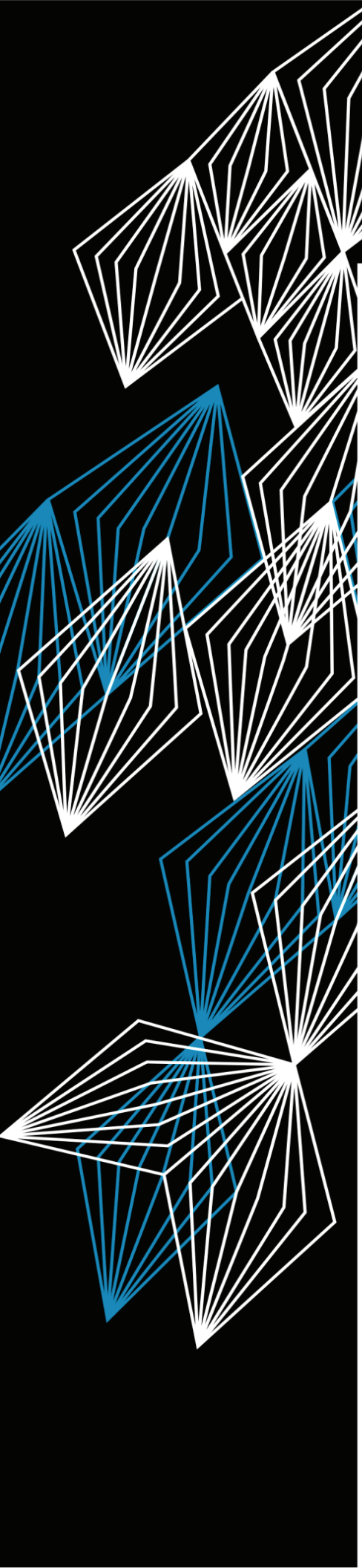

MASTER THESIS

Multi-Point Orientation Control of Discretely Magnetized Continuum Manipulators

MICHIEL RICHTER

FACULTY OF ENGINEERING TECHNOLOGY
DEPARTMENT OF BIOMECHANICAL ENGINEERING

EXAMINATION COMMITTEE
PROF. DR. SARTHAK MIRSA
DR. VENKATASUBRAMANIAN KALPATHY VENKITESWARAN
DR. IR. ARVID QUINTIJN LÉON KEEMINK

DOCUMENT NUMBER
BE-766

NOVEMBER 12, 2020

UNIVERSITY OF TWENTE.

Thesis submitted by
Michiel Richter
under the supervision of
Prof. dr. Sarthak Misra,
Dr. Venkatasubramanian Kalpathy Venkiteswaran, and
Dr. ir. Arvid Quintijn Léon Keemink
in order to fulfill the necessary requirements to obtain a Master's degree in
Biomedical Engineering at the University of Twente
and defended on
Thursday, 12th of November, 2020

Acknowledgement

This thesis is the product of ten months of work at the Surgical Robotics Laboratory (SRL), University of Twente, to achieve my Master's degree in Biomedical Engineering. The journey brought me past the theories on modelling of magnetic fields and magnetized flexible rods and beams to eventually apply this acquired knowledge onto a physical setup. I wish to thank a number of people without whom this work could not have been possible, fellow students with whom I have talked about our projects, and relatives who supported me throughout the past months.

First and foremost, I want to thank Prof. Dr. Sarthak Misra and Dr. Venkatasubramanian Kalpathy Venkiteswaran. I approached Prof. Dr. Misra in may 2019 to enquire about an internship position abroad that could be followed up by a master's project at the SRL. He introduced me to the Science and Technologies Of Robotics in Medicine (STORM) Lab at the University of Leeds, the United Kingdom, to work on a project of four months in the field of magnetic actuation. In addition, he allowed me to continue along that line of research in the SRL. Dr. Venkatasubramanian Kalpathy Venkiteswaran (Venkat) has been my daily supervisor throughout the past ten months and I want to especially thank him for his guidance and support. In addition he has had a large hand in getting our paper ready in time for submission to the IEEE International Conference on Robotics and Automation (ICRA 2021) and Robotics and Automation Letters (RA-L), which at the time of writing is under review.

I want to express a special thanks to Dr. Ir Arvid Quintijn Léon Keemink for being the external member of my graduation committee. In addition, I want to thank Dr. Jakub Sikorski who was willing to share his measurements of the magnetic field generated in the system used during this project. Furthermore I want to thank my fellow master students at the SRL, who have graduated and are still looking forward to graduating at the time I am writing this, with whom I have enjoyed getting coffee every day. A general thanks to all other members of the SRL for being an amazing group of people always willing to help each other.

Finally, a warm thanks to my girlfriend and parents for all the support throughout this project, also during the most stressful periods.

Michiel Richter
November 12, 2020

Abstract

In the past decade, remote actuation through magnetic fields has been used for position and orientation control of continuum manipulators (CMs). Complex manipulator shapes can be achieved by leveraging multiple points of actuation along the length of the CM. In this thesis we present an approach for multi-point orientation control of discretely magnetized CMs based on a quasi-static forward model. The manipulator contains two permanent magnets, which are each actuated by unique and independent magnetic fields generated by an array of mobile electromagnets. We formulate an accurate model to represent the magnetic field and field gradient of an azimuthally symmetric electromagnet based on a multipole expansion of a Cartesian scalar potential. This description is applied to predict exerted wrenches on the continuum manipulator. Cosserat rod theory is used to model the manipulator deformation under external wrenches. A numerically-computed Jacobian is developed to calculate the actuation inputs. A stereo vision setup is used to reconstruct the shape of the CM. During experiments, manual steps in target orientations are given to show independent orientation control of the two permanent magnets in 2D. Multi-point orientation control in 3D is demonstrated through a virtual clone of an electromagnetic system with movable electromagnets. In both scenarios, it is observed that the algorithm is able to successfully steer two magnets in a non-uniform magnetic field.

Table of Contents

List of Figures

List of Tables

1 Background

MAGNETS have found their application in medical research as early as the 1950s when catheterization was performed on corpses and living dogs using a large magnet positioned outside of the body to actuate an elastic steel wire spanning the lumen of the catheter [1]. Of course the techniques at the time were rudimentary and, over the years, computing power and improved mathematical models have fueled the progression of magnetism for medical applications. Magnetism and its applications are an exciting field of study because through magnetism we can exert forces on objects without being in physical contact, even through barriers such as skin. One can thus imagine that, by employing magnets to actuate e.g. a magnetized catheter, the catheter can be stripped of any conventional on-board equipment designed for actuation, making it possible to miniaturize the catheter and potentially reach smaller regions inside the human body. However, to apply magnetic actuation it is important to know what the properties of magnets are, and how they interact with each other. This problem is covered in the field of magnetism, which can be traced back to ancient Greece.

Magnets are believed to have gotten their name from the ancient Greek city of Magnesia, where lodestones (naturally occurring magnets) were found [2]. These naturally occurring magnets are now known as permanent magnets; magnets which can neither be turned non or off. It was only much later, in the early nineteenth century, that the Danish physicist Hans Christian Oersted found that electricity and magnetism were related, when, during his experiments, a compass needle would change direction as a current passed through a wire beside it (Fig. 1). Most notably he found that when he suspended a compass above a current-carrying wire, running a current from north to south, the compass needle deflected west; when suspended west, deflected south; when suspended south, deflected east; and when suspended east, deflected north [3]. Evidently a rotational force was exerted on the compass needle due to the passing current, but in different directions based on its position relative to the current. From the description of the experiments conducted by Oersted, the compass needle moved counter-clockwise. These results were later in the nineteenth century explained by the English scientist Michael Faraday, who envisioned matter to be a source of action, and space to be a conductor. From this idea of matter and space he envisioned magnetized matter (such as the lodestones in Magnesia) to have a large number of directional conducting lines within its surrounding space, conducting a force to surrounding magnetic matter. He appropriately called these lines the magnetic lines of force [4]. We now know that any magnet has a north- and south-pole, where the north-pole of a freely suspended magnet (such as a compass needle) would point towards the geographic north, as used by pathfinders today [2]. Also, the modern description of the lines of force are the magnetic field lines.

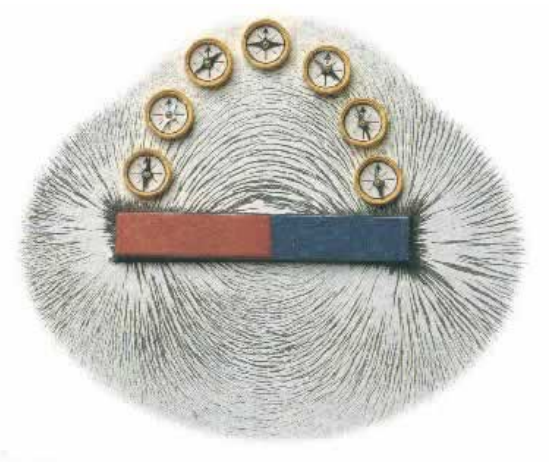

Figure 1: A bar magnet is a permanent magnet with a north-pole (red) and south-pole (blue). The bar magnet is surrounded by iron filings. Iron filing is ferromagnetic, which means it is magnetized inside a magnetic field, which is generated by the bar magnet. The iron filings line up with the magnetic lines of force to visualize the otherwise invisible magnetic field. The magnetic lines of force run from the north- to south-pole. The iron filings show the pattern but not the direction of the lines of force. A compass needle will align itself with the direction of the lines of force, running from the north- to south-pole [2].

1.1 Magnetic Field

The magnetic field lines are a visual description of the magnetic field surrounding a magnet. This is easy to visualize for a permanent magnet with a distinct north- and south-pole, but what about a magnetic field induced by a current? A magnetic field forms around a current carrying wire as shown in Fig. 2. A magnetic field is typically given the symbol $\boldsymbol{B}$ and current the symbol I. Each magnetic field line forms a circular path around the wire with varying field strength. The direction is determined by the right-hand rule; by pointing your right thumb along the direction of current, the remaining fingers curl in the direction of the magnetic field. The stronger the current, the stronger the field will be, explaining why in Fig. 2 the iron filings on the outskirts are more disorganized than those close to the wire. Another

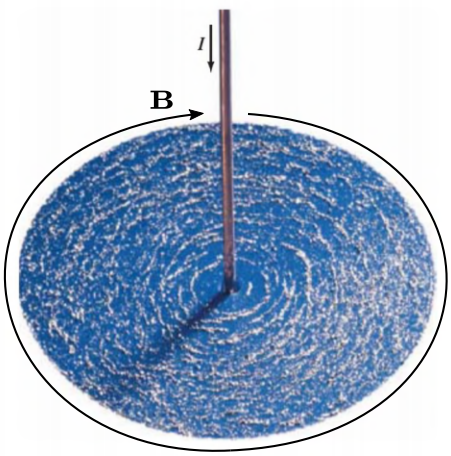

Figure 2: A current (I), also moving electrons, running through a conductive wire will generate a circular magnetic field about the direction of current. The magnetic field is weaker further away from the wire. The direction of the magnetic field is determined from the right hand rule [2].

method to increase the strength of the field is to simply add another source of a magnetic field.

Magnetic fields follow the principle of superposition. This means that the magnetic field at a point in space, due to more than one magnetic field source, is found by adding the magnetic fields from each source [5]. An important difference between the magnetic field generated by the bar magnet and line current in Figs. 1 and 2 is that the bar magnet has a distinct north- and south-pole, while the line current does not. However, a current-carrying wire can be looped such that a north- and south-pole is created from the principle of superposition. Such a loop of current-carrying wire is called a coil (Fig. 3).

A coil consists of a conductive wire typically looped into a cylindrical structure, i.e. coiled up. Increasing the number of coils also increases the generated magnetic field and is called a solenoid. By sending a current through a solenoid, each loop produces a magnetic field. Toward the center of the solenoid the fields are in the same direction and form a stronger field. The field between the wires of each loop in a solenoid are opposite and cancel. Furthermore, the magnetic field inside the solenoid is large relative to outside. The magnetic field lines of a solenoid are either directed out of the top or the bottom, depending on the direction of the current, and their direction depicts the magnetic north-pole of the solenoid. Note that, because the direction of current can be reversed, so can the north- and south-pole. In general, the more windings a solenoid has per unit length, and the higher the current is, the stronger the magnetic field will be [2]. Theoretically the wire thickness can be very small, but in reality the solenoid would overheat, melting the conductive wire, and form a single block of conductive material.

The magnetic field generated by a coil, solenoid, electromagnet, or permanent magnet has the property of being uniform or non-uniform. A uniform magnetic field has the field lines pointing in the same direction everywhere, with equal field strength; a non-uniform magnetic field differs in magnitude and direction. In case of a solenoid, the magnetic field lines depict an ellipsoid and are fairly uniform at the center of the solenoid, but change direction and decrease in strength with distance outside of the solenoid (Fig. 3).

At the start of the previous paragraph we mentioned electromagnets. An electromagnet is a type of solenoid which contains, generally, a metallic core. This metallic core is made of a material that becomes magnetic when subject to a magnetic field. The strongest of these minerals are called ferromagnetic materials (such as iron). The underlying principle comes from the atomic model of Niels Bohr, which

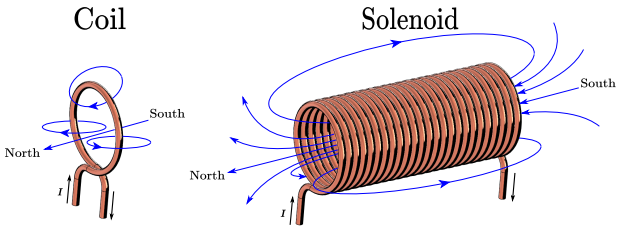

Figure 3: A current-carrying conductive wire shaped into a loop is called a coil, which mimics the characteristic north- and south-pole of a bar magnet (Fig. 1) as a result of adding the magnetic field from each infinitesimal length of the conductive wire. Many coils in succession is called a solenoid, which produces a strong field inside its volume from the superposition principle of magnetic fields.

