymbol{M}$	mass matrix of the system's kinematic tree
$\ddot{\boldsymbol{q}}$	vector of tree-joint accelerations
$\boldsymbol{c}$	vector of bias terms
$\boldsymbol{W}$	wrench jacobian matrix
$\boldsymbol{f}$	vector of external forces
$\boldsymbol{\tau}$	vector of tree-joint torques
$\boldsymbol{L}$	torque jacobian matrix
$\boldsymbol{\tau}^l$	vector of loop-joint torques
$\boldsymbol{K}$	constraint jacobian matrix
$\boldsymbol{\lambda}$	vector of constraint forces variables
$\boldsymbol{k}$	vector of constraint bias terms

which can also be written as a system of linear equations

$$\begin{bmatrix} \boldsymbol{M} & -\boldsymbol{K}^T \\ \boldsymbol{K} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} \ddot{\boldsymbol{q}} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \boldsymbol{W}\boldsymbol{f} + \boldsymbol{\tau} + \boldsymbol{L}\,\boldsymbol{\tau}^l \\ \boldsymbol{k} \end{bmatrix} \tag{1.2.30}$$

who's solution constitutes the solution to the forward dynamics problem.

1.2.4 Transformation into Independent Coordinates

By transforming the EoM of a closed-loop system into independent coordinates we may obtain a canonically identical form to that of system without kinematic loops

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{\tau} \tag{1.2.31}$$

In order to do so we must eliminate implicit motion constraints. This can be done by calculating what we will refer to as the independency jacobian matrix $\boldsymbol{G}$ and independency bias $\boldsymbol{g}$ which satisfy

$$\ddot{\boldsymbol{q}} = \boldsymbol{G}\ddot{\boldsymbol{y}} + \boldsymbol{g} \tag{1.2.32}$$

$$\boldsymbol{K}\boldsymbol{g} = \boldsymbol{k} \tag{1.2.33}$$

$$\boldsymbol{K}\boldsymbol{G} = \boldsymbol{0} \tag{1.2.34}$$

then we may pre multiply equation 1.2.28 by $\boldsymbol{G}^T$ and apply the substitution in equation 1.2.32 producing.

$$\widetilde{\boldsymbol{M}}\ddot{\boldsymbol{y}} + \widetilde{\boldsymbol{c}} = \widetilde{\boldsymbol{W}}\boldsymbol{f} + \widetilde{\boldsymbol{\tau}} \tag{1.2.35}$$

where

$$\widetilde{\boldsymbol{M}} = \boldsymbol{G}^T\boldsymbol{M}\boldsymbol{G}\ , \quad \widetilde{\boldsymbol{c}} = \boldsymbol{G}^T(\boldsymbol{M}\boldsymbol{g} + \boldsymbol{c})\ , \quad \widetilde{\boldsymbol{W}} = \boldsymbol{G}^T\boldsymbol{W}\ , \quad \widetilde{\boldsymbol{\tau}} = \boldsymbol{G}^T(\boldsymbol{\tau} + \boldsymbol{L}\boldsymbol{\tau}^l) \tag{1.2.36}$$

If we apply both 1.2.32 and 1.2.33 to equation 1.2.29 terms on both sides cancel each other out, therefore equation 1.2.35 completely describes the dynamics of closed-loop system.

Let n be the number of joint parameters, n_c the number of constraints and n_i the number of independent joint parameters. This transformation not only reduces the number of equations from $n + n_c$ to $n_i = n - n_c$ at the cost of additional operations but also allows us to apply algorithms derived further in the thesis for both open and closed loop systems. Notably for open loop systems $n = n_i$.

1.2.5 Actuation

We will differentiate between passive joint torques $\boldsymbol{p}$ and torques resulting a vector of inputs $\boldsymbol{u} \in \mathbb{R}^{n_a}$ being applied using the manipulator matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n_a}$

$$\boldsymbol{\tau} = \boldsymbol{p} + \boldsymbol{B}\boldsymbol{u} \tag{1.2.37}$$

where n_a is the number of actuated joint parameters. Similarly we may decompose the vector of loop-joint torques allowing us to write

$$\widetilde{\boldsymbol{\tau}} = \widetilde{\boldsymbol{p}} + \widetilde{\boldsymbol{B}}\boldsymbol{u} \tag{1.2.38}$$

where

$$\widetilde{\boldsymbol{p}} = \boldsymbol{G}^T(\boldsymbol{p} + \boldsymbol{L}\boldsymbol{p}^l)\ , \quad \widetilde{\boldsymbol{B}} = \boldsymbol{G}^T(\boldsymbol{B} + \boldsymbol{L}\boldsymbol{B}^l)\ , \quad \widetilde{\boldsymbol{B}} \in \mathbb{R}^{n_i \times n_a} \tag{1.2.39}$$

Substituting equation 1.2.38 into 1.2.35 we attain

$$\widetilde{\boldsymbol{M}}\ddot{\boldsymbol{y}} + \widetilde{\boldsymbol{c}} = \widetilde{\boldsymbol{W}}\boldsymbol{f} + \widetilde{\boldsymbol{p}} + \widetilde{\boldsymbol{B}}\boldsymbol{u} \tag{1.2.40}$$

which has the same canonical form as the equations of motion resulting from substituting 1.2.37 into 1.2.31

$$\boldsymbol{M}\ddot{\boldsymbol{q}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{p} + \boldsymbol{B}\boldsymbol{u} \tag{1.2.41}$$

where both $\ddot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{y}}$ are generalized acceleration coordinates.

For better visual clarity in consequent chapters of this thesis we will drop tildes from matrices in equation 1.2.40 giving us the form

$$\boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{p} + \boldsymbol{B}\boldsymbol{u} \tag{1.2.42}$$

where

$\boldsymbol{M}$	mass matrix
$\ddot{\boldsymbol{y}}$	vector of independent accelerations
$\boldsymbol{c}$	vector of bias terms
$\boldsymbol{W}$	wrench jacobian matrix
$\boldsymbol{f}$	vector of external forces
$\boldsymbol{p}$	passive joint torques
$\boldsymbol{B}$	manipulator matrix
$\boldsymbol{u}$	vector of inputs

Chapter 2

Input Optimization

When solving the inverse dynamics problem, assuming $\boldsymbol{B} \in \mathbb{R}^{n_i \times n_a}$ is full rank we may distinguish three general cases

1. $\text{rank}(\boldsymbol{B}) = n_a < n_i$ the system is under actuated : it has more motion freedoms than the actuators can control.
2. $\text{rank}(\boldsymbol{B}) = n_i = n_a$. In this case, the system is properly actuated : there is a unique solution to the inverse dynamics problem.
3. $\text{rank}(\boldsymbol{B}) = n_i < n_a$. In this case, the system is redundantly actuated : infinitely many different values of $\boldsymbol{u}$ will produce the same acceleration.

In the case of redundantly actuated systems we are presented with the opportunity of optimizing the values of $\boldsymbol{u}$. These optimization problems typically fall into the category of constrained optimization problems (COPs).

2.1 Constrained Optimization Problems

A constrained optimization problem can be written as

$$\text{minimize:} \quad f(\boldsymbol{x}) \tag{2.1.1}$$

$$\text{subject to:} \quad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{e}\ , \quad \boldsymbol{h}(\boldsymbol{x}) \geq \boldsymbol{f} \tag{2.1.2}$$

where $\boldsymbol{x}$ is the vector of optimized parameters. COPs can be generally separated into problems of

- Linear programing (LP) where $f(\boldsymbol{x})$, $\boldsymbol{g}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ are linear functions
- Non-linear programing (NLP) where $f(\boldsymbol{x})$, $\boldsymbol{g}(\boldsymbol{x})$ and $\boldsymbol{h}(\boldsymbol{x})$ are non-linear functions

[15]

Quadratic Programming Problems

A subset of NLP problems are quadratic programing (QP) problems in which the objective function is quadratic and the constraints are formulated as non-strict linear equalities

$$\text{minimize:} \quad \frac{1}{2}\boldsymbol{x}^T\boldsymbol{P}\boldsymbol{x} + \boldsymbol{q}^T\boldsymbol{x} \tag{2.1.3}$$

$$\text{subject to:} \quad \boldsymbol{a} \leq \boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b} \tag{2.1.4}$$

[16]. Specialized algorithms have been developed for solving such problems significantly outperforming generalized NLP algorithms if applicable.

2.2 Inverse dynamics of an over-actuated system

Let us suppose we are attempting to solve the inverse dynamics problem of an over-actuated system who's dynamics are described by equations of motion in the form

$$\boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{p} + \boldsymbol{B}\boldsymbol{u} \tag{1.2.42}$$

Input optimization generally desirable for such systems is that of minimizing the input vector's norm $\|\boldsymbol{u}\|_2$ which is also subject to the constraint $\forall i : |u_i| \leq u_{MAX}$ while also having to satisfying the aforementioned equations of motion.

2.2.1 NLP formulation

In this case the COP can be formulated as being solved for $\boldsymbol{x} \equiv \boldsymbol{u}$, having an objective function

$$\boldsymbol{f}(\boldsymbol{u}) = \boldsymbol{u}^T\boldsymbol{u} \tag{2.2.1}$$

equality constraints

$$\boldsymbol{g}(\boldsymbol{u}) \equiv \boldsymbol{B}\boldsymbol{u} \;, \quad \boldsymbol{e} \equiv \boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p} \tag{2.2.2}$$

and inequality constraints

$$\boldsymbol{h}(\boldsymbol{u}) \equiv \begin{bmatrix} \boldsymbol{u} \\ -\boldsymbol{u} \end{bmatrix} \;, \quad \boldsymbol{f} \equiv \begin{bmatrix} \boldsymbol{u}_{MAX} \\ -\boldsymbol{u}_{MAX} \end{bmatrix} \tag{2.2.3}$$

2.2.2 Direct QP formulation

It is important to note that the objective function $f(\boldsymbol{u})$ is quadratic while the constraints are linear meaning the problem can be reformulated as a QP problem. We may do so directly optimizing for $\boldsymbol{x} \equiv \boldsymbol{u}$ with objective function's terms

$$\boldsymbol{P} \equiv \boldsymbol{1} \;, \quad \boldsymbol{q} = \boldsymbol{0} \tag{2.2.4}$$

and constraints' terms

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p} \\ \boldsymbol{u}_{MIN} \end{bmatrix} \;, \quad \boldsymbol{A} = \begin{bmatrix} \boldsymbol{B} \\ \boldsymbol{1} \end{bmatrix} \;, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p} \\ \boldsymbol{u}_{MAX} \end{bmatrix} \tag{2.2.5}$$

2.2.3 Nullspace QP formulation

Alternatively we may decrease the number of elements in the vector of optimized parameters by first calculating a solution that satisfies the equations of motion $\boldsymbol{u}_0$ and the null matrix of $\boldsymbol{B}$

$$\forall \boldsymbol{z} \in \mathbb{R}^{n_a - n_i} : \ \boldsymbol{B}\boldsymbol{N}\boldsymbol{z} = \boldsymbol{0} \quad \Leftrightarrow \quad \boldsymbol{N} = \mathrm{Null}(\boldsymbol{B}) \tag{2.2.6}$$

This allows us to search for a final solution to the inverse dynamics problem in the form $\boldsymbol{u} = \boldsymbol{u}_0 + \boldsymbol{N}\boldsymbol{z}$ which satisfies the equations of motion for all $\boldsymbol{z} \in \mathbb{R}^{n_a - n_i}$, meaning we are free to optimize $\boldsymbol{x} \equiv \boldsymbol{z}$ only for the constraint

$$\boldsymbol{u}_{MIN} \leq \boldsymbol{u}_0 + \boldsymbol{N}\boldsymbol{z} \leq \boldsymbol{u}_{MAX} \tag{2.2.7}$$

the objective function being expressed as a function of $\boldsymbol{z}$

$$f(\boldsymbol{z}) = \boldsymbol{u}_0^T \boldsymbol{u}_0 + 2\, \boldsymbol{u}_0^T \boldsymbol{N}\boldsymbol{z} + \boldsymbol{z}^T \boldsymbol{N}^T \boldsymbol{N}\boldsymbol{z} \tag{2.2.8}$$

where $\boldsymbol{u}_0$ can be any solution to the inverse dynamics problem that will serve as the starting point of the optimization. An intuitive choice for $\boldsymbol{u}_0$ is the the least square solution to the equations of motion

$$\boldsymbol{u}_0 = \boldsymbol{B}^{+}(\boldsymbol{M}\ddot{\boldsymbol{y}} + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p}) \tag{2.2.9}$$

which means that if $\boldsymbol{u}_{MIN} \leq \boldsymbol{u}_0 \leq \boldsymbol{u}_{MAX}$ it is by itself the problem's solution.