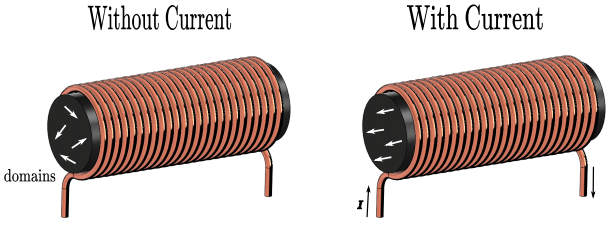

Figure 4: A ferromagnetic core can be placed within the coils of a solenoid. A ferromagnetic core is made up of a material that consists of small magnetic domains which align in the presence of an external magnetic field, increasing the strength of the magnetic field outside the boundaries of the coils. This strong magnet is called an electromagnet.

describes an element as having a dense nucleus, with protons and neutrons, surrounded by orbiting electrons. While the electrons have an orbital angular momentum (they fly around the nucleus), they also have their own spin (electron spin). Since electrons carry charge, and are in continuous motion, they behave like small magnets, producing their own magnetic field. Depending on the direction of spin, the north-pole of the electron can point in two directions. The orbiting electrons move in electron shells that envelop the nucleus. Each shell can hold an even number of electrons. Electrons exist in pairs with opposite spin within these shells, such that the magnetic fields are cancelled. Some atoms have an uneven number of electrons, meaning that the atom only has partially filled shells, and thus has a net magnetic field, or dipole moment. While materials, consisting of many atoms may have an uneven number of electrons, they are not magnetic by default. This is because the dipole moments of free electrons, for all atoms within a material, may still be in different directions, still cancelling each other. In ferromagnetic materials, the dipoles tend to point in the same direction. This occurs in smaller groups of atoms, called domains, that are randomly oriented, meaning that the material as a whole may not be magnetized (Fig. 4). However, under influence of an external magnetic field, these domains become aligned and the material becomes magnetized. When the magnetic field is removed, the domains gradually return to their prior states. However, if the externally applied field is strong enough, the material is said to be saturated, and forms a permanent magnet where the majority of the domains remain aligned even after removing the applied field [5]. By adding a ferromagnetic material to the center of a solenoid to produce an electromagnet the generated field strength is increased.

In research on magnetics for medical applications, very often an iron (or alloy of iron) core is used, because less current is required to achieve a desired magnetic field. An example of an actuation system, that is used in the experiments presented in this thesis, is BigMag, shown in Fig. 5. BigMag contains six electromagnets, each attached on a side of a hexagonal fixture. The hexagon is divided into a top and bottom half, each half containing two soft iron-cored electromagnets and one air-cored solenoid. The iron-cored electromagnets serve to produce a strong magnetic field, while the air-cored solenoids are used to increase the bandwidth of the field generation along the vertical axis [6]. An increased bandwidth means that the fields can be generated more quickly. Iron-cored electromagnets generally have a lower bandwidth when it comes to generating magnetic fields, because alignment of domains does not happen instantly.

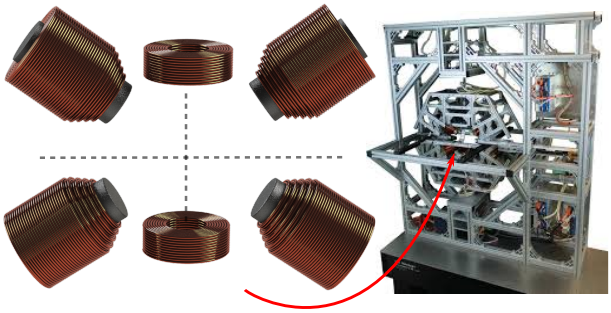

Figure 5: A magnetic actuation system contains a set of solenoids and electromagnets. These systems are used to generate desired magnetic fields at any point within their actuation workspace in order to actuate other magnetized objects [6].

The objecive of a magnetic actuation system is to control magnetic elements within the actuation workspace. For accurate actuation, it is important to know how the currents through the electromagnets of the system, I, relate to their produced magnetic fields ($\boldsymbol{B}(\boldsymbol{p}, I)$) at some position $\boldsymbol{p}$ within that workspace. To obtain a suitable description of the field, we should first know the mathematical rules that the magnetic field must conform to. These rules are called Maxwell's equations, which are presented at the end of the following section.

1.2 Magnetic Field Constraints

The equations formulated by James Clark Maxwell describe the spatial properties that a magnetic field ($\boldsymbol{B}(\boldsymbol{p}, I)$) conforms to. More specifically, how the field changes in space is determined by the spatial gradients of the magnetic field, and Maxwell's equations prescribe constraints to these spatial gradients.

In an effort to visualize the field constraints before diving into mathematical notations, look at Fig. 6. A current ($\boldsymbol{I} = I\hat{\boldsymbol{z}}$) runs perpendicular to the plane of the page and a magnetic field ($\boldsymbol{B}$) makes a circular path in the counter-clockwise direction about $\boldsymbol{I}$. The magnetic field lines are represented as arrows; the direction of the arrows show the direction of the field; the lengths of the arrows represent the field strength. For the 2D magnetic field in Fig. 6 ($\boldsymbol{B} = [B_x, B_y, 0]^T$) the change of the magnetic field in space is represented by its partial derivatives with respect to Cartesian coordinates (magnetic field gradients), i.e.

$$\frac{\partial B_x}{\partial x}, \quad \frac{\partial B_x}{\partial y}, \quad \frac{\partial B_y}{\partial y}, \quad \frac{\partial B_y}{\partial x}$$

As was mentioned before, Maxwell's equations set

constraints onto these magnetic field gradients and these are discussed below.

1.2.1 Divergence

The divergence of the magnetic field is a property that describes the outward spreading of the field at a point $\boldsymbol{p}$. Any point in space $\boldsymbol{p}$ can be a source, sink, or neither, of a vector field ($\boldsymbol{B}(\boldsymbol{p}, I)$). A source would be a point $\boldsymbol{p}$, represented as an infinitesimal volume, where the number of magnetic field lines leaving the volume is greater than the number that enters, a sink would be a volume where the number of magnetic field lines entering the volume is greater than leaving, and the volume would be neither a source or sink if the number of entering and leaving magnetic field lines is equal. A magnetic field generated by a magnetic-dipole is divergence-free (that is it is neither a source or sink) because the field is circular, meaning that the number of field lines that enter an infinitesimal volume in space is equal to those leaving (shaded region in fig. 6). This is mathematically represented as

$$\nabla \cdot \boldsymbol{B}(\boldsymbol{p}, I) = \frac{\partial B_x}{\partial x} + \frac{\partial B_y}{\partial y} + \frac{\partial B_z}{\partial z} = 0,$$

where ∇ is the gradient operator. If we look at the position $\boldsymbol{p}_3$, which has a field $\boldsymbol{B}(\boldsymbol{p}_3, I)$, we notice from the change in direction of arrows that

$$\boldsymbol{B}(\boldsymbol{p}_3, I): \quad \frac{\partial B_x}{\partial x} < 0, \quad \frac{\partial B_y}{\partial y} > 0, \quad \frac{\partial B_z}{\partial z} = 0,$$

and the divergence-free property says that how B_x changes along x is equal and opposite to how B_y changes along y, which is intuitive for a circular magnetic field.

1.2.2 Curl

The curl of the magnetic field is a property that describes how the magnetic field curls around a point ($\boldsymbol{p}$) in space. In Fig. 6 the field clearly only curls about the point $\boldsymbol{p}_0$, which lies on the current vector $\boldsymbol{I}$. Thus, a magnetic field only curls about the source of the field. This is mathematically represented as

$$\nabla \times \boldsymbol{B}(\boldsymbol{p}, I) = \begin{bmatrix} \frac{\partial B_z}{\partial y} - \frac{\partial B_y}{\partial z} \\ \frac{\partial B_x}{\partial z} - \frac{\partial B_z}{\partial x} \\ \frac{\partial B_y}{\partial x} - \frac{\partial B_x}{\partial y} \end{bmatrix} = \mu_0 \boldsymbol{J}(\boldsymbol{p}) + \mu_0 \epsilon_0 \frac{\partial \boldsymbol{E}(\boldsymbol{p})}{\partial t},$$

where $\boldsymbol{J}(\boldsymbol{p})$ is the current-density, defined as the directional current $\boldsymbol{I}$ divided by the cross-sectional area of the current-carrying wire, and $\boldsymbol{E}(\boldsymbol{p})$ an electric field. From this definition, clearly at $\boldsymbol{p}_0$ the curl is non-zero.

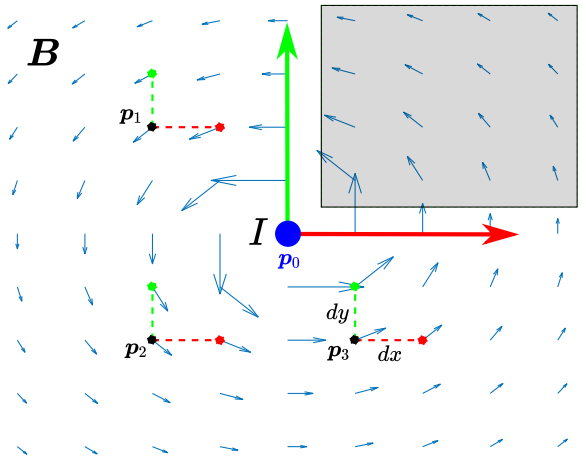

Figure 6: The magnetic field ($\boldsymbol{B}$) strength and direction is represented by the length and direction of arrows, respectively. A magnetic field always curls about a region of non-zero current ($\boldsymbol{I}$) and does under no circumstance radiate outwards as is the case with electric fields. The latter case is illustrated by the magnetic field lines entering a shaded infinitesimal volume; the magnetic field lines entering the volume is equal to those leaving, depicting that the volume is neither a source nor a sink of the magnetic vector field.

However, if we look at position $\boldsymbol{p}_3$, cleary $\boldsymbol{J}(\boldsymbol{p}_3) = \boldsymbol{0}$ and for a steady current $\dot{\boldsymbol{E}}(\boldsymbol{p}_3) = \boldsymbol{0}$, then

$$\nabla \times \boldsymbol{B}(\boldsymbol{p}_3, I) = \boldsymbol{0},$$

which we can notice from the change in directions of arrows

$$\boldsymbol{B}(\boldsymbol{p}_3, I): \quad \frac{\partial B_z}{\partial y} = 0, \quad \frac{\partial B_y}{\partial z} = 0, \quad \frac{\partial B_x}{\partial z} = 0, \quad \frac{\partial B_z}{\partial x} = 0, \quad \frac{\partial B_y}{\partial x} < 0, \quad \frac{\partial B_x}{\partial y} < 0.$$

1.2.3 Maxwell's Equations

From the discussions of the divergence and curl of the magnetic field, Maxwell's equations are formulated as

$$\nabla \cdot \boldsymbol{B}(\boldsymbol{p}, I) = 0,$$
$$\nabla \times \boldsymbol{B}(\boldsymbol{p}, I) = \mu_0 \boldsymbol{J}(\boldsymbol{p}) + \mu_0 \epsilon_0 \frac{\partial \boldsymbol{E}(\boldsymbol{p})}{\partial t}.$$

The first expression constrains the magnetic field lines to be purely curling and therefore do not spread out; this is called the divergence-free property of the field. The second expression constrains the magnetic field

lines to only curl about the field source. For any point outside a permanent magnet or current-carrying wire, the curl is therefore zero, and Maxwell's equations can be simplified to

$$\begin{aligned} \nabla \cdot \boldsymbol{B}(\boldsymbol{p}, I) &= 0, \\ \nabla \times \boldsymbol{B}(\boldsymbol{p}, I) &= \boldsymbol{0}, \end{aligned} \tag{1}$$

which are called Maxwell's equations for a current-free workspace, known as the divergence-free and curl-free property of the magnetic field.

The divergence-free and curl-free properties limit the number of independent spatial gradients of the field. The spatial gradients can be grouped into a second-order gradient tensor ($\boldsymbol{B}_\nabla(\boldsymbol{p}, I) \in \mathbb{R}^{3\times 3}$). The divergence-free property puts a constraint on the trace of $\boldsymbol{B}_\nabla$, and the curl-free property constrains $\boldsymbol{B}_\nabla$ to be symmetric,

$$\boldsymbol{B}_\nabla(\boldsymbol{p}, I) = \begin{bmatrix} \frac{\partial B_x}{\partial x} & \frac{\partial B_x}{\partial y} & \frac{\partial B_x}{\partial z} \\ \frac{\partial B_x}{\partial y} & \frac{\partial B_y}{\partial y} & \frac{\partial B_y}{\partial z} \\ \frac{\partial B_x}{\partial z} & \frac{\partial B_y}{\partial z} & -\left(\frac{\partial B_x}{\partial x} + \frac{\partial B_y}{\partial y}\right) \end{bmatrix}.$$

The magnetic field ($\boldsymbol{B}$) and the spatial gradients of the field ($\boldsymbol{B}_\nabla$) have a total of eight independent terms. Therefore magnetic actuation systems such as BigMag (Fig. 5) are designed to have eight degrees of freedom to have full control of the field and field gradients at any position ($\boldsymbol{p}$) within the actuation workspace [6–8].

Up to now we have used the notation $\boldsymbol{B}(\boldsymbol{p}, I)$ to denote the magnetic field. However we have not yet discussed any mathematical model of the field that conforms to the equations of Maxwell for the magnetic field in a current-free workspace. Such a mathematical model uniquely follows from the magnetic scalar potential.