The terms of the objective function will then take the form

$$\boldsymbol{P} \equiv \boldsymbol{N}^T \boldsymbol{N}\ , \quad \boldsymbol{q} \equiv 2\, \boldsymbol{N}^T \boldsymbol{u}_0 \tag{2.2.10}$$

and constraints' terms

$$\boldsymbol{a} \equiv \boldsymbol{u}_{MIN} - \boldsymbol{u}_0\ , \quad \boldsymbol{A} \equiv \boldsymbol{N}\ , \quad \boldsymbol{b} \equiv \boldsymbol{u}_{MAX} - \boldsymbol{u}_0 \tag{2.2.11}$$

Chapter 3

Dynamics of Tensegrity Structures

In order to simulate and control tensegrity structures we will develop a suitable method for modelling the cable-pulley interaction as well as cable elasticity within the confines of rigid-body dynamics. We will do so by deriving an *involute joint* which will mimic the behaviour of an axially rigid cable (does not elongate) wrapping around a pulley. We will also describe several variants of a *compliant involute joint*, with a suitably chosen and applied internal torque model, accounting for elasticity, damping and general uni-directionality of the cable.

In our model the successors of the *involute joints* will be *cable-end* bodies. We will have the option of connecting them to other bodies using various joints (not only a spherical joint) with properties of their own allowing for greater freedom when describing how the cable is attached. Although it is not required in RBD.jl the *cable-end* itself can poses inertial properties, which can be for example prescribed as those of a length of cable which we assume to be always unwound. If the accuracy of such a model would still be unsatisfactory, there is a possibility of creating bodies with state dependent mass and spacial inertia, although this approach would not fall under rigid-body dynamics.

Consequently we will describe an approach to approximating the inverse dynamics of a system with compliant cables partially overcoming the challenges associated with controlling an under actuated system.

In section 3.1 different variants of the compliant involute joint as well as the approximation of the inverse dynamics problem will be tested on a simple class 2 tensegrity consisting of a strut connected to the base of the system using a universal joint with three cables attached to its free end.

3.1 Involute Joint

The kinematics of an involute joint closely resemble those of a circle's involute. Origin of the frame preceding the joint P coincides with the circle's centre while the succeeding frame's origin S is located at the end of the string (figure 3.1). Given the evolute's radius r the relative position and orientation of the succeeding frame can be described by the angle of unwrapping ϕ and the taught length of the string l.

If the string is modelled as axially rigid $l = r\phi$ but for an axially compliant cable $l = r\phi(1+\varepsilon)$ meaning the unwound length of the string is described as having a free length $l_0 = r\phi$ and strain (proportional deformation) ε.

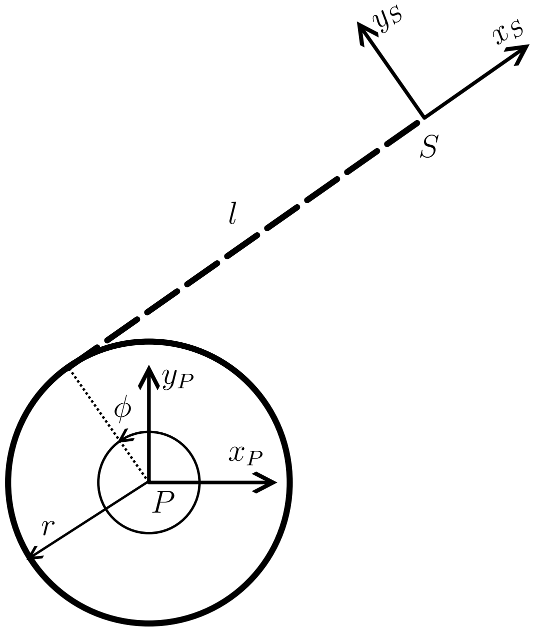


Figure 3.1: involute Joint

3.1.1 Kinematics

Let us then define the vector $\boldsymbol{p}$ as $\overrightarrow{PS}$, expressed in F_s coordinates

$$^{s}\boldsymbol{p} = \hat{\boldsymbol{\jmath}} r + \hat{\boldsymbol{\imath}} l \tag{3.1.1}$$

and $\boldsymbol{E}$ as the rotation matrix that transforms 3D vectors from F_p to F_s coordinates

$$\boldsymbol{E} = \boldsymbol{R}_z^{-1}(\phi)\ , \quad \boldsymbol{R}_z(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1.2}$$

where $\hat{\boldsymbol{\imath}}$, $\hat{\boldsymbol{\jmath}}$ and $\hat{\boldsymbol{k}}$ are unit vectors

$$\hat{\boldsymbol{\imath}} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}\ , \quad \hat{\boldsymbol{\jmath}} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\ , \quad \hat{\boldsymbol{k}} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{3.1.3}$$

The transformation $^{s}\boldsymbol{X}_p$ between F_p and F_s can then be described as consisting of a rotation followed by a translation characterized by $\boldsymbol{E}$ and $^{s}\boldsymbol{p}$ respectively

$$^{s}\boldsymbol{X}_p = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ -{}^{s}\boldsymbol{p}\times & \mathbf{1} \end{bmatrix} \begin{bmatrix} \boldsymbol{E} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{E} \end{bmatrix} = \begin{bmatrix} \boldsymbol{E} & \mathbf{0} \\ -{}^{s}\boldsymbol{p} \times \boldsymbol{E} & \boldsymbol{E} \end{bmatrix} \tag{3.1.4}$$

(see section 2.8 of [13]). Such transformation corresponds with a joint velocity $\boldsymbol{v}_J$ in the form

$$\boldsymbol{v}_J = \begin{bmatrix} ^{s}\boldsymbol{\omega} \\ ^{s}\boldsymbol{v} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\omega} \times {}^{s}\boldsymbol{p} + {}^{s}\dot{\boldsymbol{p}} \end{bmatrix} \tag{3.1.5}$$

Allowing for the more general case where $l = r\phi(1+\varepsilon)$ we will define both the angle of unwrapping and unwound length as functions of joint parameters $\phi = \phi(\boldsymbol{q})$, $l = l(\boldsymbol{q})$ where $\boldsymbol{q} = [q_1]$ for the rigid variant and $\boldsymbol{q} = [q_1\, q_2]^T$ for the remaining compliant variants. Such definition allows us to derive the the motion subspace $\boldsymbol{S}$ of each variant using the same formula

$$\boldsymbol{S} = \frac{\partial \boldsymbol{v}_J}{\partial \dot{\boldsymbol{q}}} \tag{3.1.6}$$

To completely describe each variant we will also provide the acceleration bias term $\boldsymbol{c}_J(\boldsymbol{q}, \dot{\boldsymbol{q}})$ which appear as a term in the joint acceleration

$$\boldsymbol{a}_J = \boldsymbol{S}\ddot{\boldsymbol{q}} + \boldsymbol{c}_J \ , \quad \boldsymbol{c}_J = \dot{\boldsymbol{S}}\dot{\boldsymbol{q}} \tag{3.1.7}$$

3.1.2 Internal Dynamics

In the case of one dimensional tension the total strain energy accumulated within a deformed body can be expressed as an integral functional of strain energy density Λ over its volume before deformation V_0

$$U = \int_{V_0} \Lambda(\xi, u(\xi))d\xi \ , \quad \Lambda = \frac{1}{2}\varepsilon \cdot \sigma \tag{3.1.8}$$

where ξ is the body coordinate, u displacement, ε strain and σ tension. We may determine the internal deformation force F by varying U

$$\delta U = \frac{\partial U}{\partial u}\delta u = F\delta u \tag{3.1.9}$$

after substituting

$$\sigma = E\varepsilon \ , \quad \varepsilon = \frac{u}{l_0} \tag{3.1.10}$$

into equation 3.1.8. The undeformed volume of the unwound cable being

$$V_0 = Al_0 \ ; \quad A = \pi\rho^2 \ , \ l_0 = r\phi \tag{3.1.11}$$

where E is Young's module of elasticity and ρ the cable's radius the result of 3.1.9 yields

$$\delta U = AE\varepsilon\delta u \tag{3.1.12}$$

from which we may determine the internal force F to be

$$F = AE\varepsilon \tag{3.1.13}$$

As we have chosen to work in the realm of linear deformation, it is impossible for us to model buckling and thus the cable's behaviour under compression must be significantly simplified. As we will always attempt to maintain the cable under tension it is acceptable for us to work with a model where the cable offers no resistance to compression which can be implemented by premultiplying F with a unit step function

$$H(\varepsilon) = \begin{cases} 1, & \varepsilon > 0 \\ 0, & \varepsilon \leq 0 \end{cases} \tag{3.1.14}$$

As internal dynamics of the joint must relate to parameters $\boldsymbol{q}$ we will define a vector of elasticity torques $\boldsymbol{e}$ as

$$\boldsymbol{e} \cdot \delta \boldsymbol{q} = H(\varepsilon) F \delta u \tag{3.1.15}$$

After applying the parametrization of $\varepsilon = \varepsilon(\boldsymbol{q})$ and $u = u(\boldsymbol{q})$ specific to each variant of the joint we may simply express the $\boldsymbol{e}$ from the right side of the equation 3.1.15 as

$$\boldsymbol{e} = H(\varepsilon) F \frac{\partial u}{\partial \boldsymbol{q}} \ , \quad F = AE\varepsilon \tag{3.1.16}$$

due to the fact that $\delta u = \frac{\partial u}{\partial \boldsymbol{q}} \delta \boldsymbol{q}$.

We may similarly derive a vector of damping torques $\boldsymbol{d}$ by substituting tension in 3.1.8 for $\sigma = \mu \dot{\varepsilon}$ where μ is the damping coefficient. Performing manipulations identical to those when deriving the vector of elasticity torques we then attain

$$\boldsymbol{d} = H(\varepsilon) F \frac{\partial u}{\partial \boldsymbol{q}} \ , \quad F = A\mu\dot{\varepsilon} \tag{3.1.17}$$

3.1.3 Individual Variants of the Involute Joint

We will derive four variants of the joint in total, first of which will be the rigid (non-compliant) variant requiring only a single joint parameter and no internal torques. The remaining three compliant variants will differ in their parametrizations of l, u, ε which will determine the form of internal torques $\boldsymbol{e}$ and $\boldsymbol{d}$. This will also effect how the joint model can be implemented and which control strategies are applicable to the resulting system.

We will distinguish between the compliant variants by which quantity is represented by the the joint's second parameter, e.g. "ε variant" will have $q_2 = \varepsilon$.

rigid variant

In it's most basic non-compliant form the vector of joint parameters has a single element $\boldsymbol{q} = [q_1]$ with no strain or internal torque being inherently calculated. Therefore the variant is completely described by its motion subspace and acceleration bias term.

- motion subspace

$$\boldsymbol{S} = \begin{bmatrix} \hat{\boldsymbol{k}} \\ \hat{\boldsymbol{\jmath}} r q \end{bmatrix} \tag{3.1.18}$$

- acceleration bias term

$$\boldsymbol{c}_J = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\jmath}} r \dot{q}^2 \end{bmatrix} \tag{3.1.19}$$

***u* variant**

In the first of the compliant variants q_2 is equivalent to u. Here we will use the alternate definition of the unwound cable's deformed length $l = l_0 + u$ to derive the necessary quantities.

- substitutions

$$u = q_2,\ \varepsilon = \frac{q_2}{rq_1},\ l = rq_1 + q_2 \tag{3.1.20}$$

- motion subspace

$$\boldsymbol{S} = \begin{bmatrix} \hat{\boldsymbol{k}} & \mathbf{0} \\ \hat{\boldsymbol{\jmath}}(rq_1 + q_2) & \hat{\boldsymbol{\imath}} \end{bmatrix} \tag{3.1.21}$$

- acceleration bias term

$$\boldsymbol{c}_J = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\jmath}}\dot{q}_1(r\dot{q}_1 + \dot{q}_2) \end{bmatrix} \tag{3.1.22}$$

- vector of elasticity torques

$$\boldsymbol{e} = H(q_2)AE\frac{q_2}{rq_1}\begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.1.23}$$

- vector of damping torques

$$\boldsymbol{d} = H(q_2)A\mu\frac{q_1\dot{q}_2 - \dot{q}_1 q_2}{rq_1^2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.1.24}$$

ε variant

In this variant q_2 is chosen as equivalent to the ε. This makes it suitable for a form of a system's reduction where ε is approximated as constant.

- substitutions

$$u = rq_1q_2,\ \varepsilon = q_2,\ l = rq_1(1 + q_2) \tag{3.1.25}$$

- motion subspace

$$\boldsymbol{S} = \begin{bmatrix} \hat{\boldsymbol{k}} & \mathbf{0} \\ \hat{\boldsymbol{\jmath}}rq_1(1 + q_2) + \hat{\boldsymbol{\imath}}rq_2 & \hat{\boldsymbol{\imath}}rq_1 \end{bmatrix} \tag{3.1.26}$$

- acceleration bias term

$$\boldsymbol{c}_J = \begin{bmatrix} \mathbf{0} \\ \hat{\boldsymbol{\jmath}}r(1 + q_2)\dot{q}_1^2 + \hat{\boldsymbol{\jmath}}rq_1\dot{q}_1\dot{q}_2 + 2\hat{\boldsymbol{\imath}}r\dot{q}_1\dot{q}_2 \end{bmatrix} \tag{3.1.27}$$

- vector of elasticity torques

$$\boldsymbol{e} = H(q_2)AEq_2 \begin{bmatrix} rq_2 \\ rq_1 \end{bmatrix} \tag{3.1.28}$$

- vector of damping torques

$$\boldsymbol{d} = H(q_2)A\mu\dot{q}_2 \begin{bmatrix} rq_2 \\ rq_1 \end{bmatrix} \tag{3.1.29}$$

l variant

Another approach is to express the angle and length using two independent variables $\phi = q_1$, $l = q_2$. Then the joints transformation can be characterized as a rotation around the z-axis by q_1 succeeded by a rigid translation in the direction of the y-axis by r followed by a translation along the x-axis by q_2. This allows us to create a compliant involute joint from a revolute and prismatic joint with externally calculated torques in software such as simscape multibody (see section 5.1.4).

- substitution

$$u = q_2 - rq_1,\ \varepsilon = \frac{q_2 - rq_1}{rq_1},\ l = q_2 \tag{3.1.30}$$

- motion subspace

$$\boldsymbol{S} = \begin{bmatrix} \hat{\boldsymbol{k}} & \boldsymbol{0} \\ \hat{\boldsymbol{\jmath}}q_2 - \hat{\boldsymbol{\imath}}r & \hat{\boldsymbol{\imath}} \end{bmatrix} \tag{3.1.31}$$

- acceleration bias term

$$\boldsymbol{c}_J = \begin{bmatrix} \boldsymbol{0} \\ \hat{\boldsymbol{\jmath}}\dot{q}_1\dot{q}_2 \end{bmatrix} \tag{3.1.32}$$

- vector of elasticity torques

$$\boldsymbol{e} = H(q_2 - rq_1)AE\frac{q_2 - rq_1}{rq_1} \begin{bmatrix} -r \\ 1 \end{bmatrix} \tag{3.1.33}$$

- vector of damping torques

$$\boldsymbol{d} = H(q_2 - rq_1)A\mu\frac{q_1\dot{q}_2 - \dot{q}_1 q_2}{rq_1^2} \begin{bmatrix} -r \\ 1 \end{bmatrix} \tag{3.1.34}$$

3.2 Approximation of Inverse Dynamics

Including cable compliance transforms what would be an over-actuated system if the cables were rigid to an under-actuated system. While the system remains controllable if kept under tension the end-effector's acceleration is not directly dependent on the input torques which means the inverse dynamics problem is unsolvable.

Thankfully it is generally desirable for the cables to be as stiff as possible allowing us to approximate the cables as rigid at every discrete step of input torques' calculation, invalidating the equations of motion. We may restore their equality by introducing fictitious internal joint torques which correspond to the expected deformation. We will use this expected deformation to maintain a certain degree of pre-tension within the cables and therefore controllability of the entire system.

For this method of control to work properly, we will have to use the ε variant of the compliant involute for which we may write the vector of elasticity torques as

$$\boldsymbol{e} = \boldsymbol{A}\boldsymbol{q} \tag{3.2.1}$$

where $\boldsymbol{A}$ is the accumulation Jacobian

$$\boldsymbol{A} = H(q_2)EA \begin{bmatrix} 0 & rq_2 \\ 0 & rq_1 \end{bmatrix} \tag{3.2.2}$$

We will differentiate between desired accelerations $\ddot{\boldsymbol{y}}_d$ present in the system without accounting for cable elasticity and superfluous accelerations $\ddot{\boldsymbol{y}}_s = \ddot{\boldsymbol{q}}_s$ corresponding to $\boldsymbol{e} = \boldsymbol{A}_s\boldsymbol{q}_s$. The modified equations of motion can then be written as

$$\boldsymbol{M}_d\ddot{\boldsymbol{y}}_d + \underbrace{\boldsymbol{M}_s\ddot{\boldsymbol{y}}_s}_{\boldsymbol{0}} + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{p} + \boldsymbol{A}_s\Delta\boldsymbol{q}_s + \boldsymbol{B}\boldsymbol{u} \tag{3.2.3}$$

where $\Delta\boldsymbol{q}_s$ are the aforementioned expected changes in deformation. In addition we will subject $\boldsymbol{q}_s + \Delta\boldsymbol{q}_s$ to the constraint

$$\boldsymbol{q}_{s_{\min}} \leq \boldsymbol{q}_s + \Delta\boldsymbol{q}_s \leq \boldsymbol{q}_{s_{\max}} \tag{3.2.4}$$

so that a certain degree of pre-tension can be expected.

Adding the usual constraints on inputs $\boldsymbol{u}$ we may write the QP problem of approximating inverse dynamics while maintaining tension within the cables as optimizing parameters

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{u} \\ \Delta\boldsymbol{q}_s \end{bmatrix} \tag{3.2.5}$$

with objective function terms

$$\boldsymbol{P} = \begin{bmatrix} \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad , \quad \boldsymbol{q} = \mathbf{0} \tag{3.2.6}$$

and constraints' terms

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{\tau}_{\text{eq}} \\ \boldsymbol{u}_{\min} \\ \boldsymbol{q}_{s_{\min}} - \boldsymbol{q}_s \end{bmatrix} \quad , \quad \boldsymbol{A} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}_s \\ \mathbf{1} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \quad , \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{\tau}_{\text{eq}} \\ \boldsymbol{u}_{\max} \\ \boldsymbol{q}_{s_{\max}} - \boldsymbol{q}_s \end{bmatrix} \tag{3.2.7}$$

where

$$\boldsymbol{\tau}_{\text{eq}} = \boldsymbol{M}_d\ddot{\boldsymbol{y}}_d + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p} \tag{3.2.8}$$

which most resembles the problem described in section 2.2.2.

Chapter 4

Cubic Hermite Splines

When approximating inverse dynamics of cable driven manipulators we will be using Hermite $\mathbb{C}^1$ continuous splines as desired trajectories. Segments of these splines are cubic Hermite curves defined by the function value $\boldsymbol{P}$ and its first derivative $\dot{\boldsymbol{P}}$ at both ends of the segments. For the resulting splines to be $\mathbb{C}^1$ continuous adjacent segment must share not only the function value but also the first derivative [17].

4.1 Uniform Curves

Let $\boldsymbol{P}(t)\,,\ t \in \langle 0,1\rangle$ be a cubic Hermite curve defined by control values $\boldsymbol{P}_0$, $\dot{\boldsymbol{P}}_0$, $\boldsymbol{P}_1$, $\dot{\boldsymbol{P}}_1$ where $\boldsymbol{P}(t)$ must satisfy

$$\boldsymbol{P}_0 = \boldsymbol{P}(0)\,, \quad \dot{\boldsymbol{P}}_0 = \dot{\boldsymbol{P}}(0)\,, \quad \boldsymbol{P}_1 = \boldsymbol{P}(1)\,, \quad \dot{\boldsymbol{P}}_1 = \dot{\boldsymbol{P}}(1) \tag{4.1.1}$$

Such curve can be written in the form

$$\boldsymbol{P}(t) = \boldsymbol{C}\boldsymbol{M}_H\boldsymbol{\phi}(t)\,, \quad t \in \langle 0,1\rangle \tag{4.1.2}$$

where $\boldsymbol{C}$ is the control matrix

$$\boldsymbol{C} = \begin{bmatrix} \boldsymbol{P}_0 & \dot{\boldsymbol{P}}_0 & \boldsymbol{P}_1 & \dot{\boldsymbol{P}}_1 \end{bmatrix} \tag{4.1.3}$$

$\boldsymbol{M}_H$ the basis matrix and $\boldsymbol{\phi}(t)$ the basis polynomial. Unlike in [17] where $\boldsymbol{\phi}(t)$ is is chosen as the monomial basis $\boldsymbol{\phi}(t) = [\ 1\ t\ t^2\, t^3\ ]^T$ we will use a basis which can be expressed in the summation form as

$$\phi_k(t) = \frac{t^k}{k!}\,, \quad k \in \langle 0,3\rangle\,,\ k \in \mathbb{N} \tag{4.1.4}$$

its l-th derivative being

$$\frac{d^l\phi_k}{dt^l}(t) = \begin{cases} \frac{t^{k-l}}{(k-l)!}, & k \geq l \\ 0, & k < l \end{cases} \tag{4.1.5}$$

To determine the values of $\boldsymbol{M}_H$ we must solve the system of equations 4.1.1 which can be written in matrix form as

$$\underbrace{\begin{bmatrix} \boldsymbol{P}_0 & \dot{\boldsymbol{P}}_0 & \boldsymbol{P}_1 & \dot{\boldsymbol{P}}_1 \end{bmatrix}}_{\boldsymbol{C}} = \boldsymbol{C}\,\boldsymbol{M}_H\,\underbrace{\begin{bmatrix} \boldsymbol{\phi}(0) & \dot{\boldsymbol{\phi}}(0) & \boldsymbol{\phi}(1) & \dot{\boldsymbol{\phi}}(1) \end{bmatrix}}_{\boldsymbol{\Phi}_C} \tag{4.1.6}$$

from equation 4.1.6 it is evident that $\boldsymbol{M}_H$ must satisfy $\boldsymbol{M}_H = \boldsymbol{\Phi}_C^{-1}$, therefore

$$\boldsymbol{M}_H = \begin{bmatrix} 1 & 0 & -6 & 12 \\ 0 & 1 & -4 & 6 \\ 0 & 0 & 6 & -12 \\ 0 & 0 & -2 & 6 \end{bmatrix} \tag{4.1.7}$$

4.2 Non-Uniform Curves

Non-uniform cubic Hermite curves $\boldsymbol{P}(t)$, $t \in \langle t_0, t_1 \rangle$, $(t_0, t_1) \in \mathbb{R}^2$ are a generalized variant of the previously derived uniform curves which we will also define using control values $\boldsymbol{P}_0$, $\dot{\boldsymbol{P}}_0$, $\boldsymbol{P}_1$, $\dot{\boldsymbol{P}}_1$ where $\boldsymbol{P}(t)$ must satisfy

$$\boldsymbol{P}_0 = \boldsymbol{P}(t_0) \ , \quad \dot{\boldsymbol{P}}_0 = \dot{\boldsymbol{P}}(t_0) \ , \quad \boldsymbol{P}_1 = \boldsymbol{P}(t_1) \ , \quad \dot{\boldsymbol{P}}_1 = \dot{\boldsymbol{P}}(t_1) \tag{4.2.1}$$

We will re-use matrices $\boldsymbol{C}$ and $\boldsymbol{M_H}$ when defining the form of non-uniform cubic Hermite curves while introducing the normalization matrix $\boldsymbol{D}$ and selecting a new polynomial basis $\boldsymbol{\psi}$ that is beside t also a dependent on constants t_0 and t_1

$$\boldsymbol{P}(t) = \boldsymbol{C}\boldsymbol{D}(t_0, t_1)\boldsymbol{M}_H\boldsymbol{\psi}(t_0, t_1, t) \ , \quad t \in \langle t_0, t_1 \rangle \tag{4.2.2}$$

where the form of $\boldsymbol{D} = \boldsymbol{D}(t_0, t_1)$ is yet to be determined and $\boldsymbol{\psi}(t_0, t_1, t)$ can be written in summation form as

$$\psi_k(t_0, t_1, t) = \phi_k\left(\frac{t - t_0}{t_1 - t_0}\right) = \frac{(t - t_0)^k}{k!\,(t_1 - t_0)^k} \tag{4.2.3}$$

Although l-th derivative of $\boldsymbol{\psi}(t_0, t_1, t)$ can be derived directly as

$$\frac{d^l\psi_k}{dt^l}(t_0, t_1, t) = \begin{cases} \frac{(t-t_0)^{k-l}}{(k-l)!\,(t_1-t_0)^k}, & k \geq l \\ 0, & k < l \end{cases} \tag{4.2.4}$$

for the purpose of determining the form of $\boldsymbol{D}(t_0, t_1)$ we will also express it using the l-th derivate of $\boldsymbol{\phi}(t)$

$$\frac{d^l\psi_k}{dt^l}(t_0, t_1, t) = \begin{cases} \frac{1}{(t_1-t_0)^l}\frac{d^l\phi_k}{dt^l}\left(\frac{t-t_0}{t_1-t_0}\right) & , \ k \geq l \\ 0 & , \ k < l \end{cases} \tag{4.2.5}$$

System of equations 4.2.1 defining the curve can then be written in matrix form as

$$\underbrace{\begin{bmatrix}\boldsymbol{P}_0 & \dot{\boldsymbol{P}}_0 & \boldsymbol{P}_1 & \dot{\boldsymbol{P}}_1\end{bmatrix}}_{\boldsymbol{C}} = \boldsymbol{C}\,\boldsymbol{D}\,\boldsymbol{M}_H\,\underbrace{\begin{bmatrix}\boldsymbol{\psi}(t_0) & \dot{\boldsymbol{\psi}}(t_0) & \boldsymbol{\psi}(t_1) & \dot{\boldsymbol{\psi}}(t_1)\end{bmatrix}}_{\boldsymbol{\Psi}_C} \tag{4.2.6}$$