1.3 Magnetic Scalar Potential

A magnetic scalar potential, equivalently a scalar harmonic function, is a scalar function ($\mathcal{V}(\boldsymbol{p})$) that satisfies the Laplace equation. The curl-free property of the magnetic field allows expressing the magnetic field as the gradient of a scalar harmonic function,

$$\nabla \times \boldsymbol{B}(\boldsymbol{p}, I) = \nabla \times (\nabla \mathcal{V}(\boldsymbol{p}) \cdot I) = \boldsymbol{0},$$

because the curl of a gradient is inherently zero. Thus, when expressing the magnetic field as the gradient of a scalar function, the divergence-free property remains, that is,

$$\nabla \cdot \boldsymbol{B}(\boldsymbol{p}, I) = \nabla \cdot (\nabla \mathcal{V}(\boldsymbol{p}) \cdot I) = \nabla^2 \mathcal{V}(\boldsymbol{p}) \cdot I = 0,$$

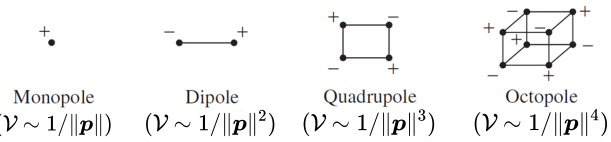

Figure 7: A magnetic field is uniquely represented by a scalar function, called a scalar potential ($\mathcal{V}(\boldsymbol{p})$), as a function of a point ($\boldsymbol{p}$) at which the field is measured. The scalar potential is formulated in electrodynamics textbooks as a multipole expansion. A multipole expansion is a sum of terms that consider a magnet as a monopole, dipole, quadrupole, octopole, etc. A dipole consists of two monopoles, a quadrupole of two dipoles, an octopole of two quadrupoles, etc. Monopoles are generally excluded because they have not yet been found in nature. Each higher term makes the scalar potential ($\mathcal{V}$), and thus the magnetic field ($\boldsymbol{B}$), decrease with higher orders of distance to the field point ($\|\boldsymbol{p}\|$) [5].

which is not inherently satisfied. Therefore a scalar function must be chosen for which

$$\frac{\partial^2 \mathcal{V}(\boldsymbol{p})}{\partial x^2} + \frac{\partial^2 \mathcal{V}(\boldsymbol{p})}{\partial y^2} + \frac{\partial^2 \mathcal{V}(\boldsymbol{p})}{\partial z^2} = 0,$$

which is the Laplace equation. It turns out that the solution for $\mathcal{V}(\boldsymbol{p})$ is unique, and can be found in electrodynamics textbooks [5]. The solution for $\mathcal{V}(\boldsymbol{p})$ is beyond the scope of this introduction, but can be found discused in Appendix C, and is represented as a multipole expansion.

A multipole expansion is a mathematical series that provides an exact description of the magnetic potential $\mathcal{V}(\boldsymbol{p})$ as long as the magnetic field source is localized close to the origin and the potential is observed away from the origin, or when the source is localized far away from the origin and the potential is observed close to the origin. The multipole expansion is a sum of a monopole, dipole, quadrupole, etc, terms. Generally the monopole term is excluded because of the absence in nature. A magnetic dipole is made up of two monopoles, a quadrupole of two dipoles, and an octopole of two quadrupoles, etc (plus and minus sign depict north- and south-poles, respectively) as shown in Fig. 7. Each consecutive term decreases with increasing powers of distance ($\|\boldsymbol{p}\|$) from the field source. Therefore, close to the field source there is greater influence of the higher pole terms, whereas far enough away from the field source the dipole term is dominant. For magnetic fields far away the magnetic field is therefore approximated with just the dipole term, giving the first-order point dipole approximation of the magnetic field.

1.4 Magnetic Point Dipole Approximation

The magnetic point dipole model is a first-order approximation that considers the magnet, viewed from a point ($\boldsymbol{p}$) far enough away from the magnet, as a magnetic dipole and disregards any higher order poles in the multipole expansion (Fig. 7). The point dipole model is formulated as

$$\boldsymbol{B}_{dip}(\boldsymbol{p}, I) = \frac{\mu_0}{4\pi\|\boldsymbol{p}\|^5}\left(3\boldsymbol{p}\boldsymbol{p}^T - \|\boldsymbol{p}\|^2\boldsymbol{I}_3\right)\boldsymbol{\mu}(I),$$

where $\boldsymbol{I}_3 \in \mathbb{R}^{3\times3}$ is a unit matrix and $\boldsymbol{\mu}(I) \in \mathbb{R}^3$ is called the magnetic dipole moment.

The magnetic dipole moment is a measure of the strength of the magnet. It has both a magnitude and direction, where the direction depicts the north-pole of the magnet. For solenoids (air-cored electromagnets) and ferromagnetic materials the magnetic dipole moment can be calculated as

$$\begin{aligned} \text{Solenoid:} \quad & \boldsymbol{\mu} = NIA\hat{\boldsymbol{a}} \\ \text{Ferro-/Permanent Magnet:} \quad & \boldsymbol{\mu} = \frac{V \cdot B_r}{\mu_0}\hat{\boldsymbol{n}}, \end{aligned}$$

where N is the number of windings, A is the area enclosed by the windings, $\hat{\boldsymbol{a}}$ a unit-vector perpendicular to the enclosed area, V the magnet volume, B_r is the residual flux density, and $\hat{\boldsymbol{n}}$ a unit-vector representing the north-pole direction.

Models, such as the point dipole model, of the magnetic field are used to compute the torques and forces that are exerted onto magnetic agents positioned within the field. This is the principle of magnetic actuation.

1.5 Magnetic Actuation

If a magnetic dipole with dipole moment $\boldsymbol{\mu} \in \mathbb{R}$ is placed at a position $\boldsymbol{p}$, it will experience a torque ($\boldsymbol{\tau} \in \mathbb{R}^3$) and force $\boldsymbol{f} \in \mathbb{R}^3$ due to the magnetic field ($\boldsymbol{B}(\boldsymbol{p}, I)$) at that position. The torque and force are experienced in directions to minimize the magnetic energy of the dipole [9]. If we again visualize the dipole as having a north- and south-pole, and the magnetic field represented by its field lines, the north-pole of the dipole will rotate to align with the direction of the field lines and the magnet displaced in the direction that minimizes its energy (Fig. 8),

$$\begin{aligned} \boldsymbol{\tau} &= \boldsymbol{\mu} \times \boldsymbol{B}(\boldsymbol{p}, I) \\ \boldsymbol{f} &= \nabla\boldsymbol{B}(\boldsymbol{p}, I)^T\boldsymbol{\mu}. \end{aligned}$$

Clearly, the torque ($\boldsymbol{\tau}$) is greater in a stronger field and with higher angular distance between the dipole moment and the field direction, and the force ($\boldsymbol{f}$) is greater in a region of higher field gradient. The above expressions for magnetic torque and force can be rewritten to

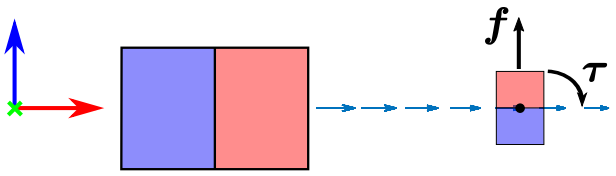

Figure 8: A magnet placed inside an external magnetic field will experience a torque ($\boldsymbol{\tau}$) and force ($\boldsymbol{f}$) in directions to minimize its magnetic energy [9].

$$\boldsymbol{\tau} = \boldsymbol{\mu} \times \boldsymbol{B} \quad = [\boldsymbol{\mu}]_\times \boldsymbol{B}, \tag{2}$$

$$\boldsymbol{f} = \nabla\left(\boldsymbol{\mu} \cdot \boldsymbol{B}\right) = \mathbb{M}(\boldsymbol{\mu})\mathbb{G}(\boldsymbol{B}_\nabla), \tag{3}$$

with the following definitions

$$[\boldsymbol{\mu}]_\times = \begin{bmatrix} 0 & -\mu_z & \mu_y \\ \mu_z & 0 & -\mu_x \\ -\mu_y & \mu_x & 0 \end{bmatrix}, \quad \boldsymbol{B} = \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix},$$

$$\mathbb{M}(\boldsymbol{\mu}) = \begin{bmatrix} \mu_x & \mu_y & \mu_z & 0 & 0 \\ 0 & \mu_x & 0 & \mu_y & \mu_z \\ -\mu_z & 0 & \mu_x & -\mu_z & \mu_y \end{bmatrix},$$

$$\mathbb{G}\left(\boldsymbol{B}_\nabla\right) = \begin{bmatrix} \partial_x B_x, & \partial_y B_x, & \partial_z B_x, & \partial_y B_y, & \partial_z B_y \end{bmatrix}^T.$$

The map $\mathbb{G}(\boldsymbol{B}_\nabla) : \mathbb{R}^{3\times3} \to \mathbb{R}^5$ maps the five independent spatial magnetic field gradient terms to a vector.

The magnetic torques and forces are used to steer magnetized objects, called magnetic actuation. A magnet can be attached to a flexible body (continuum manipulator) such that exerted torques and forces result in a deformation of that body. This the topic of the following thesis.

2 Introduction

Note: *An introductory background to magnetism and magnetic actuation was given in the previous section. In this section I present the research paper that was submitted to the IEEE International Conference on Robotics and Automation (ICRA 2021) and Robotics and Automation Letters (RA-L), and at the time of writing is under review. It starts with an introduction to motivate the title '**Multi-Point Orientation Control of Discretely Magnetized Continuum Manipulators**'. This is followed by theories on modelling the continuum manipulator and magnetic field. Then the results of experiments and simulations are provided. Finally, the results are discussed and a conclusion is given. An elaborate discussion on the applied theories is provided in the appendix.*

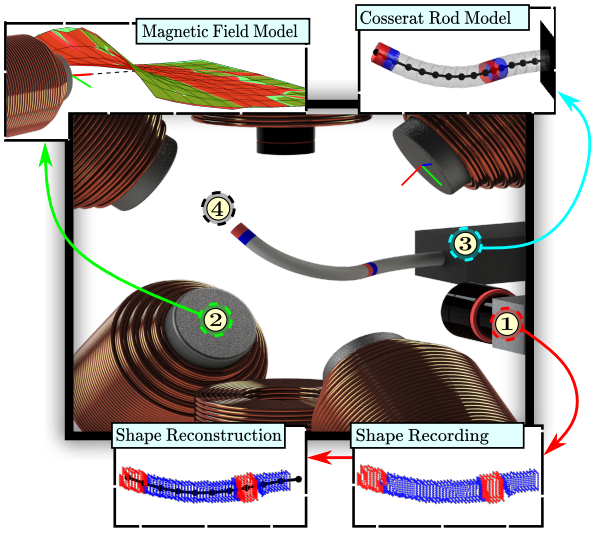

Figure 9: In this paper we use BigMag, an array of 6 mobile electromagnetic coils, to perform closed-loop orientation control of two permanent magnets carried by a continuum manipulator (CM) attached to a rigid base. (1) A stereo vision setup is used for shape reconstruction of the CM. (2) The field at the magnet positions is predicted from fitted magnetic field (and gradient) models for each electromagnet in BigMag, to predict exerted magnetic wrenches on the CM. (3) Cosserat rod theory is used to formulate a quasi-static forward model of the CM. (4) This model is used to predict changes in magnet orientations, due to changes in the currents and positions of the electromagnets, to actuate the CM within the workspace of BigMag.

Continuum manipulators (CMs) are continuously deformable elastic structures. Compared to traditional rigid link robots, CMs have infinite degrees of freedom (DOFs) which allows them to deform at any point along their length. This property makes CMs dexterous, enabling them to navigate through constricted spaces and cluttered environments [10]. The challenge lies in steering and navigation of CMs, as actuation at multiple points along the CM also requires actuators at those points. However, actuators attached to the body of the CM increase its size, which is disadvantageous in confined workspaces [11]. Magnetic actuation provides the potential for miniaturization by moving the actuation system outside the confined environment [12].

Magnetized CMs can be discretely or continuously magnetized. Discretely magnetized CMs contain rigid magnetic, and flexible non-magnetic material. The rigid magnetic material can be divided into actively magnetized electromagnets [13], or permanent magnets [14]. Alternatively, continuously magnetized CMs consist of magnetic composite polymers [15, 16]. The magnetized CMs deform due to magnetic forces and torques exerted on their magnetic regions by a magnetic field. Independent control of magnetic forces and torques, on a single magnet in space, requires eight magnetic degrees of freedom [17, 18].

Magnetic actuation systems have been designed to satisfy the requirement of eight DOFs. Such systems employ stationary electromagnets [8], mobile electromagnets [6, 19], or mobile permanent magnets [14], as sources of the magnetic field. The mobile actuation systems have the advantage of an open and typically larger work space than stationary systems, while requiring smaller external magnets [20]. Precise magnetic field information from each field source is required for actuation. Previously used approaches to model the magnetic field have been to formulate an arbitrary function and fit unknown coefficients with least squares optimization [6, 21], or use a first-order dipole approximation [14]. The former approach does not typically enforce constraints on the spatial gradients of the magnetic field, and the latter ignores higher order field effects that are more prominent closer to the field source [22]. A recently used approach, that solves the aforementioned disadvantages, is to express the magnetic field as the gradient of a scalar potential (scalar harmonic function), and use measurement-informed least squares optimization to fit unknown coefficients [23].

A mathematical model of the manipulator is used to compute its deformation under exerted magnetic torques and forces, which result from the interaction with an external magnetic field. These models generally have a trade-off between accuracy and computational efficiency. Computationally cheap rigid-link models divide the CM into rigid segments, which has

the advantage of providing analytical solutions for control [11, 24, 25], but require many segments to describe complex CM geometries. In our work, Cosserat rod theory is used to model the CM. Cosserat rod theory gives an exact solution to the statics of the CM, while additionally accommodating variations in the stiffness, body loading, and cross-section geometry of a CM [26, 27].

Rigid-link and Cosserat rod models have recently been used for position control of a CM with a single magnet at its distal tip [24, 27]. In addition, unidirectional steering within a uniform magnetic field has been shown with permanent magnets integrated into the distal tip of a CM [28, 29]. Through the actuation of a CM at a single point on its distal tip, it is only possible to achieve follow-the-leader motion of the proximal part of the body. By introducing multiple actuation points along the length of the manipulator, it is possible to achieve active shape control of the manipulator body between actuation points, which could be of particular interest in applications involving contact with the workspace. For a magnetized CM, multi-point actuation can be achieved through independently generating different magnetic fields at multiple points within the workspace.

In this paper we show closed-loop orientation control of two magnets along a discretely magnetized continuum manipulator. BigMag, an array of 6 mobile electromagnetic coils, is used as a magnetic actuation system for the purpose of experimental validation [6]. We formulate a linear least squares approach to measurement-based fitting of a magnetic field and spatial magnetic field gradient map based on a Cartesian multipole expansion of a scalar potential. The fitting procedure is applied to electromagnets in BigMag. A stereo vision setup records the workspace and is used for shape reconstruction of the CM. A forward model of the CM is constructed with Cosserat rod theory, and used to predict changes in the shape of the CM due to variations in the external magnetic field. Experiments are performed within BigMag. In addition, BigMag is extended with eight additional electromagnets in simulation to show that the control algorithm is able to achieve complex geometries of the CM. All the above-mentioned steps are detailed in the following sections, followed by discussion and conclusions.