If we define $\boldsymbol{D}(t_0, t_1)$ so that $\boldsymbol{\Psi}_C = \boldsymbol{\Phi}_C\boldsymbol{D}^{-1}$ equation 4.2.6 will be satisfied as from section 4.2 we know that $\boldsymbol{M}_H = \boldsymbol{\Phi}_C^{-1}$. From equation 4.2.5 it can then be derived that

$$\boldsymbol{D} = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & t_1 - t_0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & t_1 - t_0\end{bmatrix} \tag{4.2.7}$$

4.3 Splines

A $\mathbb{C}^1$ continuous cubic Hermite spline can be created by defining values

$$\boldsymbol{P}_i = \boldsymbol{P}(t_i)\,,\ \dot{\boldsymbol{P}}_i = \dot{\boldsymbol{P}}(t_i)\ , \quad i \in \{0, \ldots, N\} \tag{4.3.1}$$

and constructing N non-uniform cubic Hermite curves

$$\boldsymbol{P}_{i-1,i}(t) = \begin{bmatrix}\boldsymbol{P}_{i-1} & \dot{\boldsymbol{P}}_{i-1} & \boldsymbol{P}_i & \dot{\boldsymbol{P}}_i\end{bmatrix}\boldsymbol{M}_H\boldsymbol{D}(t_{i-1}, t_i)\,\boldsymbol{\psi}(t_{i-1}, t_i, t) \tag{4.3.2}$$

for

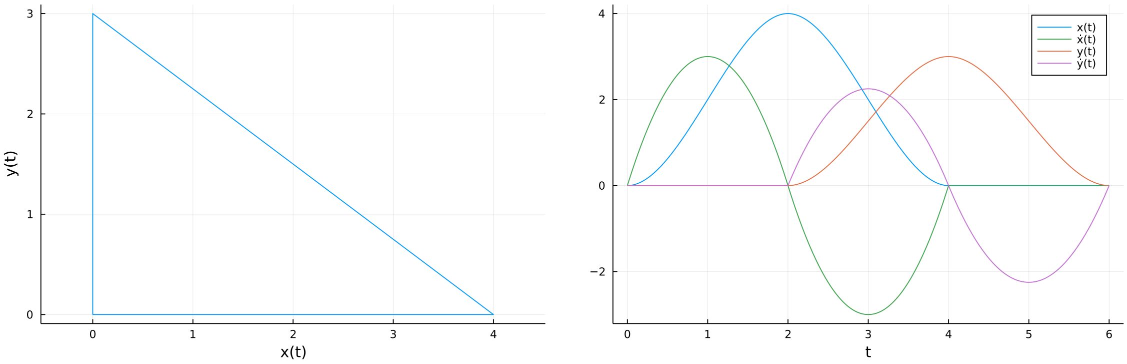
$$t \in \langle t_{i-1}, t_i\rangle\ , \quad i \in \{1, \ldots, N\} \tag{4.3.3}$$

where their l-th derivate can be simply obtained by substituting $\boldsymbol{\psi}(t_{i-1}, t_i, t)$ for $\frac{d^l\boldsymbol{\psi}}{dt^l}(t_{i-1}, t_i, t)$.

In figure 4.1a we may see an example of such $\mathbb{C}^1$ continuous spline $\boldsymbol{P}(t) = [\ x(t)\ y(t)\ ]^T$ defined by control values in table 4.1

t_i	$x(t_i)$	$y(t_i)$	$\dot{x}(t_i)$	$\dot{y}(t_i)$
0	0	0	0	0
2	4	0	0	0
4	0	3	0	0
6	0	0	0	0

Table 4.1: control values for the spline in figure 4.1



(a) $x(t) - y(t)$

(b) $t - \boldsymbol{P}(t), \dot{\boldsymbol{P}}(t)$

Figure 4.1: example of a cubic Hermite spline

Chapter 5

Application on Tensegrity Structures

5.1 Modelled Tensegrity Structures

For the purposes of this thesis two class 2 tensegrity structures were modelled in RBD.jl their bodies being generated using geometric primitives. To keep the descriptions of the structures concise a table of bodies including their name, primitive type and dimensions will be provided for each model. Table 5.1 contains all primitive types used throughout this section with their dimensions written in a specific order.

type	dimensions
cylinder	radius, height
cube	side
tube	outer diameter, inner diameter, length
box	side a, side b, side c

Table 5.1: types and properties of used geometric primitives

One way to view table 5.1 is to imagine that each row corresponds to the signature of a constructor function, for example `Cylinder(height,diameter)` for each rigid body, hence the specific order.

By convention the mass of all elements coloured red or grey in figures will be omitted while the remaining bodies will share a common density ρ its value defined on a per-model basis. The structures' cables will be characterized by their Young's module of elasticity E, damping coefficient μ and radius r from which their cross sections' area can be derived as $A = \pi r^2$. As E and μ greatly influence the behaviour of a system they will be defined separately for each simulation. All remaining values of properties describing a structure will be listed in a table.

5.1.1 Single Strut Tensegrity

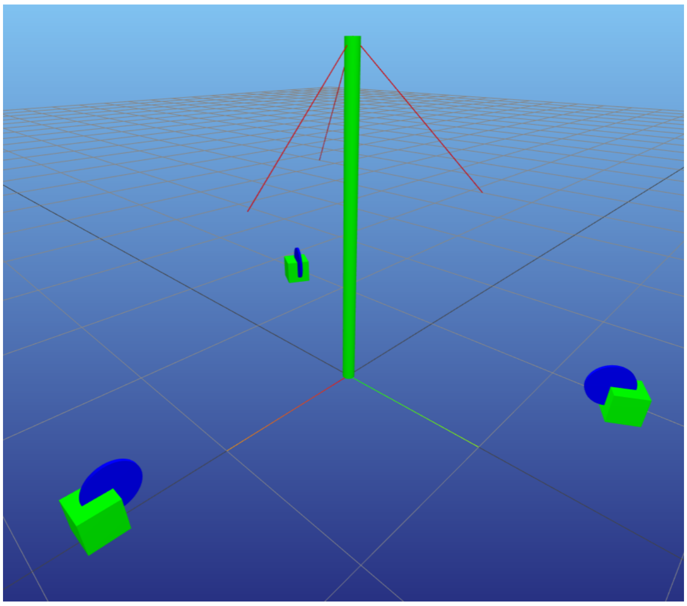


Figure 5.1: single-strut tensegrity

body/bodies	primitive type	dimensions
strut	cylinder	$D/2$, l
pulleys	cylinder	R, h
blocks	cube	Q

Table 5.2: bodies of the single strut tensegrity

The single strut tensegrity captured in figure 5.1 can be described as consisting of a single strut connected to the base of the system by an universal joint (in RBD.jl implemented as a massless cross shaft and two revolute joints) and three branches each containing a block, pulley and cable where $Q = \sqrt{2}R$. The revolute joints' axis, by which the blocks are connected to the base of the system are arranged symmetrically around the universal joint at the distance of $a = l - Q$ so that if the strut is vertical the structure's cables are at a 45° angle.

property	value
ρ	$2.7 \times 10^3\,\mathrm{kg\,m^{-3}}$
l	$1\,\mathrm{m}$
D	$0.04\,\mathrm{m}$
R	$0.07\,\mathrm{m}$
h	$0.01\,\mathrm{m}$
r	$1 \times 10^{-3}\,\mathrm{m}$

Table 5.3: properties of the single strut tensegrity

The resulting mechanism has five degrees of freedom, captured by angles φ, ψ of the universal joint and one superfluous degree of freedom for every cable.

5.1.2 Tensegrity Manipulator

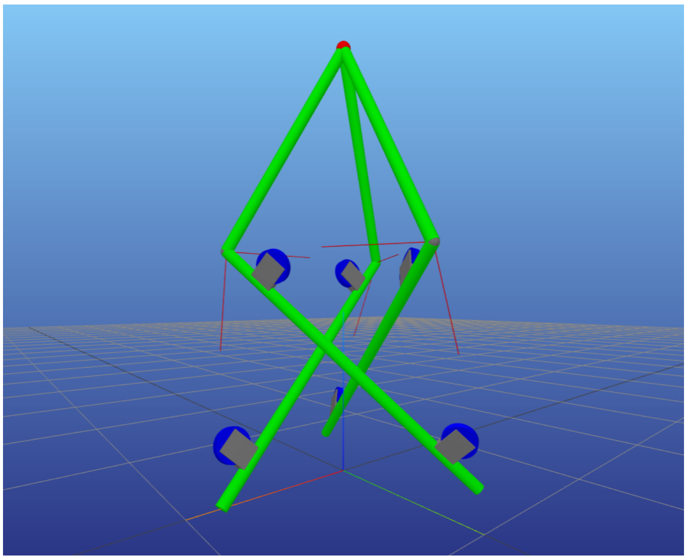


Figure 5.2: tensegrity manipulator

body/bodies	primitive type	dimensions
strut	tube	D, d, l
pulleys	cylinder	R, $2h$
blocks	box	$3h$, $2Q$, Q

Table 5.4: bodies of the tensegrity manipulator

Second model created for this thesis was that of a tensegrity manipulator which takes the form of a three strut tensegrity simplex (figure 5.3) with additional struts, attached those of the simplex using spherical joints, intersecting in a point to which we will refer to as the end-effector. The simplex itself is connected to the base of the system using the spherical joints placed in the shape of an equilateral triangle its sides being of length a. The six cables of the system are wound upon pulleys each connected to a block that rotates around the axis of the strut it is connected to.

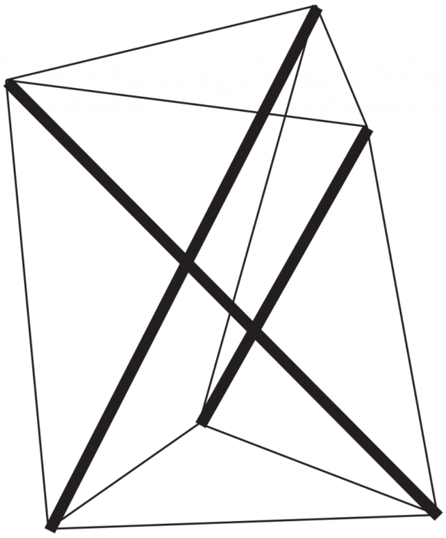


Figure 5.3: three strut tensegrity simplex
https://publish.illinois.edu/tie4rhino/what-is-tie-for-rhino/

dimension	value
ρ	$2.7 \times 10^{3}\,\mathrm{kg\,m^{-3}}$
l	$1\,\mathrm{m}$
a	$\frac{\sqrt{2}}{2}l$ m
R	$0.05\,\mathrm{m}$
h	$0.01\,\mathrm{m}$
D	$0.035\,\mathrm{m}$
d	$0.032\,\mathrm{m}$
Q	$\sqrt{2}R$ m
r	$1 \times 10^{-3}\,\mathrm{m}$

Table 5.5: properties of the tensegrity manipulator

The resulting mechanism has twelve degrees of freedom, six degrees of freedom one per each pulley and further six due to the cables' compliance.