3 Materials and Methods

3.1 Theory

In this section, the model used for analyis of the continuum manipulator is described, followed by the formulation of a cartesian multipole expansion of a scalar potential for accurate, measurement-informed, field and field gradient fitting. In this section, the model used for analyis of the continuum manipulator is described, followed by the formulation of a Cartesian multipole expansion of a scalar potential for accurate, measurement-informed fitting of magnetic field and magnetic field gradient map.

3.1.1 Cosserat Rod Model

In Cosserat rod theory the body of the manipulator is characterized by centerline parameter $s \in [0, L]$, where $L \in \mathbb{R}^+$ represents the manipulator length. In addition, a set of material state parameters $\boldsymbol{y}(s)$, relative to a global reference frame, are assigned to cross-sections along the centerline of the manipulator,

$$\boldsymbol{y}(s) = \begin{bmatrix} \boldsymbol{p}(s)^T & \boldsymbol{q}(s)^T & \boldsymbol{n}(s)^T & \boldsymbol{m}(s)^T \end{bmatrix}^T \in \mathbb{R}^{13}, \tag{4}$$

where $\boldsymbol{p} \in \mathbb{R}^3$ denotes the Cartesian position, $\boldsymbol{q} = (q_r, \boldsymbol{q}_i) \in \mathbb{H}$ the orientation quaternion (where $q_r \in \mathbb{R}$ and $\boldsymbol{q}_i \in \mathbb{R}^3$ are the real and imaginary components, respectively [30]), $\boldsymbol{n} \in \mathbb{R}^3$ the internal force, and $\boldsymbol{m} \in \mathbb{R}^3$ the internal bending [31]. The state parameters evolve along s according to the Cosserat rod ordinary differential equations (ODEs) defined as

$$\begin{aligned} \boldsymbol{p}' &= \boldsymbol{R}(\boldsymbol{q})\boldsymbol{v}, \\ \boldsymbol{v} &= \boldsymbol{K}_s^{-1}\boldsymbol{R}(\boldsymbol{q})^T\boldsymbol{n} + \hat{\boldsymbol{v}}, \\ \boldsymbol{q}' &= \frac{1}{2}\begin{bmatrix} -\boldsymbol{q}_i^T \\ q_r\boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix}\boldsymbol{R}(\boldsymbol{q})\boldsymbol{u}, \\ \boldsymbol{u} &= \boldsymbol{K}_b^{-1}\boldsymbol{R}(\boldsymbol{q})^T\boldsymbol{m} + \hat{\boldsymbol{u}}, \\ \boldsymbol{n}' &= -\boldsymbol{f}, \\ \boldsymbol{m}' &= -\boldsymbol{p}' \times \boldsymbol{n} - \boldsymbol{\tau}, \end{aligned} \tag{5}$$

where the explicit dependence on s is omitted for brevity, $\boldsymbol{I}_3 \in \mathbb{R}^{3\times3}$ is an identity matrix, and $[\cdot]_\times : \mathbb{R}^3 \to \mathbb{R}^{3\times3}$ represents a mapping to a skew-symmetric matrix [27]. We use the notation $\partial_s\boldsymbol{p} \equiv \boldsymbol{p}'$ to denote the partial derivative of the state parameter with respect to s. In addition, $\boldsymbol{R}(\boldsymbol{q}) \in SO(3)$ is the rotation matrix representation of orientation quaternion $\boldsymbol{q}$, given by

$$\boldsymbol{R}(\boldsymbol{q}) = q_r^2\boldsymbol{I}_3 + \boldsymbol{q}_i\boldsymbol{q}_i^T + 2q_r[\boldsymbol{q}_i]_\times + [\boldsymbol{q}_i]_\times^2. \tag{6}$$

The set of thirteen coupled nonlinear ODEs in Eqn. (5) relate external forces $\boldsymbol{f}(s) \in \mathbb{R}^3$ and torques $\boldsymbol{\tau}(s) \in \mathbb{R}^3$ to changes in internal forces and moments. The internal forces and torques determine the material strain $\boldsymbol{v}(s) \in \mathbb{R}^3$ and bending $\boldsymbol{u}(s) \in \mathbb{R}^3$, which

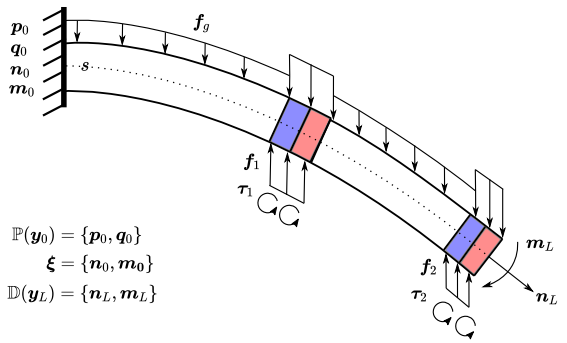

Figure 10: A discretely magnetized continuum manipulator subject to a distributed gravitational force ($\boldsymbol{f}_g$) and distributed magnetic torques ($\boldsymbol{\tau}_m$) and forces ($\boldsymbol{f}_m$) at two magnets, where m denotes the magnet index. The manipulator shape is described by position ($\boldsymbol{p}(s)$), orientation quaternion ($\boldsymbol{q}(s)$), internal force ($\boldsymbol{n}(s)$), and internal moment ($\boldsymbol{m}(s)$), where $s \in [0, L]$ is the centerline parameter. The manipulator is proximally fixed, with a constant proximal boundary condition ($\mathbb{P}(\boldsymbol{y}_0)$), and free to move at its distal end with boundary condition ($\mathbb{D}(\boldsymbol{y}_L)$). Unknown proximal states ($\boldsymbol{\xi}$) are optimized with Levenberg-Marquardt.

determine the pose of the material frame [31]. The values of $\hat{\boldsymbol{v}}$ and $\hat{\boldsymbol{u}}$ represent the intrinsic material strain and curvature of the CM in its undeformed reference state, taken as $\hat{\boldsymbol{v}} = [0, 0, 1]^T$ and $\hat{\boldsymbol{u}} = [0, 0, 0]^T$, and $\boldsymbol{K}_s, \boldsymbol{K}_b \in \mathbb{R}^{3\times3}$ are diagonal shear/extension and bending/torsion stiffness matrices, respectively [26].

Figure 10 shows a discretely magnetized continuum manipulator, consisting of alternating flexible non-magnetic, and rigid magnetic, material attached proximally, at $\boldsymbol{y}_0 \equiv \boldsymbol{y}(s = 0)$, to a rigid base. The manipulator body is subject to a distributed gravitational force $\boldsymbol{f}_g \in \mathbb{R}^3$, and the magnets experience a distributed wrench $\boldsymbol{w}_m = [\boldsymbol{f}_m^T, \boldsymbol{\tau}_m^T]^T$ when placed in a magnetic field, where m represents the magnet index.

The manipulator statics is solved as a boundary value problem (BVP), which contains information about the shape of the CM. Let $\mathbb{P}(\boldsymbol{y}_0)$ be the set of constant proximal boundary conditions and $\mathbb{D}(\boldsymbol{y}_L)$ the set of constant distal boundary conditions, which, assuming no load on the distal tip, are given by

$$\begin{aligned} \mathbb{P}(\boldsymbol{y}_0) &= \begin{bmatrix} \boldsymbol{p}_0^T & \boldsymbol{q}_0^T \end{bmatrix}^T \in \mathbb{R}^7, \\ \mathbb{D}(\boldsymbol{y}_L) &= \begin{bmatrix} \boldsymbol{n}_L^T & \boldsymbol{m}_L^T \end{bmatrix}^T = \begin{bmatrix} \boldsymbol{0}^T & \boldsymbol{0}^T \end{bmatrix}^T \in \mathbb{R}^6. \end{aligned} \tag{7}$$

In addition, $\boldsymbol{\xi} = [\boldsymbol{n}_0^T, \boldsymbol{m}_0^T]^T \in \mathbb{R}^6$ represents a vector of unknown proximal state parameters optimized with convex optimization to satisfy the distal boundary condition $\mathbb{D}(\boldsymbol{y}_L)$. We solve the BVP with forward integration using an explicit Runge-Kutta fourth order method, and convex optimization with Levenberg-Marquardt [32]. This approach requires discretization of the manipulator centerline. To fully capture the material transition between flexible and rigid material during forward integration, it is necessary to separately discretize each occurring flexible and rigid section (Fig. 10) into $N_f \in \mathbb{N}$ and $N_r \in \mathbb{N}$ subsections. Given $M \in \mathbb{N}$ recurring sections, there will exist $N = M(N_f + N_r) + 1$ material points along s. Forward integration from $\boldsymbol{y}_0$ then gives material state parameters of the CM Eqn. (4) at each discretized point along the manipulator centerline,

$$\boldsymbol{Y}(\boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{y}_0 & \boldsymbol{y}_1 & \dots & \boldsymbol{y}_N \end{bmatrix} \in \mathbb{R}^{13\times N}, \tag{8}$$

which we call a shape solution. The distal boundary condition Eqn. (7) and the distal state parameters ($\boldsymbol{y}_N$) obtained from the shape solution are compared to formulate a residual vector $\boldsymbol{r}$

$$\boldsymbol{r} = \mathbb{D}(\boldsymbol{y}_L) - \mathbb{D}(\boldsymbol{y}_N). \tag{9}$$

A solution of $\boldsymbol{\xi}$ is accepted if the residual norm is below an arbitrarily chosen error threshold, $E \in \mathbb{R}^+$, and $\boldsymbol{Y}(\boldsymbol{\xi})$ represents the shape of the CM in static equilibrium. Otherwise $\boldsymbol{\xi}$ is updated with Levenberg-Marquardt.

More information about Cosserat rod theory and the solution finding process of the BVP can be found in Appendix D.

3.1.2 Magnetic Field Map

An accurate model for the field ($\boldsymbol{B} \in \mathbb{R}^3$) and its spatial gradients ($\boldsymbol{B}_\nabla \in \mathbb{R}^{3\times3}$) is necessary to compute magnetically exerted torques and forces on magnetic agents. For electromagnets operating outside their saturation region, i.e. where their field is linear with current, a unit-current field ($\boldsymbol{\beta} \in \mathbb{R}^3$) and gradient ($\boldsymbol{\beta}_\nabla \in \mathbb{R}^{3\times3}$) model can be formulated and used to describe the field at different operating currents.

In a current-free workspace, the field is both divergence- and curl-free, which allows expressing the field as the gradient of a scalar potential

$$\boldsymbol{B}(\boldsymbol{p}) = \nabla \mathcal{V}(\boldsymbol{p}), \tag{10}$$

where $\boldsymbol{p} = [x, y, z]^T$ and $\mathcal{V}(\boldsymbol{p})$ is a nonlinear scalar function [5]. Consider a body coordinate frame attached to the face of an azimuthally symmetric field source, where $\hat{\boldsymbol{x}}$ is coincident with the axial direction (Fig. 9). The scalar potential is then defined as

$$\mathcal{V}(\boldsymbol{p}) = \sum_{k=1}^{N} \frac{\partial^k}{\partial x^k} \left(\frac{1}{\|\boldsymbol{p}\|} \right) \cdot a_k, \tag{11}$$

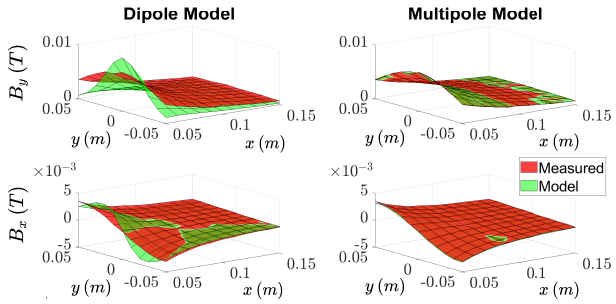

Figure 11: Surface plots of the x- and y-components of the magnetic field (B_x and B_y) obtained from measurements (red) and model (green) of the magnetic fields in the xy-plane of an air-cored electromagnet body frame of BigMag, comparing a least-squares fit of the point dipole model (left column) to our field model with $N = 6$ (right column).

where $k > 0$ to exclude the monopole term. The derivation of Eqn. (11) is given in the Appendix.

The scalar potential in Eqn. (11) is used to compute an analytical unit-current field description ($\boldsymbol{\beta} = \boldsymbol{B}|_{I=1}$) informed by field measurements. Measurements are performed in local xy-planes in front of an electromagnet, at $x \in \{0.05, \ldots, 0.15\}$ and $y \in \{-0.05, \ldots, 0.05\}$ meters. We define $\boldsymbol{b}_{ij} \in \mathbb{R}^3$ to be a field measurement at $\boldsymbol{p}_{ij} = [x_i, y_j, 0]^T$, with $i = 1, \ldots, m$ and $j = 1, \ldots, n$. To get Eqn. (11) to a suitable form for linear least squares (LLSQ) regression, expand Eqn. (11) to

$$\mathcal{V}(\boldsymbol{p}) = a_1\mathcal{V}_1(\boldsymbol{p}) + a_2\mathcal{V}_2(\boldsymbol{p}) \ldots + a_N\mathcal{V}_N(\boldsymbol{p}). \quad (12)$$

The scalar coefficients are subsequently found from

$$\begin{bmatrix} \nabla\mathcal{V}_1(\boldsymbol{p}_{11}) & \ldots & \nabla\mathcal{V}_N(\boldsymbol{p}_{11}) \\ & \vdots & \\ \nabla\mathcal{V}_1(\boldsymbol{p}_{mn}) & \ldots & \nabla\mathcal{V}_N(\boldsymbol{p}_{mn}) \end{bmatrix}^{\dagger} \begin{bmatrix} \boldsymbol{b}_{11} \\ \vdots \\ \boldsymbol{b}_{mn} \end{bmatrix} = \begin{bmatrix} a_1 \\ \vdots \\ a_N \end{bmatrix}, \quad (13)$$

where $[\cdot]^{\dagger}$ denotes the Moore-Penrose pseudoinverse. The (unit-current) field and gradient map is thereafter computed as

$$\begin{aligned} \boldsymbol{\beta}(\boldsymbol{p}) &= \frac{1}{I_f}\nabla\mathcal{V}(\boldsymbol{p}), \\ &\Rightarrow \boldsymbol{B}(\boldsymbol{p}, I) = \boldsymbol{\beta}(\boldsymbol{p}) \cdot I, \\ \boldsymbol{\beta}_{\nabla}(\boldsymbol{p}) &= \begin{bmatrix} \frac{\partial\boldsymbol{\beta}(\boldsymbol{p})}{\partial x} & \frac{\partial\boldsymbol{\beta}(\boldsymbol{p})}{\partial y} & \frac{\partial\boldsymbol{\beta}(\boldsymbol{p})}{\partial z} \end{bmatrix}, \\ &\Rightarrow \boldsymbol{B}_{\nabla}(\boldsymbol{p}, I) = \boldsymbol{\beta}_{\nabla}(\boldsymbol{p}) \cdot I, \end{aligned} \quad (14)$$

where $I_f \in \mathbb{R}^+$ represents the current at which measurements were taken.

A comparison between a LLSQ fit of a point dipole model and our model, for an air-cored electromagnet, is given in Fig. 11. The l-2 norm of the root-mean-square error for the dipole fit is 1.18 mT, and for the multipole model 0.16 mT. The resolution of the measurement equipment was 0.1 mT.

More information about modelling of the magnetic field and fitting of the multipole model can be found in Appendix C.

3.2 Orientation Control

In this section we present the control scheme used for multi-point orientation control of a discretely magnetized continuum manipulator. The actuation parameters for a mobile electromagnetic actuation system are electromagnet currents $\boldsymbol{I} \in \mathbb{R}^{N_I}$ and positions $\boldsymbol{\theta} \in \mathbb{R}^{N_\theta}$, which are updated to steer a magnetized CM in its workspace. Let $N = N_I + N_\theta$ represent the total number of DOFs provided by the system. Target orientations for each magnet are provided in terms of a desired axis of rotation $\boldsymbol{k}_m^d \in \mathbb{R}^3$ and angle of rotation $\phi_m^d \in \mathbb{R}$, defined relative to the global $\hat{\boldsymbol{z}}$ axis, and where $m \in \mathbb{N}$ denotes the magnet index. Magnet pose feedback is provided by a stereo vision setup adopted from [24]. Virtual changes in actuation parameters ($\boldsymbol{I}$ and $\boldsymbol{\theta}$) are used to predict resultant changes in magnet poses.

The control scheme in Fig. 12 is divided into four sections (**A**-**D**), which are discussed below.

3.2.1 Magnet Pose Sensing (A)

A stereo vision setup records real magnet positions $\boldsymbol{p}_m \in \mathbb{R}^3$ and orientation quaternions $\boldsymbol{q}_m \in \mathbb{R}^4$. In addition, the theoretical magnet poses $\{{}^t\boldsymbol{p}_m, {}^t\boldsymbol{q}_m\}$ are computed by solving the statics of the CM, subject to an external magnetic wrench ($\boldsymbol{w}_m = [\boldsymbol{\tau}_m^T, \boldsymbol{f}_m^T]^T \in \mathbb{R}^6$), as a BVP. Computation of the magnetically exerted wrench is discussed in section 3.2.2.

The stereo vision setup and shape reconstruction algorithm generates a 3D polynomial fit ($\boldsymbol{P}(s)$) to describe the shape of the manipulator, which is used to compute the position ($\boldsymbol{p}_m$) and orientation in the axis-angle representation, $\boldsymbol{k}_m \in \mathbb{R}^3$ ($\|\boldsymbol{k}_m\| = 1$) and $\phi_m \in \mathbb{R}$. The magnet orientation quaternion ($\boldsymbol{q}_m \in \mathbb{H}$) follows from the axis-angle representation as

$$\begin{aligned} &\boldsymbol{p}_m = \boldsymbol{P}(s_m), \quad \boldsymbol{p}'_m = \left(\boldsymbol{P}(s_m + \delta s) - \boldsymbol{P}(s_m)\right)/\delta s \\ &\phi_m = \cos^{-1}\left(\hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{p}}'_m\right) \\ &\boldsymbol{k}_m = \widehat{[\hat{\boldsymbol{z}} \times \hat{\boldsymbol{p}}'_m]} \end{aligned} \Rightarrow \boldsymbol{q}_m = \begin{bmatrix} \cos\left(\frac{\phi_m}{2}\right) \\ \boldsymbol{k}_m \sin\left(\frac{\phi_m}{2}\right) \end{bmatrix}, \quad (15)$$

The shape polynomial does not consider torsion and we assume that magnetically exerted forces are insufficient for extension of the manipulator along its

centerline. It is important to note that while the rotation of coordinate frames is unique, the rotation of vectors, i.e. the rotation of $\hat{\boldsymbol{z}}$ to $\hat{\boldsymbol{p}}'$, is not. As such we must define our desired orientation to remove any ambiguity caused from manipulator torsion,

$$\begin{aligned} \boldsymbol{p}_m^{d\prime} &= \boldsymbol{R}_z(\alpha_z)\boldsymbol{R}_y(\alpha_y)\hat{\boldsymbol{z}} \\ \phi_m^d &= \cos^{-1}\left(\hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{p}}_m^{d\prime}\right) \\ \boldsymbol{k}_m^d &= [\widehat{\hat{\boldsymbol{z}} \times \hat{\boldsymbol{p}}_m^{d\prime}}]. \end{aligned} \tag{16}$$

Similarly the magnet orientation quaternions obtained from the Cosserat rod model (${}^t\boldsymbol{q}_m$) cannot be immediately compared to measured orientations, and we must perform an additional step,

$$\begin{aligned} {}^t\boldsymbol{p}_m' &= \boldsymbol{R}({}^t\boldsymbol{q}_m)\hat{\boldsymbol{z}} \\ {}^t\phi_m &= \cos^{-1}\left(\hat{\boldsymbol{z}} \cdot {}^t\hat{\boldsymbol{p}}_m'\right) \\ {}^t\boldsymbol{k}_m &= [\widehat{\hat{\boldsymbol{z}} \times {}^t\hat{\boldsymbol{p}}_m'}]. \end{aligned} \tag{17}$$

Now the measured, desired, and theoretical magnet orientations in Eqns. (15)-(17) are compatible. Also, this approach will not result in errors in real and measured magnetic dipole moments of the magnets as long as they are oriented along the manipulator centerline. The inherent sign ambiguity of the axis-angle representation is solved by choosing the $\boldsymbol{k}_m$ and ${}^t\boldsymbol{k}_m$ of least angular distance from $\boldsymbol{k}_m^d$.

More information about the manipulator shape reconstruction algorithm can be found in Appendix B.

3.2.2 Magnetic Wrench (B)

The actuation parameters ($\boldsymbol{I}$ and $\boldsymbol{\theta}$) are used to compute a current-to-wrench map, which is composed of a current-to-field map, $\boldsymbol{T}_\beta(\boldsymbol{p}_m, \boldsymbol{\theta}) : \mathbb{R}^{N_I} \to \mathbb{R}^{8M}$ where $M \in \mathbb{N}$ is the total number of permanent magnets, and field-to-wrench map, $\boldsymbol{T}_\mu(\boldsymbol{q}_m) : \mathbb{R}^{8M} \to \mathbb{R}^{6M}$, defined at the center of mass of the magnets.

To compute $\boldsymbol{T}_\beta(\boldsymbol{p}_m, \boldsymbol{\theta})$, let $\{\mathcal{G}\}$ and $\{\mathcal{C}_j\}$ represent the global reference frame and body frame of an electromagnetic coil, $j = 1, \ldots, N_I$. A global description of the unit-current field (gradient) is then given by

$$\begin{aligned} {}^{\mathcal{G}}_{\mathcal{C}_j}\boldsymbol{\beta}(\boldsymbol{p}_m, \boldsymbol{\theta}) &= {}^{\mathcal{G}}_{\mathcal{C}_j}R(\boldsymbol{\theta})\,{}^{\mathcal{C}_j}\boldsymbol{\beta}\left({}^{\mathcal{C}_j}\boldsymbol{p}_m\right) \\ {}^{\mathcal{G}}_{\mathcal{C}_j}\boldsymbol{\beta}_\nabla(\boldsymbol{p}_m, \boldsymbol{\theta}) &= {}^{\mathcal{G}}_{\mathcal{C}_j}R(\boldsymbol{\theta})\,{}^{\mathcal{C}_j}\boldsymbol{\beta}_\nabla\left({}^{\mathcal{C}_j}\boldsymbol{p}_m\right){}^{\mathcal{G}}_{\mathcal{C}_j}R(\boldsymbol{\theta})^T. \end{aligned} \tag{18}$$

We define a map $\mathbb{G}(\boldsymbol{\beta}_\nabla) : \mathbb{R}^{3\times3} \to \mathbb{R}^5$ that returns a vector of the 5 linearly independent spatial gradient components of the field [18]. The current-to-field map for a single magnet is subsequently defined as

$$\boldsymbol{T}_\beta(\boldsymbol{p}_m, \boldsymbol{\theta}) = \begin{bmatrix} {}^{\mathcal{G}}_{\mathcal{C}_1}\boldsymbol{\beta} & \cdots & {}^{\mathcal{G}}_{\mathcal{C}_{N_I}}\boldsymbol{\beta} \\ \mathbb{G}\left({}^{\mathcal{G}}_{\mathcal{C}_1}\boldsymbol{\beta}_\nabla\right) & \cdots & \mathbb{G}\left({}^{\mathcal{G}}_{\mathcal{C}_{N_I}}\boldsymbol{\beta}_\nabla\right) \end{bmatrix}. \tag{19}$$

The wrench on a magnet with dipole moment $\boldsymbol{\mu}_m \in \mathbb{R}^3$ at position $\boldsymbol{p}_m$ is given by

$$\boldsymbol{w}_m = \begin{bmatrix} \boldsymbol{\tau}_m \\ \boldsymbol{f}_m \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_m \times \boldsymbol{B}(\boldsymbol{p}_m) \\ \nabla\left(\boldsymbol{\mu}_m \cdot \boldsymbol{B}(\boldsymbol{p}_m)\right) \end{bmatrix}. \tag{20}$$

The field-to-wrench map is then formulated, for a single dipole, as

$$\boldsymbol{T}_\mu(\boldsymbol{q}_m) = \begin{bmatrix} [\boldsymbol{\mu}_m]_\times & 0 \\ 0 & \mathbb{M}(\boldsymbol{\mu}_m) \end{bmatrix}, \tag{21}$$

where $\mathbb{M}(\boldsymbol{\mu}) : \mathbb{R}^3 \to \mathbb{R}^{3\times5}$ maps the independent spatial field gradients to forces on the dipole [18]. From this, the theoretical magnetic wrench is given by

$$\begin{aligned} \boldsymbol{w}_m &= \boldsymbol{T}_\mu(\boldsymbol{q}_m)\boldsymbol{T}_\beta(\boldsymbol{p}_m, \boldsymbol{\theta})\boldsymbol{I}, \\ &= \boldsymbol{T}(\boldsymbol{p}_m, \boldsymbol{q}_m, \boldsymbol{\theta})\boldsymbol{I} \end{aligned} \tag{22}$$

For two magnets, we then obtain $\boldsymbol{w} = [\boldsymbol{w}_1^T, \boldsymbol{w}_2^T]^T$.

3.2.3 Orientation Jacobian (C)

The orientation Jacobian ($\boldsymbol{J}_{k\phi} \in \mathbb{R}^{2M\times N}$) is computed numerically and describes how increments in the actuation parameters of the magnetic actuation system result in changes in magnet orientations. To compute the orientation Jacobian, each of the actuation parameters is iteratively and virtually incremented, the new virtual wrench computed, and the manipulator response predicted from the Cosserat rod model. The predicted change in magnet orientation is then included into $\boldsymbol{J}_{k\phi}$. This approach requires that N boundary value problems are solved, and we attempt to improve computational efficiency by providing an informed guess for manipulator base state parameters ($\boldsymbol{\xi}^i$), while solving each BVP.

Algorithm 1 presents the procedure of computing $\boldsymbol{J}_{k\phi}$. We simulate N open loop control cycles where, each time, one of the actuation parameters is incremented and the change in wrench, $\tilde{\boldsymbol{w}} = [\tilde{\boldsymbol{w}}_1^T, \tilde{\boldsymbol{w}}_2^T]^T$, computed. The informed guess $\boldsymbol{\xi}^i$ is then given by

$$\boldsymbol{\xi}^i\left(\boldsymbol{\xi}, \tilde{\boldsymbol{w}}, \boldsymbol{p}_m\right) = \boldsymbol{\xi} + \sum_{m=1}^{M} \begin{bmatrix} \tilde{\boldsymbol{f}}_m \\ \tilde{\boldsymbol{\tau}}_m + (\boldsymbol{p}_m - \boldsymbol{p}_0) \times \tilde{\boldsymbol{f}}_m \end{bmatrix}, \tag{23}$$

The manipulator shape, $\boldsymbol{Y}(\boldsymbol{\xi}^i)$ in Eqn. (8), is subsequently solved as a BVP, giving the change in theoretical magnet pose

$$\begin{aligned} \frac{d\left(\boldsymbol{k}_m^d \cdot \boldsymbol{k}_m\right)}{dI_j} &= \frac{\boldsymbol{k}_m^d \cdot \left({}^t\bar{\boldsymbol{k}}_m - {}^t\boldsymbol{k}_m\right)}{dI}, \\ \frac{d\phi_m}{dI_j} &= \frac{{}^t\bar{\phi}_m - {}^t\phi_m}{dI}, \end{aligned} \tag{24}$$