5.1.3 Topology in RigidBodyDynamics.jl

In RBD.jl the mechanism's topology is determined by the order in which its joints are included. First bodies are attached using tree-joints to the existing mechanism, creating a kinematic tree with loop-joints being included last. The creation of branches and closing of loops can be done within for loops, simplifying the process. The resulting topologies, corresponding to scripts `SingleStrut/SingleStrut.jl` and `TensegrityManipulator/TensegrityManipulator.jl` in the appendix, can be seen in listings 5.1 and 5.2.

```
Spanning tree:
Vertex: world (root)
  Vertex: cross-shaft, Edge: x-revoluteJt
    Vertex: strut, Edge: y-revoluteJt
  Vertex: B1-block, Edge: B1-blockJt
    Vertex: B1-pulley, Edge: B1-pulleyJt
      Vertex: B1-cable, Edge: B1-involuteJt
  Vertex: B2-block, Edge: B2-blockJt
    Vertex: B2-pulley, Edge: B2-pulleyJt
      Vertex: B2-cable, Edge: B2-involuteJt
  Vertex: B3-block, Edge: B3-blockJt
    Vertex: B3-pulley, Edge: B3-pulleyJt
      Vertex: B3-cable, Edge: B3-involuteJt
Non-tree joints:
B1-loopJt, predecessor: strut, successor: B1-cable
B2-loopJt, predecessor: strut, successor: B2-cable
B3-loopJt, predecessor: strut, successor: B3-cable
```

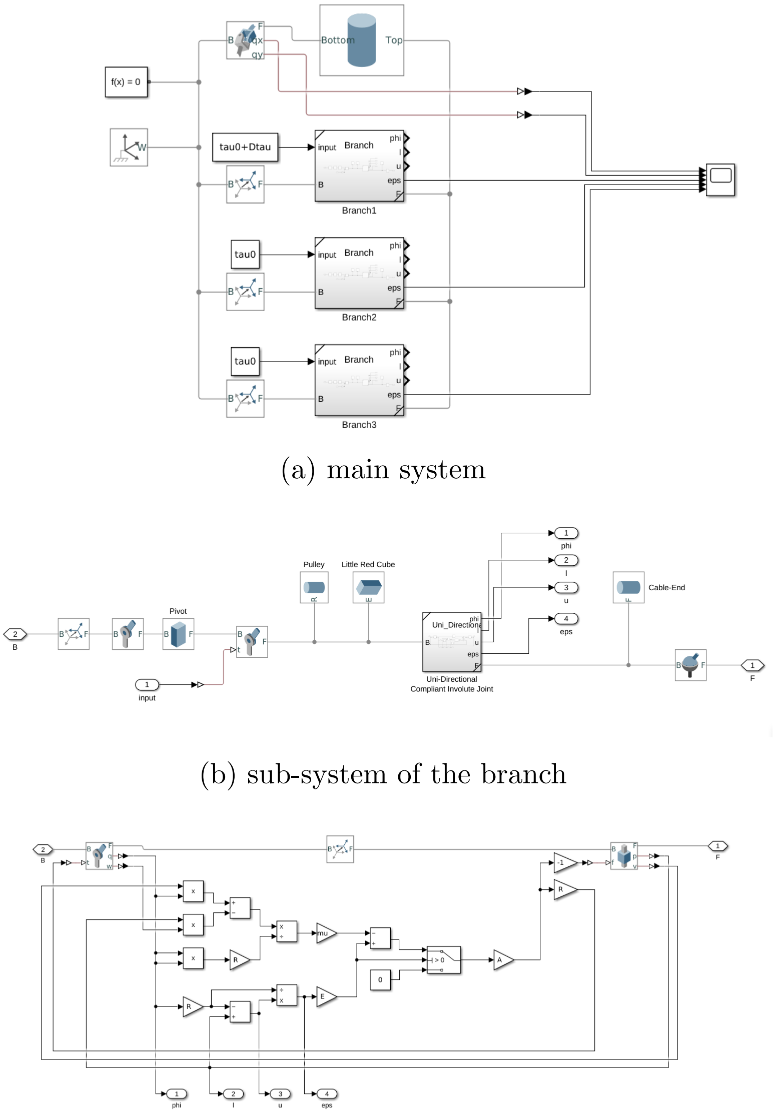
Listing 5.1: topology of the single strut tensegrity in RBD.jl

```
Spanning tree:
Vertex: world (root)
  Vertex: end-effector, Edge: floatingJt
    Vertex: B1-link, Edge: B1-linkJt
    Vertex: B2-link, Edge: B2-linkJt
    Vertex: B3-link, Edge: B3-linkJt
  Vertex: B1-dummy, Edge: B1-dummyJt
    Vertex: B1-strut, Edge: B1-strutJt
      Vertex: B1-top-block, Edge: B1-top-blockJt
        Vertex: B1-top-pulley, Edge: B1-top-revoluteJt
          Vertex: B1-top-cable, Edge: B1-top-involuteJt
      Vertex: B1-bot-block, Edge: B1-bot-blockJt
        Vertex: B1-bot-pulley, Edge: B1-bot-revoluteJt
          Vertex: B1-bot-cable, Edge: B1-bot-involuteJt
  Vertex: B2-dummy, Edge: B2-dummyJt
    Vertex: B2-strut, Edge: B2-strutJt
      Vertex: B2-top-block, Edge: B2-top-blockJt
        Vertex: B2-top-pulley, Edge: B2-top-revoluteJt
          Vertex: B2-top-cable, Edge: B2-top-involuteJt
      Vertex: B2-bot-block, Edge: B2-bot-blockJt
        Vertex: B2-bot-pulley, Edge: B2-bot-revoluteJt
          Vertex: B2-bot-cable, Edge: B2-bot-involuteJt
  Vertex: B3-dummy, Edge: B3-dummyJt
    Vertex: B3-strut, Edge: B3-strutJt
      Vertex: B3-top-block, Edge: B3-top-blockJt
        Vertex: B3-top-pulley, Edge: B3-top-revoluteJt
          Vertex: B3-top-cable, Edge: B3-top-involuteJt
      Vertex: B3-bot-block, Edge: B3-bot-blockJt
        Vertex: B3-bot-pulley, Edge: B3-bot-revoluteJt
          Vertex: B3-bot-cable, Edge: B3-bot-involuteJt
Non-tree joints:
B1-top-loopJt, predecessor: B3-strut, successor: B1-top-cable
B1-bot-loopJt, predecessor: B3-strut, successor: B1-bot-cable
B2-top-loopJt, predecessor: B1-strut, successor: B2-top-cable
B2-bot-loopJt, predecessor: B1-strut, successor: B2-bot-cable
B3-top-loopJt, predecessor: B2-strut, successor: B3-top-cable
B3-bot-loopJt, predecessor: B2-strut, successor: B3-bot-cable
B1-level-loopJt, predecessor: B3-strut, successor: B1-link
B2-level-loopJt, predecessor: B1-strut, successor: B2-link
B3-level-loopJt, predecessor: B2-strut, successor: B3-link
```

Listing 5.2: topology of the tensegrity manipulator in RBD.jl

5.1.4 Topology in Simscape Multibody

The single strut tensegrity was also modelled in simscape multibody where mechanisms are described using block diagrams a completely different approach to RBD.jl. Here I made the use of subsystems, first creating a subsystem of the compliant involute joint's *l* variant (section 3.1.3) which I embedded within a subsystem of the *branch*. The main system then includes three instances of the *branch* in addition to other blocks (figure 5.4 and folder `SingleStrut-Simscape` of the appendix).



(a) main system

(b) sub-system of the branch

(c) sub-system of the compliant involute joint

Figure 5.4: simscape multibody model of the single-strut tensegrity

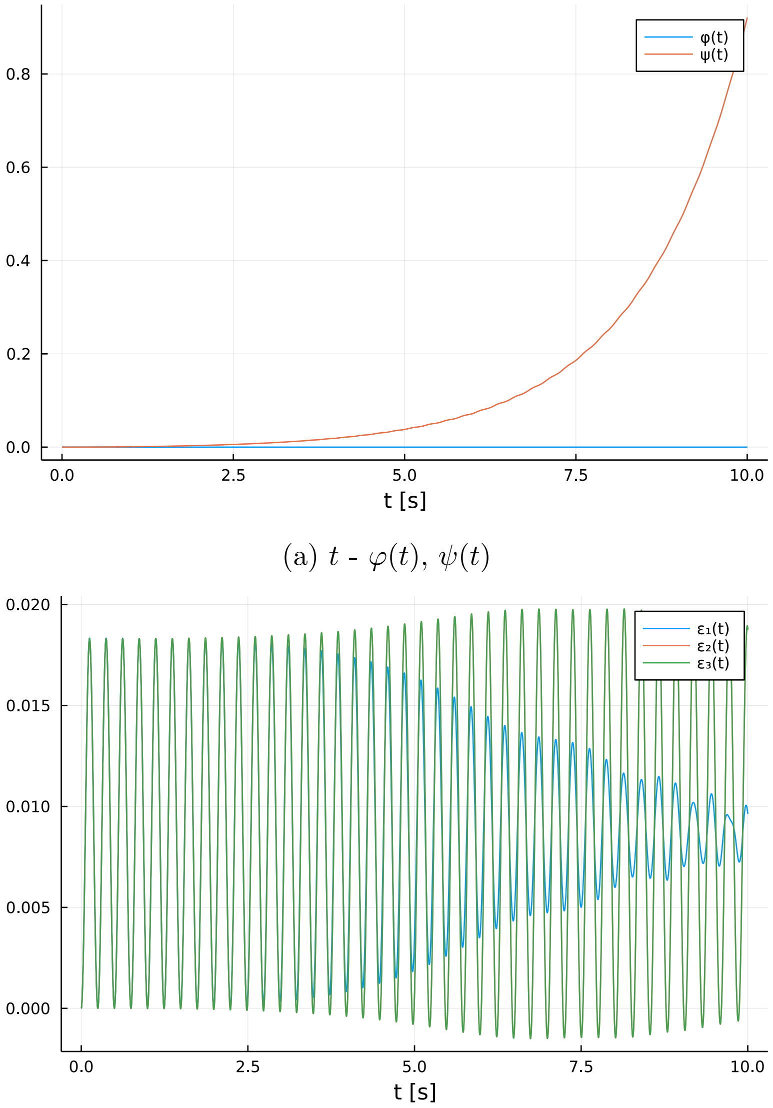
5.2 Comparison of Involute Joint's Variants

All three variants of the compliant involute joint were implemented in RBD.jl. In addition to creating the joint's variants, support for internal joint torques had to be added to the library, directly modifying its source code in the process. Without going into greater detail we will simply refer to the folder `RBD-Modifications` in the appendix which contains diffs between the upstream of RBD.jl and branches of the fork created for this thesis, each corresponding to one variant of the joint.

As mentioned before in simscape multibody only the l variant, specifically derived for the use in closed-source multibody dynamics libraries was implemented. In contrast to RBD.jl once derived the implementation was rather straight forward requiring the creation of a single subsystem displayed in figure 5.4c.

The comparison was performed on a model with Young's module of elasticity $E = 5 \times 10^1\,\mathrm{MPa}$ and no damping coefficient by prescribing constant torques to the revolute joints that connect pulleys to the blocks of individual branches, precisely $0.1001\,\mathrm{N\,m}$ to the first and $0.1\,\mathrm{N\,m}$ to the remaining two. This resulted in the strut rotating around the world frame's y-axis which was also reflected in the universal joint's angle ψ while φ remained constant (figure 5.5). Beside ψ and φ also strains ε_i of the cables were compared as can be seen in figure 5.6 where the ε variant was chosen a the reference. Due to how simscape multibody handles motion constraints, for the comparison plotted separately in figure 5.7, inertial properties had to be prescribed to the cable-ends, the model in RBD.jl also being modified.

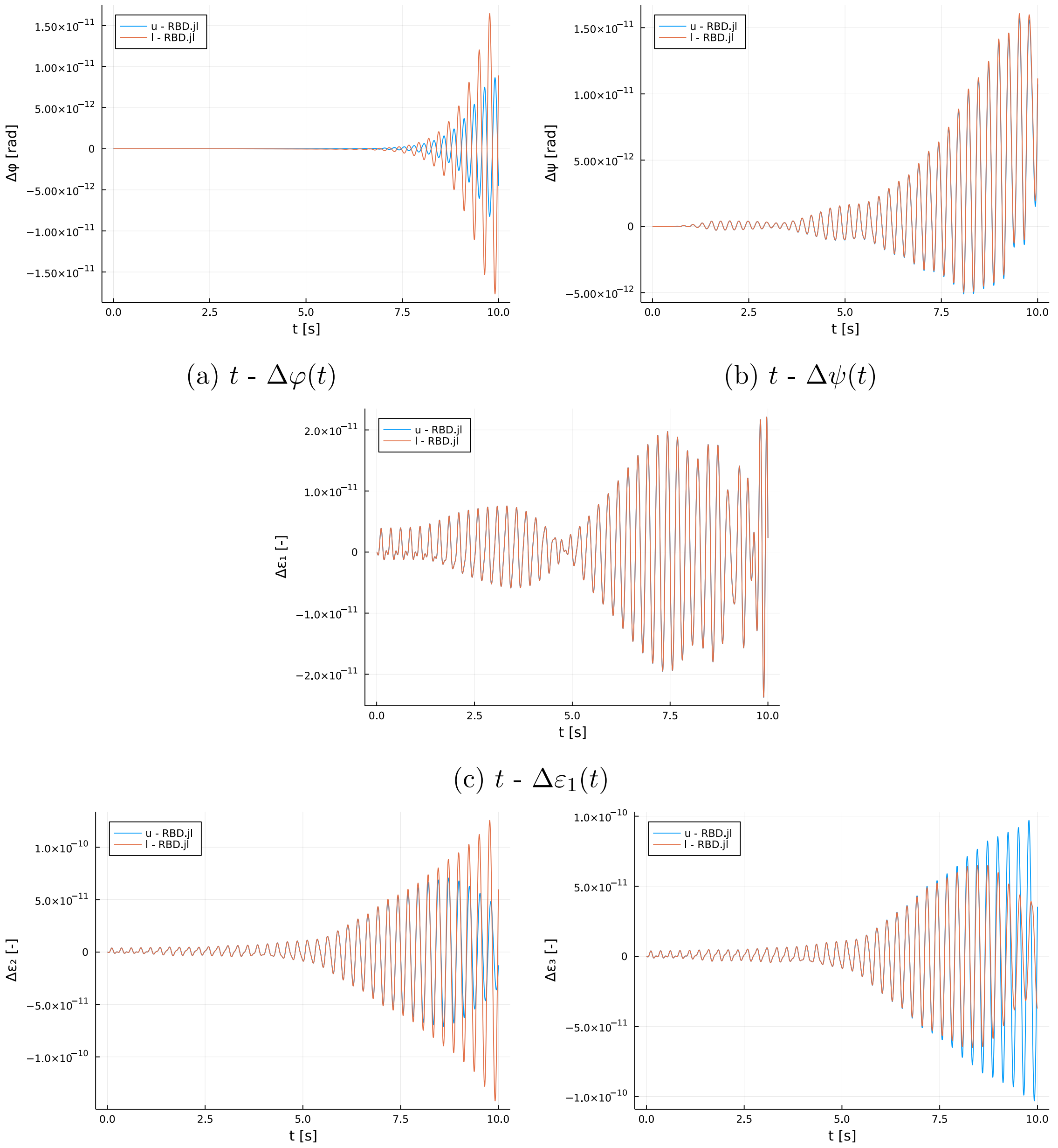
Evaluating figure 5.6 we may conclude that the deviations of results for individual variants implemented in RBD.jl are likely the result of numeric inaccuracies especially if we consider the fact that they accumulated across 10^4 steps of the simulation. Although deviations associated with the l variant implemented in simscape multibody (figure 5.7) are more significant the results in general are still comparable which is laudable considering the factors that cannot be controlled when working with a closed-source library. Overall I would consider implementations of all the variants equivalent in terms of forward dynamics simulations attributing the deviations to external factors.