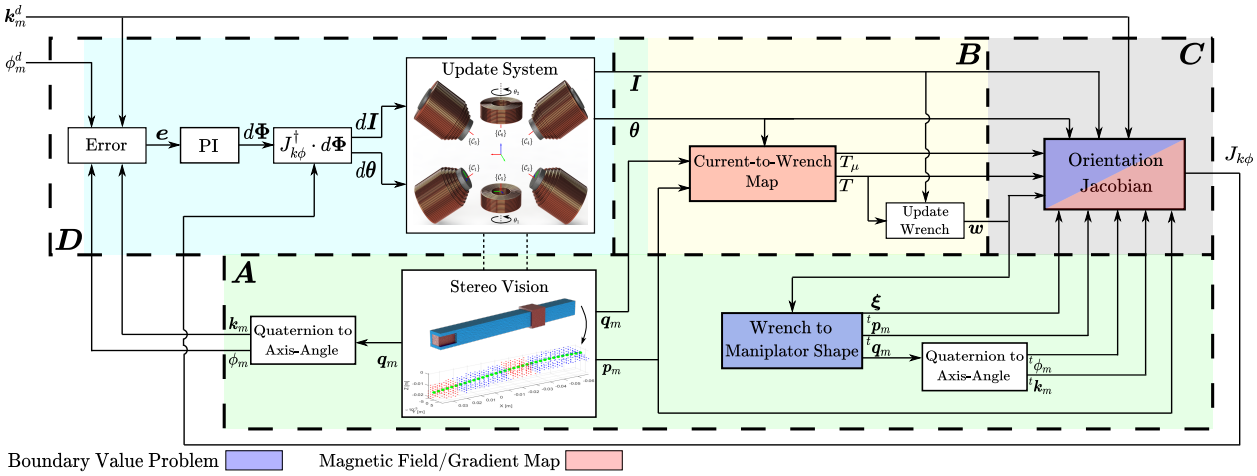

Figure 12: Block diagram for closed-loop orientation control, where magnet orientations are steered to desired axes and angles of rotation $\{\boldsymbol{k}_m^d, \phi_m^d\}$ by iteratively updating actuation parameters (currents $\boldsymbol{I}$ and positions $\boldsymbol{\theta}$). The manipulator is recorded using stereo vision to obtain the magnet positions ($\boldsymbol{p}_m$) and orientations ($\boldsymbol{q}_m$). The magnet poses are used, in conjuction with $\boldsymbol{I}$ and $\boldsymbol{\theta}$, to compute the theoretical wrench $\boldsymbol{w}$, from which the theoretical manipulator shape is computed, giving theoretical magnet poses $\{{}^t\boldsymbol{p}_m, {}^t\boldsymbol{q}_m\}$. Virtual increments of $\boldsymbol{I}$ and $\boldsymbol{\theta}$ are used to calculate the orientation Jacobian $\boldsymbol{J}_{k\phi}$. The error $\boldsymbol{e}$ between the magnet axes and angles of rotation, as measured $\boldsymbol{k}_m$ and ϕ_m, and as desired, $\boldsymbol{k}_m^d$ and ϕ_m^d, are fed to a proportional-integral controller to produce a desired orientation step $d\boldsymbol{\Phi}$. The orientation Jacobian and desired orientation step are used to compute actuation parameter increments $d\boldsymbol{I}$ and $d\boldsymbol{\theta}$.

and similarly for $d\theta$. The orientation Jacobian is then defined as

$$\boldsymbol{J}_{k\phi} = \begin{bmatrix} \frac{d(\boldsymbol{k}_1^d \cdot \boldsymbol{k}_1)}{dI_1} & \cdots & \frac{d(\boldsymbol{k}_1^d \cdot \boldsymbol{k}_1)}{d\theta_1} & \frac{d(\boldsymbol{k}_1^d \cdot \boldsymbol{k}_1)}{d\theta_2} \\ \frac{d\phi_1}{dI_1} & \cdots & \frac{d\phi_1}{d\theta_1} & \frac{d\phi_1}{d\theta_2} \\ \frac{d(\boldsymbol{k}_2^d \cdot \boldsymbol{k}_2)}{dI_1} & \cdots & \frac{d(\boldsymbol{k}_2^d \cdot \boldsymbol{k}_2)}{d\theta_1} & \frac{d(\boldsymbol{k}_2^d \cdot \boldsymbol{k}_2)}{d\theta_2} \\ \frac{d\phi_2}{dI_1} & \cdots & \frac{d\phi_2}{d\theta_1} & \frac{d\phi_2}{d\theta_2} \end{bmatrix}. \quad (25)$$

Its Moore-Penrose pseudoinverse is computed through singular value decomposition as $\boldsymbol{J}_{k\phi}^\dagger = \boldsymbol{V}\boldsymbol{S}^\dagger\boldsymbol{U}^T$, where $\boldsymbol{S} \in \mathbb{R}^{2M \times N}$ is a diagonal matrix of singular values σ_i for which $\sigma_i > \sigma_{i+1}$. To improve numerical stability we select singular values according to

$$S_{ii} = \begin{cases} \sigma_i & \text{If } \sigma_1/\sigma_i \leq 10^5 \\ 0 & \text{If } \sigma_1/\sigma_i > 10^5 \end{cases}. \quad (26)$$

3.2.4 Update Actuation Parameters (D)

The actuation parameters are updated to steer the magnet orientations to their respective target orientations. The orientation error is computed from the angular distances between desired and measured axes and angles of rotation,

$$\boldsymbol{e}_m = \begin{bmatrix} 1 - \boldsymbol{k}_m^d \cdot \boldsymbol{k}_m \\ \phi_m^d - \phi_m \end{bmatrix} \Rightarrow \boldsymbol{e} = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{e}_2 \end{bmatrix}. \quad (27)$$

The error is given to a proportional-integral controller to compute an orientation step ($d\boldsymbol{\Phi}$). Then the actuation parameters are updated according to

$$\begin{bmatrix} d\boldsymbol{I}^T, & d\boldsymbol{\theta}^T \end{bmatrix}^T = \boldsymbol{J}_{k\phi}^\dagger \cdot d\boldsymbol{\Phi}. \quad (28)$$

4 Results

Through experiments and simulations, we aim to show that it is possible to achieve convergence of multiple magnets to desired orientations by generating a non-uniform magnetic field. First, experimental validation of planar steering is described, followed by simulated results of 3D steering. The actuation of the CM in the experiment and simulation can be seen in the supplementary video.

4.1 Experiment

The dimensions of the CM in millimeters, time-evolved shapes of the CM during the experiment, and the magnet orientation trajectories are shown in Fig. 13. The manipulator consists of proximal and distal subsections of flexible polyurethane rubber (PMC-770, Smooth-On Inc., USA) and rigid neodymium (NdFeb) magnets. Material properties of the CM are given in table 1. The dipole moments ($\boldsymbol{\mu}_m$) are oriented perpendicular to, and down the, axial direction of the

Algorithm 1: Computation of the Orientation Jacobian

Input : Magnet pose info $\boldsymbol{p}_m, {}^t\boldsymbol{p}_m, \{{}^t\boldsymbol{k}_m, {}^t\phi_m\}$
Model base state parameters $\boldsymbol{\xi}$
Current/Field-Wrench maps $\boldsymbol{T}, \boldsymbol{T}_\mu$
Magnetic wrench $\boldsymbol{w}$
Actuation parameters $\boldsymbol{I}, \boldsymbol{\theta}$
Desired magnet axes of rotation $\boldsymbol{k}_m^d$

Output : Orientation Jacobian, $\boldsymbol{J}_{k\phi}$

$dI, d\theta \in \mathbb{R}^+$
/* Current to Orientation Change */
for $j \leftarrow 1$ **to** N_I **do**
  $\boldsymbol{I}_j = \boldsymbol{I}_j + dI$
  $\bar{\boldsymbol{w}} = \boldsymbol{T}\boldsymbol{I}$ // New wrench
  $\tilde{\boldsymbol{w}} = \bar{\boldsymbol{w}} - \boldsymbol{w}$ // Change in wrench
  Compute informed guess $\boldsymbol{\xi}^i\,(\boldsymbol{\xi}, \tilde{\boldsymbol{w}}, \boldsymbol{p}_m)$
  Solve manipulator shape $\boldsymbol{Y}(\bar{\boldsymbol{w}}, \boldsymbol{\xi}^i)$ // BVP
  Get new magnet poses $\{{}^t\bar{\boldsymbol{p}}_m, {}^t\bar{\boldsymbol{k}}_m, {}^t\bar{\phi}_m\} \leftarrow \boldsymbol{Y}$
  Fill column of $\boldsymbol{J}_{k\phi}$
  $\boldsymbol{I}_j = \boldsymbol{I}_j - dI$
end for
/* Position to Orientation Change */
for $j \leftarrow 1$ **to** N_θ **do**
  $\boldsymbol{\theta}_j = \boldsymbol{\theta}_j + d\theta$
  Recompute $\boldsymbol{T}_\beta(\boldsymbol{p}_m, \boldsymbol{\theta})$ and $\boldsymbol{T} = \boldsymbol{T}_\mu \boldsymbol{T}_\beta$
  $\bar{\boldsymbol{w}} = T\boldsymbol{I}$ // New wrench
  ⋮
  $\boldsymbol{\theta}_j = \boldsymbol{\theta}_j - d\theta$
end for
return $\boldsymbol{J}_{k\phi}$

CM, respectively. The CM is suspended horizontally in the workspace as shown in Fig. 13 at $t = 0$s. BigMag is used for actuation of the CM ($N_I = 6$ and $N_\theta = 2$), where the six electromagnets are oriented in plane with the CM. A side view of time-evolved shapes of the CM are presented. The magnet angles of rotation (AoR, ϕ_m, $m = 1, 2$) are defined relative to a global $\hat{\boldsymbol{z}}$ axis, and for the planar experiment the axes of rotation are $\boldsymbol{k}_m^d = \boldsymbol{k}_m = \hat{\boldsymbol{y}}$ (Fig. 13 at $t = 320$s). Steps in AoR are provided manually during the experiment until one of the electromagnet currents reached the upper limit of ±8 A. The AoR are shown against the elapsed time. The computation time required to numerically compute the orientation Jacobian ($J_{k\phi}$) is 158 ± 64 and 165 ± 53 ms with and without the informed guess in Eqn. (23), respectively. The absolute errors between the measured and theoretical magnet orientations are provided in Fig. 14.

A second experiment is performed where the magnet orientations were assiged trajectories. Time-evolved shapes of the CM and the trajectories for the desired, measured, and theoretical magnet orientations are provided in Fig. 15. The trajectory experiment is performed only with the informed guess. The computation time required to compute the orientation Jacobian is 200 ± 20 ms. The trajectory of the absolute errors between the measured and theoretical magnet orientations with time are provided in Fig. 16.

4.2 Simulation

Side, top, and isometric views of the CM shape during simulation with iteration number i ∈ ℕ are shown in Fig. 17. The magnetic field is generated with a virtual clone of BigMag extended with eight electromagnets ($N_I = 14$, $N_\theta = 2$). A unique non-uniform field is generated at the center of mass of each magnet to simultaneously steer the theoretical AoR (${}^t\phi_m$) of the magnets to their respective target AoR (ϕ_m^d). In addition, the theoretical axes of rotation (${}^t\boldsymbol{k}_m$) of the magnets are steered towards their respective target ($\boldsymbol{k}_m^d$) to minimize the angular distance (∠Axis) between them. The computation time required to compute the orientation Jacobian with and without providing an informed guess is 260±41 ms and 420±16, respectively. An additional simulation is shown in the Appendix.

5 Discussion

The closed-loop response of the magnet orientations in Figs. 13 and 15 shows the ability of the control algorithm to achieve accurate independent multi-point orientation control in 2D, with both small and higher errors between the measured and theoretical magnet orientations in Figs. 14 and 16. We observe, in the errors between the measured and theoretical magnet orientations, that the model error of the distal magnet moves with the model error of the proximal magnet. The model error is a result of trial-and-error guessed

	$\rho\ (kg \cdot m^{-3})$	$E\ (MPa)$	ν	$\mu\ (A \cdot m^2)$
PMC-770	1101	2.5	0.5	0
Magnet 1	7000	41.3	0.3	0.875
Magnet 2	7000	41.3	0.3	0.239

Table 1: Density (ρ), Elastic modulus (E), Poisson ratio (ν), and magnetic dipole moment (μ) of the CM used during experiments and simulation.

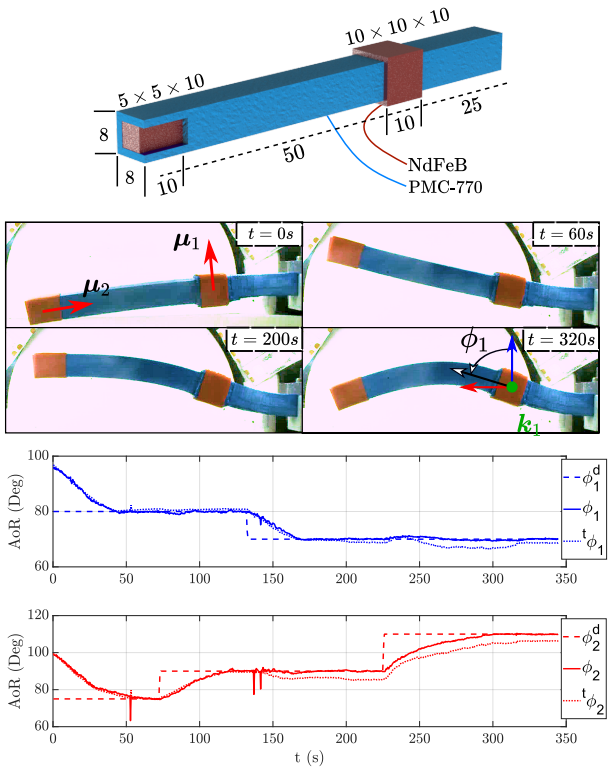

Figure 13: Model of the continuum manipulator (CM) with dimensions given in millimeters. The CM is made of flexible polyurethane rubber (PMC-770) and rigid neodymium magnets (NdFeB). Side views of time-evolved shapes of the continuum manipulator are given. Desired, ϕ_m^d, measured ϕ_m, and predicted, ${}^t\phi_m$, angles of rotation (AoR) are shown for magnet 1 (blue) and 2 (red) in degrees against time t.

material properties of the CM and therefore subject to inaccuracies. The trajectories of the errors suggest that the model error of prior magnet orientations primarily affect those of subsequent magnets, and minimally in the other direction, and less vice versa.

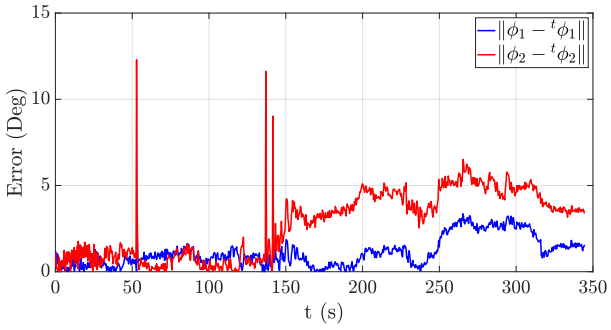

Figure 14: Absolute error between the measured magnet orientations (ϕ_m) and predicted theoretical magnet orientations (${}^t\phi_m$) during the experiment shown in Fig. 13.