(a) t - $\varphi(t)$, $\psi(t)$

(b) t - $\varepsilon_1(t)$, $\varepsilon_2(t)$, $\varepsilon_3(t)$

Figure 5.5: forward dynamics of the ε variant serving as a reference for the comparison



(a) t - $\Delta\varphi(t)$

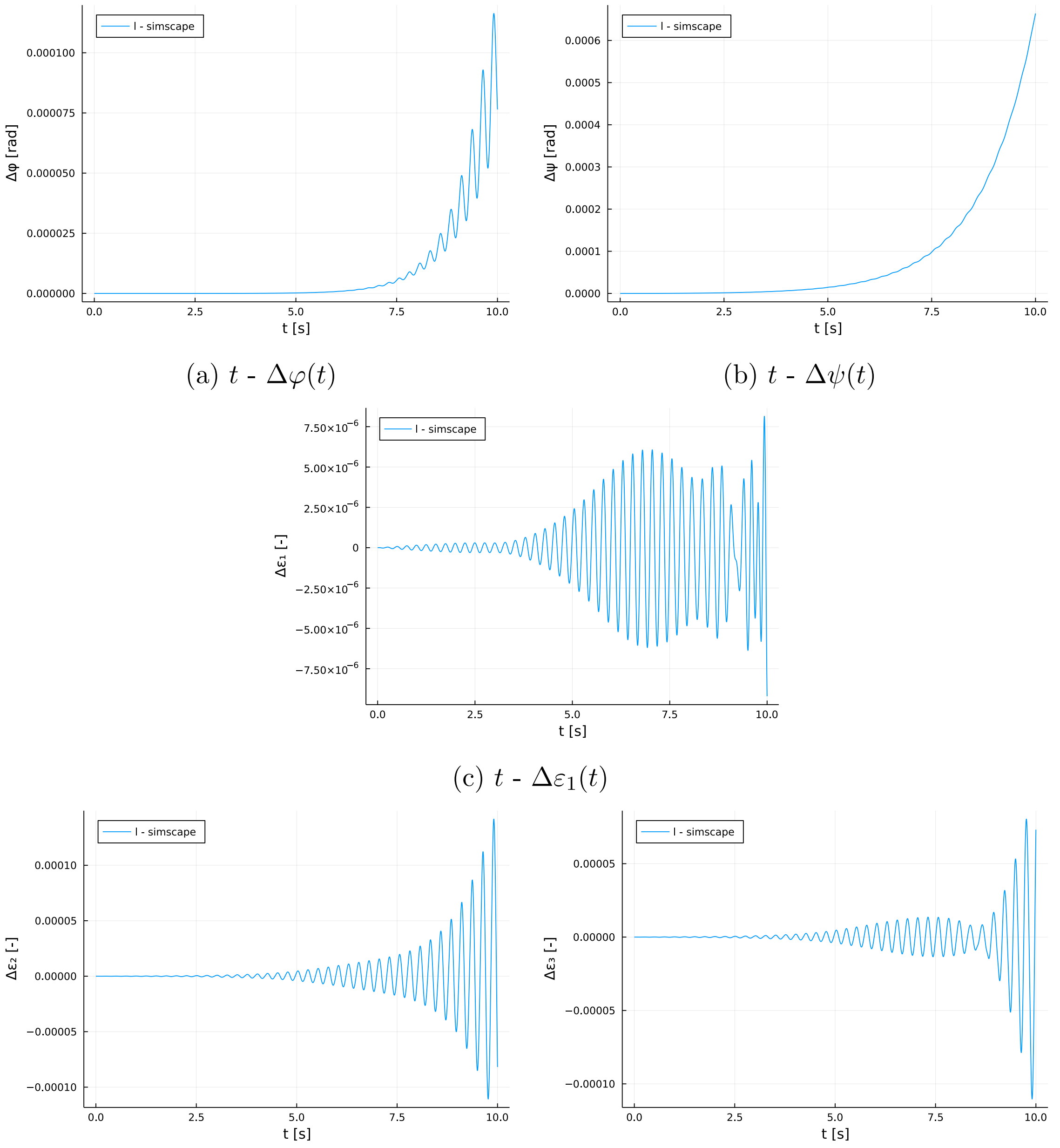
(b) t - $\Delta\psi(t)$

(c) t - $\Delta\varepsilon_1(t)$

(d) t - $\Delta\varepsilon_2(t)$

(e) t - $\Delta\varepsilon_3(t)$

Figure 5.6: deviations of compliant involute joint variants implemented in RBD.jl in comparison to the ε variant



(a) t - $\Delta\varphi(t)$

(b) t - $\Delta\psi(t)$

(c) t - $\Delta\varepsilon_1(t)$

(d) t - $\Delta\varepsilon_2(t)$

(e) t - $\Delta\varepsilon_3(t)$

Figure 5.7: deviations of the compliant involute joint variant implemented in simscape multibody in comparison to RBD.jl

5.3 Validating the Approximation of Inverse Dynamics

The approach to approximating inverse dynamics in section 3.2 was also validated using the single-strut tensegrity structure, particularly one with the ε variant of the compliant involute joint. A series of simulations for multiple values E was performed where the inverse dynamics were approximated for desired accelerations $\ddot{\boldsymbol{P}}(t)$ along a cubic Hermite spline $\boldsymbol{P}(t) = [\varphi_d(t)\ \psi_d(t)]^T$ (figure 5.8) defined by control values in table 5.6.

t_i	$\varphi_d(t_i)$	$\psi_d(t_i)$	$\dot{\varphi}_d(t_i)$	$\dot{\psi}_d(t_i)$
[s]	[rad]		[rad s^{-1}]	
0	0	0	0	0
1	0	0	0	0
3	0	$\frac{\pi}{4}$	0	0
7	$\frac{\pi}{4}$	0	0	0
9	0	0	0	0

Table 5.6: control values of the desired trajectory

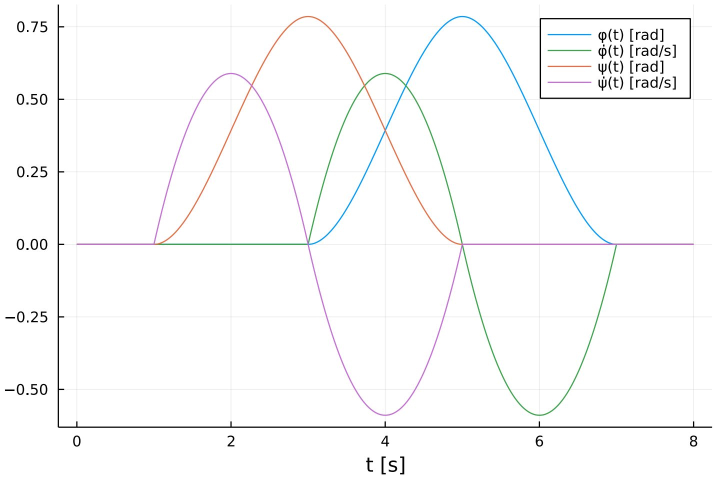


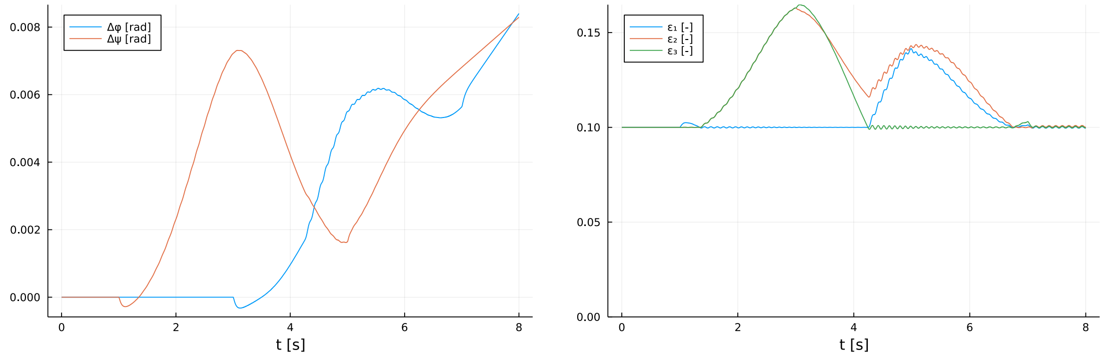
Figure 5.8: desired trajectory

At the start of the simulation all cables had an initial strain of $\varepsilon_0 = \frac{\sigma_0}{E}$ where $\sigma_0 = 20\,\mathrm{MPa}$ which also served as the lower bound for the cables' strain throughout the simulation while the input torques were left unbound. From the results of the simulation, series $\varphi(t)$, $\psi(t)$, $\varepsilon_1(t)$, $\varepsilon_2(t)$ and $\varepsilon_3(t)$ were extracted. Values of $\varphi(t)$ and $\psi(t)$ were then compared to the those on the desired trajectory where we may see the resulting series of deviations $\Delta\varphi(t)$, $\Delta\psi(t)$ as well as the cables' strains plotted in figures 5.9, 5.10 and 5.11 for different values of E. As a quantitative representation of how this approach performs for different values of E we extracted the maximal deviations (in absolute values) from the desired trajectory into table 5.7. Here we may see that the approximation's accuracy increases with the stiffness of the cable which is logical as to a certain degree it assumes the cable to be rigid.

E [MPa]	$\max\|\Delta\varphi(t)\|$ [m]	$\max\|\Delta\psi(t)\|$ [m]
2e2	8.403e-3	8.293e-3
2e3	1.080e-3	8.105e-4
2e4	2.162e-4	1.330e-4

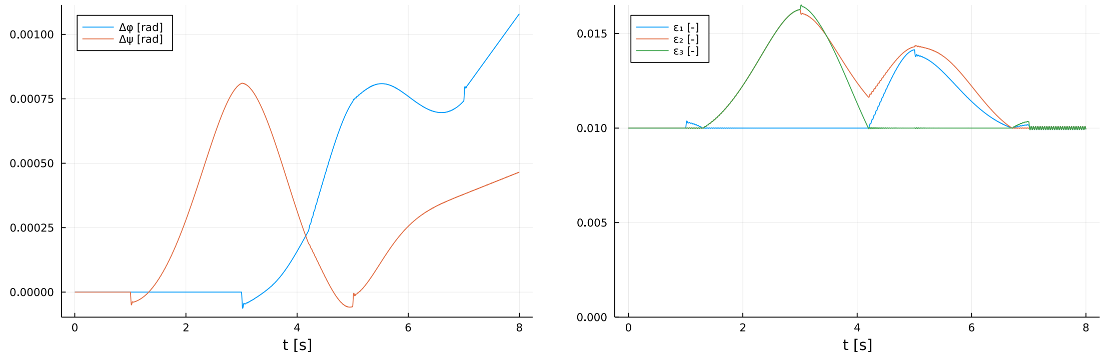
Table 5.7: overview of the simulation's results

Two separate julia packages implementing the cubic Hermite splines as well as the approximation of inverse dynamics were written as part of the thesis. In their entirety they can be found in the folders `Hermite3` and `NearRigidControl` of the appendix.