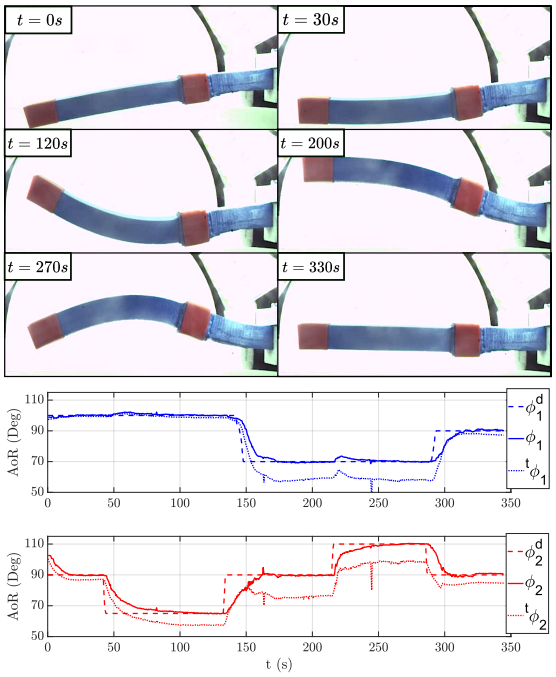

Figure 15: Time-evoled shapes of the continuum manipulator during a second experiment. Magnet orientation trajectories were manually assigned during runtime. The error between measured (ϕ_m) and theoretical (${}^t\phi_m$) is larger than those presented in the paper, despite not changing the used material properties. The measured magnet orientations showed simultaneous and independent convergence to desired orientations (ϕ_m^d). The subscript $m = 1, 2$ denotes the magnet index, as counted from the manipulator base.

We also observe peaks in the orientation and error trajectories. These are most likely caused by errors during shape reconstruction of the manipulator in Eqn. (15). In simulation it was shown that the control algorithm is able to achieve 3D orientation control of multiple magnets, given that the magnetic actuation

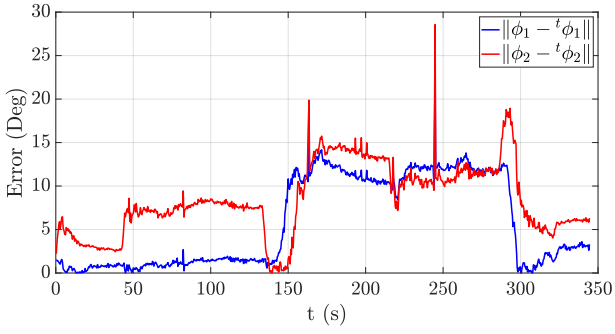

Figure 16: Absolute error between the measured magnet orientations (ϕ_m) and predicted theoretical magnet orientations (${}^t\phi_m$) during the experiment shown in Fig. 15

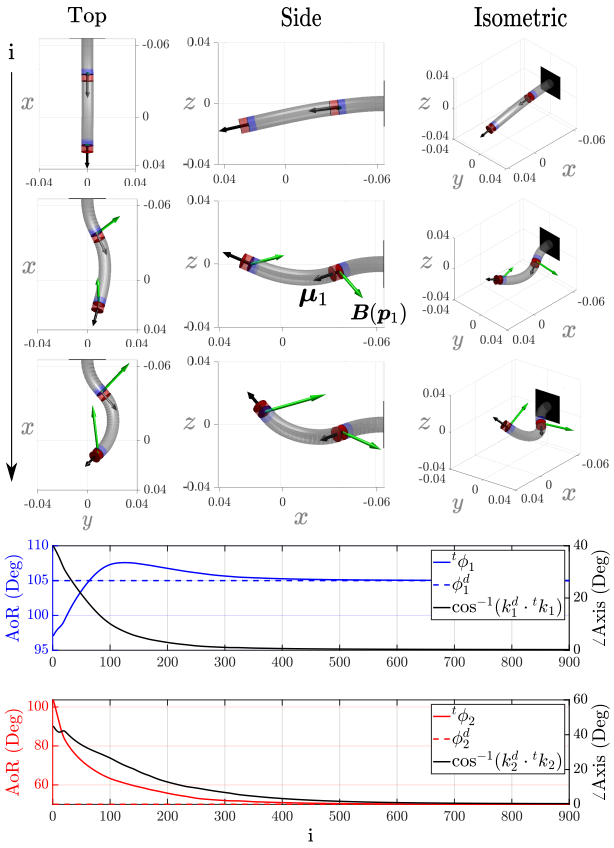

Figure 17: Top, side, and isometric views of continuum manipulator (CM) shape at a start, intermediate, and final iteration numbers (i) with dimensions shown in meters. The trajectory of the magnet angles of rotation (AoR), for magnets 1 and 2, are given in blue and red, respectively. In addition, the trajectories of the angular distances between the theoretical and desired axes of rotation (∠Axis) are shown.

system provides enough degrees of freedom. The methodology shown here can be extended to other systems consisting of moving electromagnets, provided the magnetic field can be expressed as a function of positions and currents of the electromagnets.

The computation time required for computing the orientation Jacobian ($J_{k\phi}$) limits the achievable control frequency. Together with the shape reconstruction algorithm, a control frequency of just 2 Hz was achieved. The primary drawback of the presented control strategy is that it requires a boundary value problem (BVP) to be solved for each available degree of freedom of the magnetic actuation system. For BigMag, this meant that eight BVPs had to be solved. With 15 points of discretization along the centerline of the manipulator, finding a solution to the BVP took on average 20 ms. In open-loop the informed guess in Eqn. (23) improves the computation time since the feedback and the forward model are both calculated using the Cosserat rod model. However, the informed guess did not result in an improvement during closed-loop control, because when stereo vision is used for closed-loop control, the feedback diverges from the forward model and therefore we do not see significant improvement to the computation time by using the informed guess.

The orientation Jacobian becomes increasingly computationally expensive for larger (redundant) magnetic actuation systems. Therefore it is desirable to use a control strategy where the numerical computation is independent of the number of magnetic degrees of freedom rather than the number of permanent magnets. In contrast to the current method, this would involve providing desired magnet orientations to the BVP, and computing the required wrenches (inverse model based control). Furthermore, previous works have assumed that the effect of magnetic forces is negligible, which can also serve to reduce the computational load [24].

Finally, we notice from simulation that there is a noticeable difference between the results when the magnetic forces are taken into account, as opposed to when they are not, as shown in Fig. 18. Therefore the formulated magnetic field (gradient) model has shown able to accurately describe the generated magnetic field of an azimuthally symmetric field source. In addition, because the model conforms to Maxwell's equations for the magnetic field in a current-free workspace, it can also be applied to accurately estimate the spatial gradients of the field. Compared to previous works, this provides the ability to utilize both magnetically exerted forces in addition to torques for closed-loop control of magnetized CMs.

6 Conclusions and Future Work

In this thesis and paper, an approach for closed-loop multi-point orientation control of discretely magnetized continuum manipulators is presented. An accurate model is developed to compute the magnetic field (gradients), based on a Cartesian multipole expansion of a scalar potential, which is applied to BigMag, an array of six mobile electromagnetic coils. In addition, a quasi-static model of the manipulator is developed using Cosserat rod theory. This model incorporates distributed magnetic torques and forces on two permanent magnets, positioned at discrete locations along the manipulator centerline, and a distributed gravitational force. Feedback of the manipulator shape is provided by a stereo vision system to track magnet positions and orientations. A numerically-computed

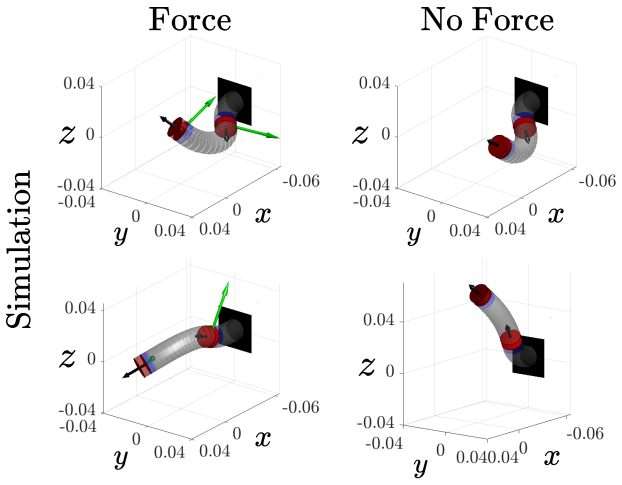

Figure 18: Effect of magnetic forces on final continuum manipulator shapes. The figures in the left column show the final manipulator shapes at the end of the performed 3D simulation in Fig. 17 and additional 3D simulation in Fig. A.1, including both magnetically exerted torques and forces. The figures in the right column shows the respective manipulator shapes when acting magnetic forces on the dipoles are set to zero (gravitational forces are still included). The black arrows originating from the magnet centers show the direction of the magnetic dipole moments; the green arrows show the local direction of the field, with lengths proportional to the field strength. Dimensions are shown in meters.

Jacobian based on an informed guess is used for estimation of necessary updates to the actuation parameters.

The presented control strategy has shown able to achieve accurate independent orientation control of two permanent magnets in 2D inside BigMag, and in 3D using a virtual clone of BigMag extended with eight additional electromagnets. However, the forward model approach limits achievable control frequencies as the computational cost of numerical solutions increases with degrees of freedom of the magnetic actuation system. Instead, for larger (redundant) actuation systems, we hypothesize that inverse model control with Cosserat rod modelling will significantly reduce computational cost, as numerical problems become independent of the number of degrees of freedom provided by the magnetic actuation system.

Future work will focus on reformulating the control strategy to inverse model based control for closed-loop multi-point steering of magnetized continuum manipulators, because we hypothesize that this will significantly increase the achievable control frequency. Furthermore, since the presented magnetic field map is both accurate in describing the magnetic field and the spatial gradients of the magnetic field, we would like to apply it to magnetic force-based manipulation tasks.

Appendices

Appendix A Additional Simulation

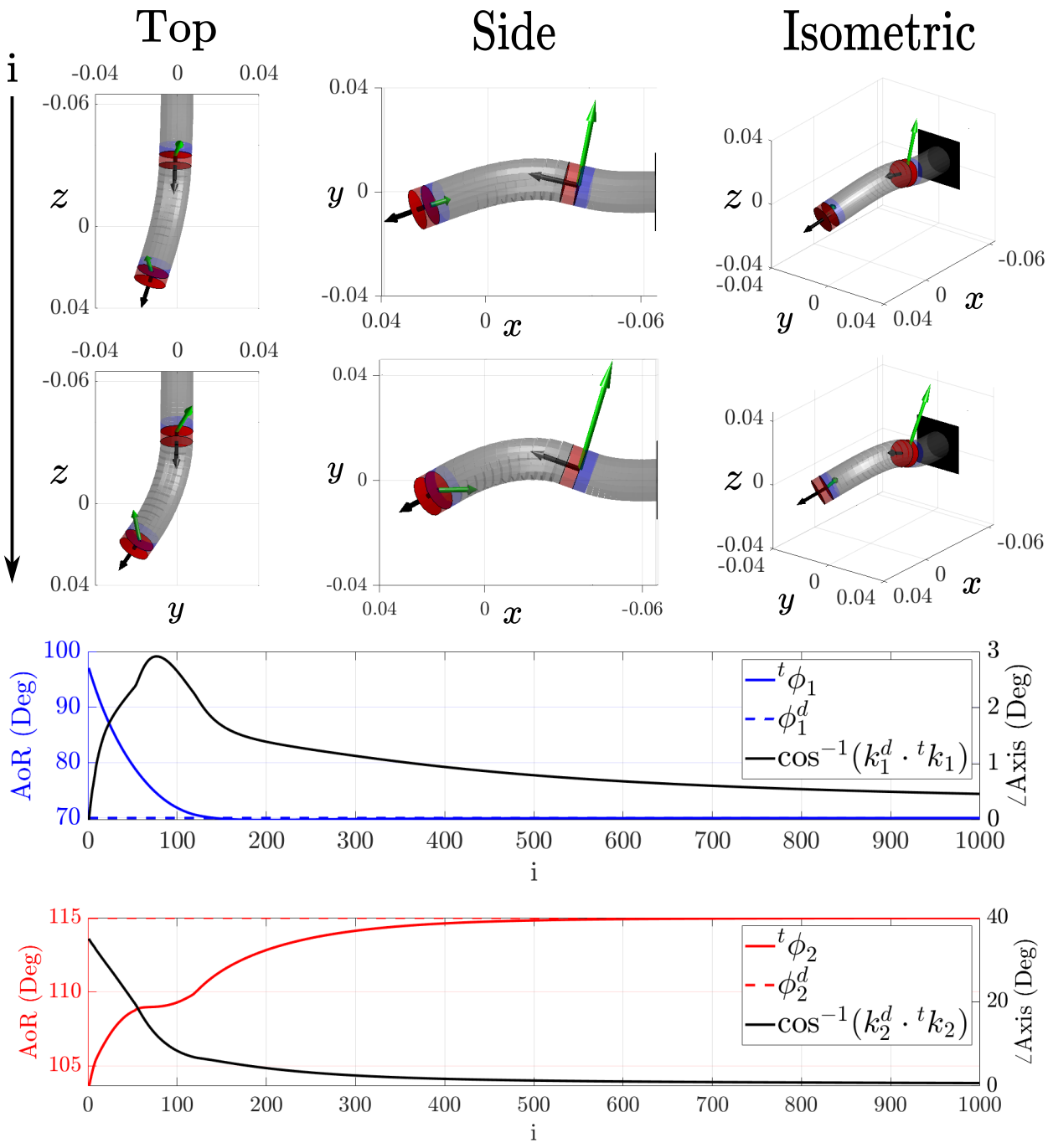

Figure A.1: Top, side, and isometric views of continuum manipulator (CM) shape at intermediate and final iteration numbers (i) with dimensions shown in meters. The starting shape is equal to the one shown in Fig. 17. The trajectory of the magnet angles of rotation (AoR), for magnets 1 and 2, are given in blue and red, respectively. In addition, the trajectories of the angular distances between the theoretical and desired axes of rotation (∠Axis) are shown.