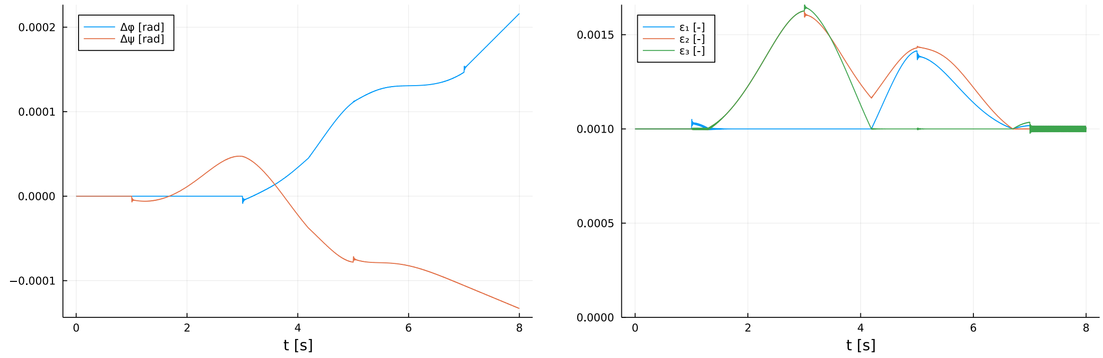
(a) t - $\Delta\varphi(t)$, $\Delta\psi(t)$ (b) t - $\varepsilon_1(t)$, $\varepsilon_2(t)$, $\varepsilon_3(t)$

Figure 5.9: $E = 2 \times 10^2$ MPa



(a) t - $\Delta\varphi(t)$, $\Delta\psi(t)$ (b) t - $\varepsilon_1(t)$, $\varepsilon_2(t)$, $\varepsilon_3(t)$

Figure 5.10: $E = 2 \times 10^3$ MPa



(a) t - $\Delta\varphi(t)$, $\Delta\psi(t)$ (b) t - $\varepsilon_1(t)$, $\varepsilon_2(t)$, $\varepsilon_3(t)$

Figure 5.11: $E = 2 \times 10^4$ MPa

5.4 Extended Approach to Inverse Dynamics Approximation

As was mentioned in section 5.1.2 the modelled tensegrity manipulator has a total of twelve degrees of freedom. A way in which we may divide the degrees of freedom is into desired (which we will control), superfluous (due to the cables' compliance) and free which will be controlled indirectly. As the desired degrees of freedom we will choose Cartesian accelerations of the end-effector $\ddot{\boldsymbol{y}}_d = [\ddot{x}\ \ddot{y}\ \ddot{z}]$. Then we will have six superfluous DoF, one per each cable $\ddot{\boldsymbol{y}}_s = [\ddot{\varepsilon}_1\ \dots\ \ddot{\varepsilon}_6]$ dealt with similarly as with those of the single-strut tensegrity in section 5.3. Lastly the remaining three free degrees of freedom will be chosen algorithmically from the remaining elements of the vector of tree-joint accelerations $\ddot{\boldsymbol{q}}$ so that certain internal matrices remain regular throughout the simulations. We may write the system's equations of motion with the newly introduced degrees of freedom as

$$\boldsymbol{M}_d\ddot{\boldsymbol{y}}_d + \boldsymbol{M}_s\ddot{\boldsymbol{y}}_s + \boldsymbol{M}_f\ddot{\boldsymbol{y}}_f + \boldsymbol{c} = \boldsymbol{W}\boldsymbol{f} + \boldsymbol{p} + \boldsymbol{B}\boldsymbol{u} \tag{5.4.1}$$

Applying the same modifications as in equation 3.2.3 and including the vector of free accelerations $\ddot{\boldsymbol{y}}_f$ among the optimized parameters we may write the QP problem of approximating inverse dynamics while maintaining tension within the cables as optimizing parameters

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{u} \\ \Delta\boldsymbol{q}_s \\ \ddot{\boldsymbol{y}}_f \end{bmatrix} \tag{5.4.2}$$

with objective function terms

$$\boldsymbol{P} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad \boldsymbol{q} = \mathbf{0} \tag{5.4.3}$$

and constraints' terms

$$\boldsymbol{a} = \begin{bmatrix} \boldsymbol{\tau}_{\mathrm{eq}} \\ \boldsymbol{u}_{\min} \\ \boldsymbol{q}_{s_{\min}} - \boldsymbol{q}_s \end{bmatrix}, \quad \boldsymbol{A} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}_s & -\boldsymbol{M}_f \\ \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{\tau}_{\mathrm{eq}} \\ \boldsymbol{u}_{\max} \\ \boldsymbol{q}_{s_{\max}} - \boldsymbol{q}_s \end{bmatrix} \tag{5.4.4}$$

where

$$\boldsymbol{\tau}_{\text{eq}} = \boldsymbol{M}_d\ddot{\boldsymbol{y}}_d + \boldsymbol{c} - \boldsymbol{W}\boldsymbol{f} - \boldsymbol{p} \tag{5.4.5}$$

Two simulations were performed demonstrating the approache's ability to guide the manipulator's end-effector along desired trajectories in the form of $\mathbb{C}^1$ cubic Hermite splines with its cables having a Young's module of elasticity $E = 2 \times 10^3$ MPa and damping coefficient $\mu = 1 \times 10^1$ MPa. The first of the two trajectories, defined by control values in table 5.8, elicits movement along the world frame's vertical axis while the second (table 5.9) follows a line parallel to one of the horizontal axes

t_i	$x_d(t_i)$	$y_d(t_i)$	$z_d(t_i)$	$\dot{x}_d(t_i)$	$\dot{y}_d(t_i)$	$\dot{z}_d(t_i)$
[s]		[m]			[m s^{-1}]	
0	0	0	h_0	0	0	0
5	0	0	$h_0 - 0.5$	0	0	0
15	0	0	$h_0 + 0.25$	0	0	0
20	0	0	h_0	0	0	0

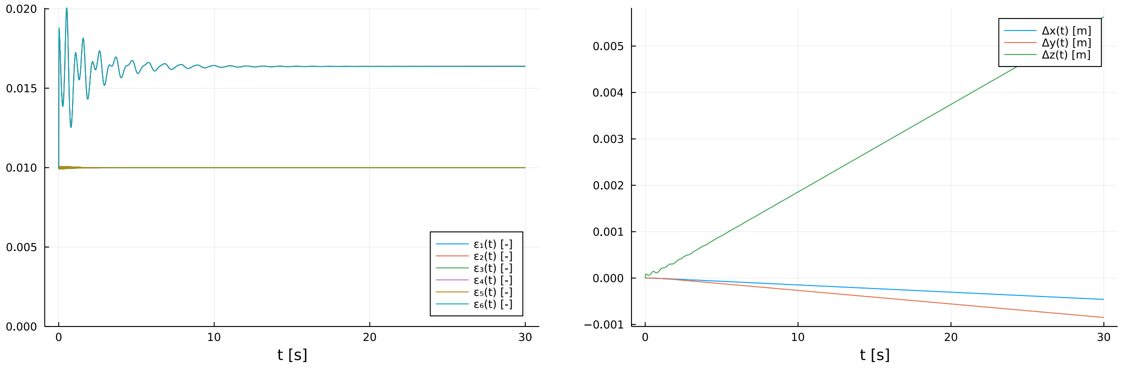
Table 5.8: control values of the desired trajectory z

t_i	$x_d(t_i)$	$y_d(t_i)$	$z_d(t_i)$	$\dot{x}_d(t_i)$	$\dot{y}_d(t_i)$	$\dot{z}_d(t_i)$
[s]		[m]			[m s^{-1}]	
0	0	0	h_0	0	0	0
5	0	-0.35	h_0	0	0	0
15	0	0.35	h_0	0	0	0
20	0	0	h_0	0	0	0

Table 5.9: control values of the desired trajectory x

where $h_0 = a\left(1 + \sqrt{\frac{2}{3}}\right)$.

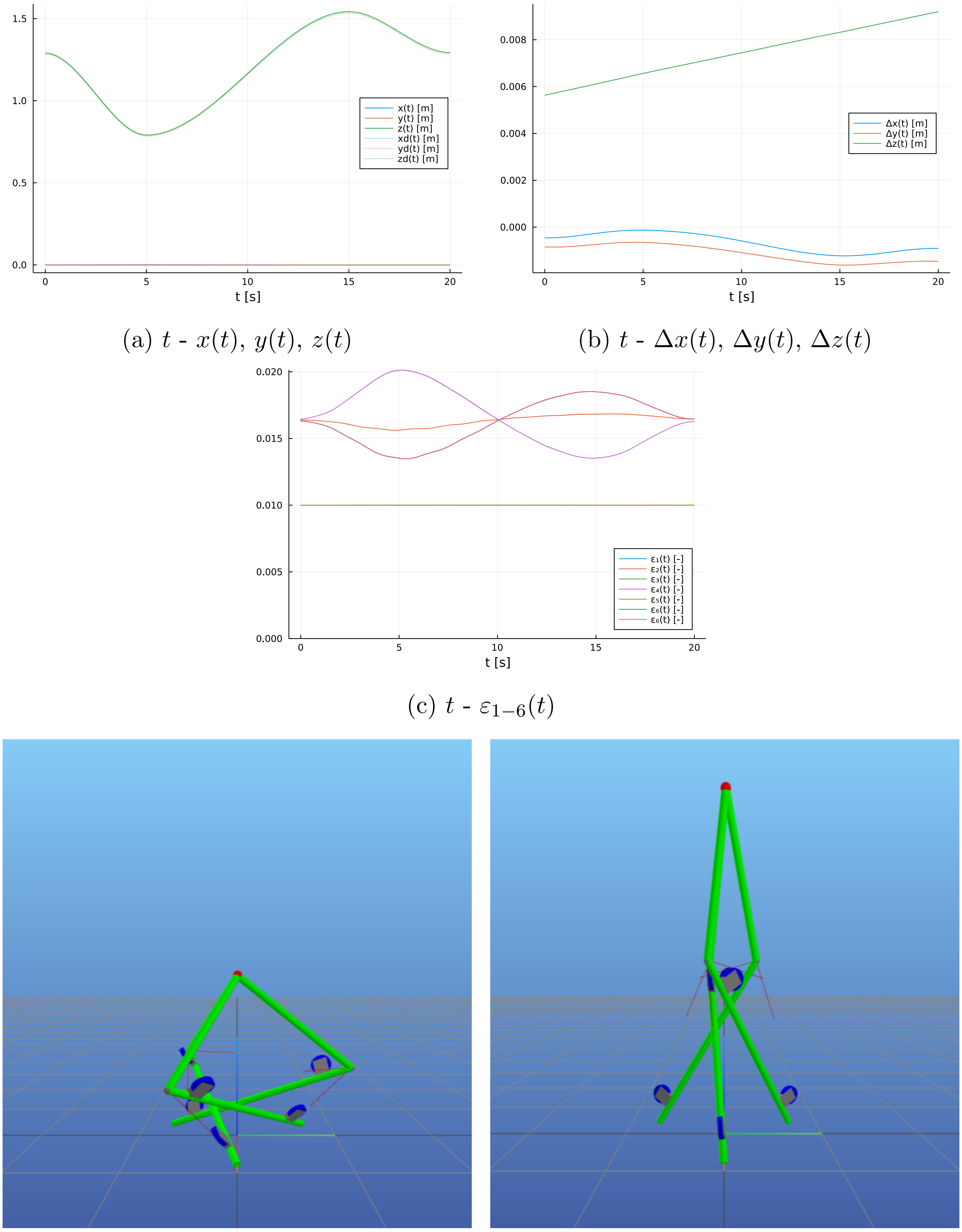
Whereas in section 5.3 the simulations start with the manipulator at a user defined initial state, the simulations of the tensegrity manipulator are preceded by a period in which the manipulator stabilizes itself in a singular configuration while reasonably maintaining the end-effectors position (figure 5.12).



(a) t - $\varepsilon_{1-6}(t)$ (b) t - $\Delta x(t)$, $\Delta y(t)$, $\Delta z(t)$

Figure 5.12: stabilization

As can be seen in figure 5.13 the trajectory along the vertical axis is followed with considerable accuracy while tracking along the horizontal axis in figure 5.14 is comparably worse. These excesive deviations are likely tied to the free degrees of freedom and infinitisimal mechanisms [6] of the structure as similar behaviour was not observed in section 5.3 on a structure with no free degrees of freedom. An explanation to why the movement along the vertical axis might not have been effected is the fact that it also is the manipulator's axis of symmetry. Regardless I would consider the further study of this phenomenon worthwhile as it might impose severe limitations to the accuracy of tensegrity of tensegrity manipulators' motion.



(a) t - $x(t)$, $y(t)$, $z(t)$

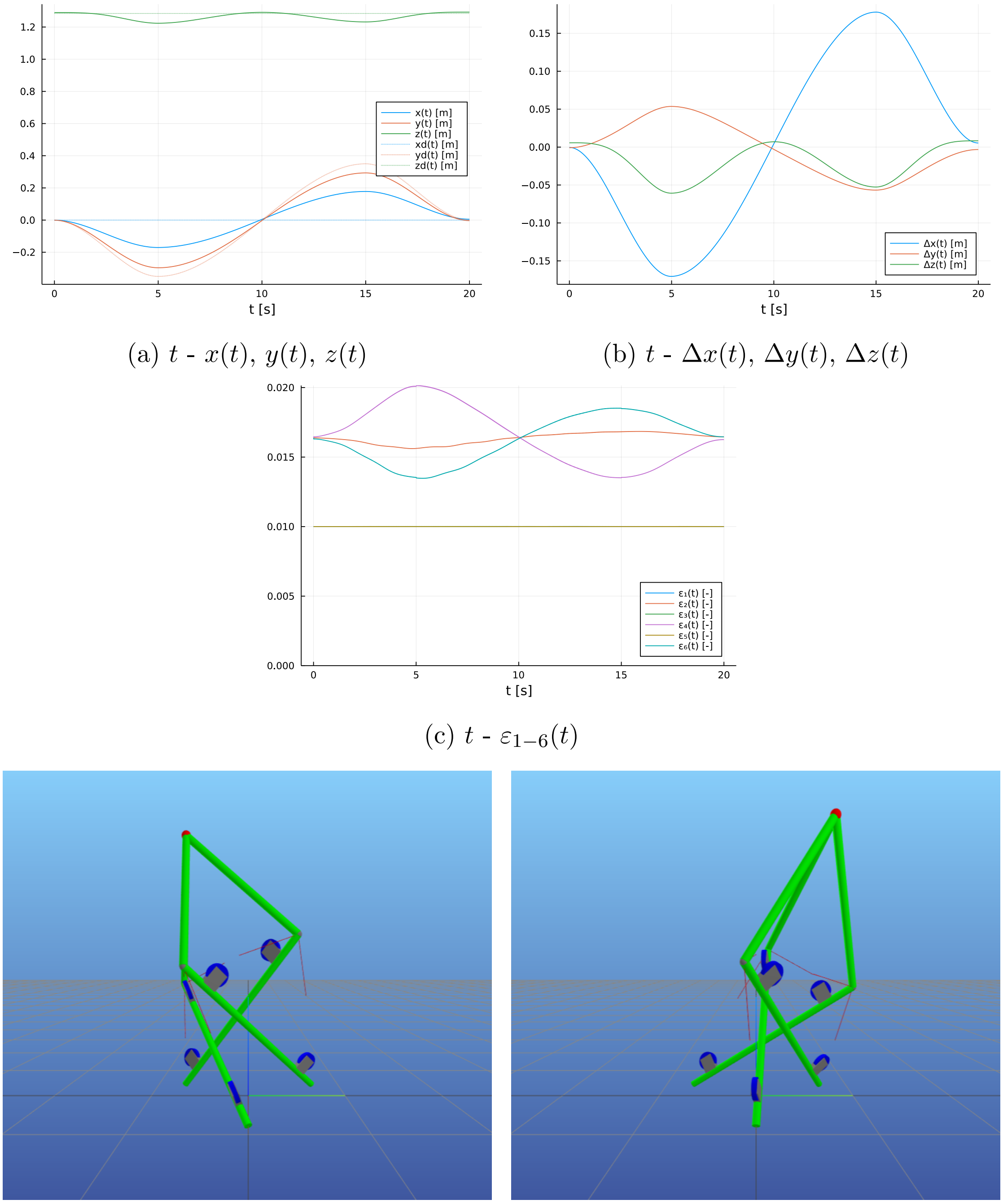
(b) t - $\Delta x(t)$, $\Delta y(t)$, $\Delta z(t)$

(c) t - $\varepsilon_{1-6}(t)$

(d) manipulator a $t = 5\,\mathrm{s}$

(e) manipulator a $t = 15\,\mathrm{s}$

Figure 5.13: control z



(a) t - $x(t)$, $y(t)$, $z(t)$

(b) t - $\Delta x(t)$, $\Delta y(t)$, $\Delta z(t)$

(c) t - $\varepsilon_{1-6}(t)$

(d) manipulator a $t = 5\,\mathrm{s}$

(e) manipulator a $t = 15\,\mathrm{s}$

Figure 5.14: control x

Conclusion

First objective of this thesis was gaining an understanding of how equations of motion of a closed-loop system can be algorithmically constructed. This was mainly done by reviewing Roy Featherstone's algorithms, synthesis of which is presented in chapter 1. A method for modelling the interactions between pulleys and compliant cables, which was the second objective is detailed in chapter 3.

Here three variants of what we refer to as the compliant involute joint were derived and implemented in an extended version of the julia programming language [10] package RigidBodyDynamics.jl [9] (its modifications are included in the appendix). One of the variants, specifically derived for the purpose, was also implemented in simscape multibody [8] as part of a simple tensegrity structure on which the equivalence of all of the variants was validated. The third objective of creating a model of a tensegrity manipulator including the aforementioned interactions was also achieved using RigidBodyDynamics.jl.

The last objective of developing an approach to acquiring a solution to what would be the inverse dynamics problem if the manipulator's cables were rigid was also completed, its theory detailed in section 3.2 and implemention in the package NearRigidControl.jl written as part of this thesis. Another package created during the writing of this thesis was Hermite3.jl which implements cubic Hermite splines as described in chapter 4. Both of these packages are also incluced in the appendix.

While the approach to control described in section 3.2 was perfectly applicable to the single strut tensegrity described in section 5.1.1 its extended version, derived in section 5.4 specifically for the tensegrity manipulator, exhibited unexpected behaviour I would consider worth studying further when eliciting motion along a trajectory parallel to one of the horizontal axes. I would also find comparing the approach to other methods of control such as model predictive control interesting although they themselves would likely have to be first modified due to cables offering little resistance to compression. Lastly I would like to create a joint that connects two pulleys using a compliant cable which would allow the modelling of more complex mechanisms.

Glossary

EoM equations of motion. 7, 10, 13, 30, 31, 53, 58

DoF degrees of freedom. 11, 40, 42, 53, 55, 61

frame of reference kinematic device that serves as a standard relative to which motion can be measured. 9

inertial frame reference-frame with a time-scale, relative to which the motion of a body not subject to forces is always rectilinear and uniform, accelerations are always proportional to and in the direction of applied forces, and applied forces are always met with equal and opposite reactions. 9

tree-joint joint included in the kinematic tree. 10, 43

loop-joint joint outside of the kinematic tree. 9, 10, 43

implicit motion constraints Constraints on a bodies motion through the use of Lagrangian multipliers. 12, 13

explicit motion constraints Constraints on a bodies motion enforced by expressing its state using a generalized coordinate. 12, 13

COP constrained optimization problem. 7, 17, 18

LP linear programing. 18

NLP non-linear programing. 18

QP quadratic programing. 18, 31, 53

RBD.jl RigidBodyDynamics.jl. 3, 7, 22, 38, 39, 43, 45, 46, 48, 49, 58

Symbols

$p(j)$ predecessor of joint j. 9, 10

$s(j)$ successor of joint j. 9, 10

$\mu(i)$ child set of body i. 9, 10

$\lambda(i)$ parent of body i. 9, 10

$\nu(i)$ subtree set of joint i. 9, 13

$\kappa(i)$ support set of body i. 9, 13

N_B number of rigid bodies. 10

N_J number of joints. 10

$\boldsymbol{I}_i$ spatial inertia of body i. 10

$\boldsymbol{a}_i$ acceleration of body i. 10, 12

$\boldsymbol{v}_i$ velocity of body i. 10, 12

$\boldsymbol{f}_i^x$ external force (if any) acting on body i. 10

$\boldsymbol{f}_j$ force transmited across joint j. 10

${}^B\boldsymbol{X}_A^*$ transformation of a force vector from frame A to frame B. 10

${}^B\boldsymbol{X}_A$ transformation of a motion vector from frame A to frame B. 10–12, 24

N number of DoF of an unconstrained rigid body. 11

N_C number of constraints imposed by a joint. 11

$\boldsymbol{T}_a$ active force subspace. 11, 12

$\boldsymbol{T}_c$ constraint force subspace. 11, 12

$\boldsymbol{\tau}$ vector of joint torques. 11, 14

$\boldsymbol{\lambda}$ vector of constraint forces variables. 11, 12, 14

$\boldsymbol{v}_J$ joint velocity. 11, 24

$\boldsymbol{q}$ vector of joint parameters. 11, 26, 27

$\boldsymbol{S}$ motion subspace. 11, 12, 24, 27–29

$\boldsymbol{a}_J$ joint acceleration. 11, 12, 25

$\boldsymbol{c}_J$ acceleration bias term. 12, 25, 27–29

$\boldsymbol{M}$ mass matrix. 14, 16

$\ddot{\boldsymbol{q}}$ vector of tree-joint accelerations. 14, 53

$\boldsymbol{c}$ vector of bias terms. 14, 16

$\boldsymbol{W}$ wrench jacobian matrix. 14, 16

$\boldsymbol{f}$ vector of external forces. 14, 16

$\boldsymbol{L}$ torque jacobian matrix. 14

$\boldsymbol{\tau}^l$ vector of loop-joint torques. 14

$\boldsymbol{K}$ constraint jacobian matrix. 14

$\boldsymbol{k}$ vector of constraint bias terms. 14

$\boldsymbol{G}$ independency jacobian matrix. 15

$\boldsymbol{g}$ independency bias. 15

$\boldsymbol{p}$ passive joint torques. 16

$\boldsymbol{u}$ vector of inputs. 16, 17, 19, 20, 31

$\boldsymbol{B}$ manipulator matrix. 16, 17, 20

$\ddot{\boldsymbol{y}}$ vector of independent accelerations. 16

n number of joint parameters. 15, 16

n_i number of independent joint parameters. 15, 16

n_a number of actuated joint parameters. 16

n_c number of constraints. 15

f objective function. 18–20, 31, 53

$\boldsymbol{x}$ vector of optimized parameters. 18–20

$\boldsymbol{N}$ null matrix. 20

ϕ angle of unwrapping. 23

l deformed length of the unwound cable. 23, 27

l_0 free length of the unwound cable. 23

U total strain energy. 25

Λ strain energy density. 25

V_0 volume before deformation. 25

ξ body coordinate. 25

u displacement. 25, 27

E Young's module of elasticity. 25, 38, 46, 50, 51, 54

μ damping coefficient. 26, 38, 46, 54

ε strain. 25, 27, 28

σ tension. 25

$\boldsymbol{e}$ vector of elasticity torques. 26–30

$\boldsymbol{d}$ vector of damping torques. 26–29

$\boldsymbol{A}$ accumulation Jacobian. 30

$\boldsymbol{P}$ function value of a spline or curve. 32, 34

$\boldsymbol{C}$ control matrix. 34

$\boldsymbol{M}_H$ basis matrix. 33

$\boldsymbol{\phi}$ basis polynomial of a uniform Hermite curve. 33

$\boldsymbol{\psi}$ basis polynomial of a non-uniform Hermite curve. 34

$\boldsymbol{D}$ normalization matrix. 34, 35

Bibliography

[1] Buckminister Richard Fuller. Tensile-integrity structures. Patent: US3063521A, 1959.

[2] Robert Skelton and Mauricio Oliveira. *Tensegrity Systems*. 01 2009.

[3] Chandana Paul, Francisco Valero-Cuevas, and Hod Lipson. Design and control of tensegrity robots for locomotion. *Robotics, IEEE Transactions on*, 22:944 – 957, 11 2006.

[4] Tomáš Kaňka. Tensegrity mechanisms for replacement of spatial serial robots. Master's thesis, 2021.

[5] Aleš Balon. Optimization and control of mechatronic tensegrity for robotics. Master's thesis, 2019.

[6] Cornel Sultan. Tensegrity deployment using infinitesimal mechanisms. *International Journal of Solids and Structures*, 51, 10 2014.

[7] Philipp Miermeister, Werner Kraus, Tian Lan, and Andreas Pott. *An Elastic Cable Model for Cable-Driven Parallel Robots Including Hysteresis Effects*, volume 32, pages 17–28. 08 2015.

[8] Simscape Documentation. Simscape multibody. https://www.mathworks.com/products/simscape-multibody.html, July 2022.

[9] Twan Koolen and contributors. Rigidbodydynamics.jl. 2016.

[10] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[11] Jeongseok Lee, Michael X. Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S. Srinivasa, Mike Stilman, and C. Karen Liu. Dart: Dynamic animation and robotics toolkit. *Journal of Open Source Software*, 3(22):500, 2018.

[12] Michael A. Sherman, Ajay Seth, and Scott L. Delp. Simbody: multibody dynamics for biomedical research. *Procedia IUTAM*, 2:241–261, 2011. IUTAM Symposium on Human Body Dynamics.

[13] Roy Featherstone. *Rigid Body Dynamics Algorithms*. 01 2008.

[14] Robert DiSalle. Space and Time: Inertial Frames. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2020 edition, 2020.

[15] Wikipedia contributors. Constrained optimization — Wikipedia, the free encyclopedia, 2022. [Online; accessed 26-July-2022].

[16] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. OSQP: an operator splitting solver for quadratic programs. *Mathematical Programming Computation*, 12(4):637–672, 2020.

[17] Hermite splines. https://splines.readthedocs.io/en/latest/euclidean/Hermite.html, Jan 2021. [accessed 24-July-2022].