Appendix B Stereo Vision

In this part of the appendix we first discuss the stereo vision setup and the shape reconstruction algorithm that reconstructs the continuum manipulator centerline. This is followed by a discussion on how the orientations of the magnets along the continuum manipulator are obtained from the reconstructed centerline.

B.1 Shape Reconstruction

The stereo vision algorithm is adapted from [24]. The workspace of BigMag is recorded with 2 cameras, providing a side and top view of the continuum manipulator (CM). The recorded images are transformed to a binary image, segmenting the manipulator silhouette as a 3D point cloud as shown in Fig. B.1. Each point represents a voxel, $\boldsymbol{v} \in \boldsymbol{R}^3$, containing the Cartesian position of the point. The set of all voxels that make up the 3D point cloud of the CM is defined as

$$\mathbb{V} = \{\boldsymbol{v}_i\} \subset \Omega_w, \qquad i = 1, \dots, n \tag{B.1}$$

where Ω_w is the set of all voxels (part of the manipulator or not) that make up the workspace. Given the set of voxels ($\mathbb{V}$) that represent the manipulator body in 3D space, we now want to find a shape function to mathematically represent the manipulator shape, called the shape polynomial.

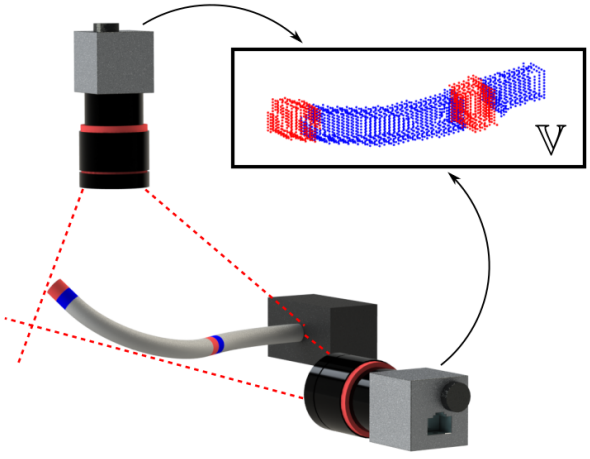

Figure B.1: A stereo vision setup records the actuation workspace and measures the spatial volume occupied by the body of the continuum manipulator as a 3D spatial point cloud. Each point represents a small volume inside the workspace, called a voxel. The set of points, or set of voxels, is contained in the set $\mathbb{V}$.

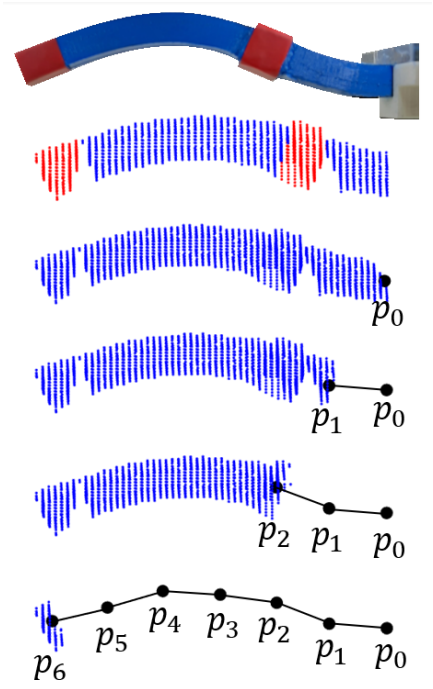

Figure B.2: Representation of the shape reconstruction algorithm. The body of the continuum manipulator (CM) is obtained as a 3D spatial point cloud from the stereo vision setup. We discretize the centerline of the CM into N small segments. We know the origin of the CM ($\boldsymbol{p}_0$) and search for voxels within its vicinity. A straight line is plot through the found voxels towards a new point ($\boldsymbol{p}_1$). This procedure is repeated until the last point ($\boldsymbol{p}_N$) is reached.

To formulate the shape polynomial, consider the CM material centerline to be characterized by a centerline parameter $s \in [0, L]$, where $L \in \mathbb{R}^+$ represents the CM length. Each point along the centerline has a position, $\boldsymbol{p}(s) \equiv \boldsymbol{p}_s \in \mathbb{R}^3$. The iterative process to approach material positions along the centerline is shown in Fig. B.2. First, we decide on an arbitrary number, $N = 6$, of discrete (first order) subsections along s. Each subsection will have a length $d = L/N$. Starting from the known origin ($\boldsymbol{p}_j$), collect all voxels within its vicinity according to $\mathbb{V}^j \leftarrow \{\boldsymbol{v}_i \in \mathbb{V} \mid \|\boldsymbol{v}_i - \boldsymbol{p}_j\| \leq d\}$, where $j = 1, \dots N$. A line $\boldsymbol{d}^j$ is fit with linear least squares from $\boldsymbol{p}_{j-1}$ through the set of voxels $\mathbb{V}^j$

$$\boldsymbol{d}^j = \begin{bmatrix} \|\boldsymbol{v}_1^j - \boldsymbol{p}_{j-1}\|\boldsymbol{I}_3 \\ \vdots \\ \|\boldsymbol{v}_m^j - \boldsymbol{p}_{j-1}\|\boldsymbol{I}_3 \end{bmatrix}^{\dagger} \begin{bmatrix} \boldsymbol{v}_1^j - \boldsymbol{p}_{j-1} \\ \vdots \\ \boldsymbol{v}_m^j - \boldsymbol{p}_{j-1} \end{bmatrix}. \tag{B.2}$$

To Limit possible zig-zagging between links, we limit the angular distance between them, enforcing

$\angle\left(\hat{\boldsymbol{d}}^j \cdot \hat{\boldsymbol{d}}^{j-1}\right) \leq \theta_t$, where $\theta_t \in \mathbb{R}^+$ represent a threshold angle. We find the angular distance between links from

$$\theta_j = \cos^{-1}\left(\hat{\boldsymbol{d}}^j \cdot \hat{\boldsymbol{d}}^{j-1}\right). \tag{B.3}$$

If $\theta_j > \theta_t$, we rotate $\boldsymbol{d}^j$ back to θ_t as follows. Let $\boldsymbol{k}_j \in \mathbb{R}^3$ represent the axis of rotation

$$\boldsymbol{k}_j = \frac{\boldsymbol{d}^j \times \boldsymbol{d}^{j-1}}{\|\boldsymbol{d}^j \times \boldsymbol{d}^{j-1}\|}, \tag{B.4}$$

and let $\boldsymbol{R}(\boldsymbol{k}, \theta) \in SO(3)$ represent the rotation matrix associated with an axis-angle representation, defined as

$$\boldsymbol{R}(\boldsymbol{k}, \theta) = \boldsymbol{I}_3 + \sin\theta\,[\boldsymbol{k}]_\times + (1 - \cos\theta)\,[\boldsymbol{k}]_\times^2, \tag{B.5}$$

where $[\cdot]_\times : \mathbb{R}^3 \to \mathbb{R}^{3\times 3}$ represents a map to a skew-symmetric matrix. We now define the orientation threshold difference as $\tilde{\theta} = \theta_t - \theta_j$, and map the vector $\boldsymbol{d}^j$ according to

$$\boldsymbol{d}^j = \begin{cases} \boldsymbol{d}^j & \text{If } \tilde{\theta} \geq 0 \\ \boldsymbol{R}(\boldsymbol{k}_j, \tilde{\theta})\boldsymbol{d}^j & \text{If } \tilde{\theta} < 0 \end{cases}. \tag{B.6}$$

Finally the next material point, $\boldsymbol{p}_j$, is computed from

$$\boldsymbol{p}_j = \boldsymbol{p}_{j-1} + d \cdot \hat{\boldsymbol{d}}^j. \tag{B.7}$$

The above procedure is repeated until a set of material points, $\mathbb{P} = \{\boldsymbol{p}_0, \ldots, \boldsymbol{p}_N\}$, is obtained. Finally, a shape polynomial, $\boldsymbol{P}(s)$, is fit through the points in as a function of s, noting that $\boldsymbol{p}_j \equiv \boldsymbol{p}(s = j \cdot d)$,

$$\boldsymbol{P}(s) = \boldsymbol{c}_\zeta s^\zeta + \ldots + \boldsymbol{c}_1 s + \boldsymbol{p}_0, \tag{B.8}$$

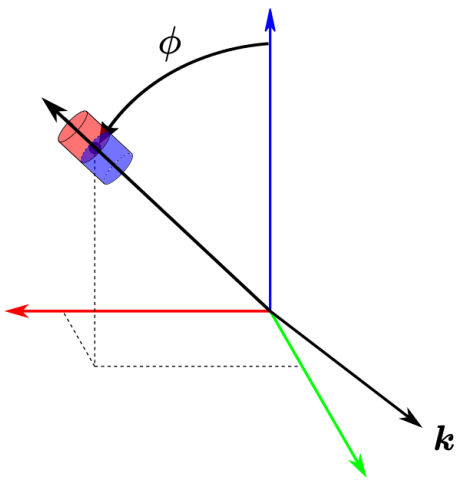

Figure B.3: Measured magnet orientations are represented with an angle of rotation ($\phi \in \mathbb{R}$) and axis of rotation ($\boldsymbol{k} \in \mathbb{R}^3$) relative to a reference axis ($\hat{\boldsymbol{z}}$, blue).

where $\boldsymbol{c}_\zeta \in \mathbb{R}^3$ is a vector of coefficients for the polynomial term of order $\zeta \in \mathbb{N}$. The coefficients are found, again, with linear least squares regression as

$$\begin{bmatrix} \boldsymbol{c}_\zeta \\ \vdots \\ \boldsymbol{c}_1 \end{bmatrix} = \begin{bmatrix} s_1^\zeta \boldsymbol{I}_3 & \ldots & s_1 \boldsymbol{I}_3 \\ \vdots & \ldots & \vdots \\ s_N^\zeta & \ldots & s_N \boldsymbol{I}_3 \end{bmatrix}^\dagger \begin{bmatrix} \boldsymbol{p}_1 - \boldsymbol{p}_0 \\ \vdots \\ \boldsymbol{p}_N - \boldsymbol{p}_0 \end{bmatrix}. \tag{B.9}$$

From the CM shape polynomial in Eqn. (B.8) the orientation of the magnets, carried by the CM, can be computed.

B.2 Magnet Orientation Estimation

Magnet orientations are defined according to the axis-angle representation. The axis-angle representation of orientation parameterizes the orientation of a body by an unit axis of rotation, $\boldsymbol{k} \in \mathbb{R}^3$, and angle of rotation, $\phi \in \mathbb{R}$, as shown in Fig. B.3). Of particular importance is to note that the rotation from the triad $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$ to a new triad $\{\hat{\boldsymbol{x}}', \hat{\boldsymbol{y}}', \hat{\boldsymbol{z}}'\}$ is defined by a unique axis and angle of rotation. However, as shown in Fig. B.3, the mapping of a vector $\hat{\boldsymbol{z}}$ to another vector by an orientation defined by $\boldsymbol{k}$ and ϕ is not unique, since the mapped vector can rotate about its own axis and remain the same vector. This is an important property when considering the shape polynomial of Eqn. (B.8). From the shape polynomial we can only find the local axial direction of the CM centerline, and it is not possible to find the principal axes of its cross-section to formulate a triplet. Therefore the solution of the magnet orientation given below is not unique.

The magnet positions along the CM centerline is known and given by centerline position $s_m \in (0, L]$. Under the assumption that the exerted wrenches on the CM is insufficient for extension, s_m is constant, and the magnet spatial position and direction vector (gradient) is given by

$$\begin{aligned} \boldsymbol{p}_m &= \boldsymbol{P}(s_m), \\ \boldsymbol{p}'_m \equiv \partial_s \boldsymbol{P}(s_m) &= \frac{\boldsymbol{P}(s_m + \delta s) - \boldsymbol{P}(s_m)}{\delta s}, \end{aligned} \tag{B.10}$$

where $\delta s = 0^+$, shown in Fig. B.4. The axis and angle of rotation of the magnet are thereafter computed relative to the global $\hat{\boldsymbol{z}}$ axis as

$$\begin{aligned} \boldsymbol{k}_m &= [\widehat{\hat{\boldsymbol{z}} \times \hat{\boldsymbol{p}}'_m}] \\ \phi_m &= \cos^{-1}\left(\hat{\boldsymbol{z}} \cdot \hat{\boldsymbol{p}}'_m\right), \end{aligned} \tag{B.11}$$

with an associated orientation quaternion

$$q_m = \begin{bmatrix} \cos\left(\frac{\phi_m}{2}\right) \\ \boldsymbol{k}_m \cdot \sin\left(\frac{\phi_m}{2}\right) \end{bmatrix}. \tag{B.12}$$

We reiterate that a vector mapping of $\hat{\boldsymbol{z}}$ to $\boldsymbol{p}'_m$ is not unique, since twisting of the manipulator cannot be measured.

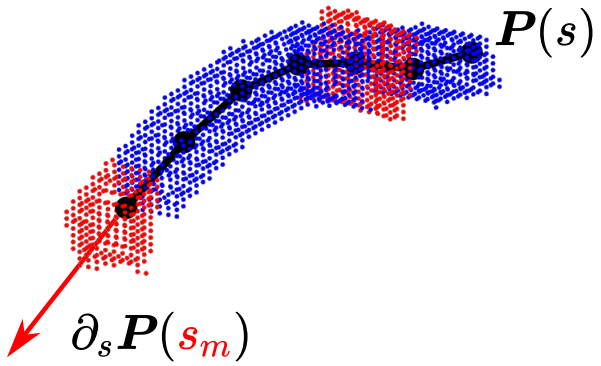

Figure B.4: A 3D shape polynomial ($\boldsymbol{P}(s)$) as a function of centerline parameter ($s \in [0, L]$) is fit through the set of points in figure B.2. The partial derivative of the shape polynomial at known magnet centerline position ($\partial_s \boldsymbol{P}(s_m)$) gives the direction of the shape polynomial at the magnet, which is used to compute the axis and angle of rotation associated with the magnet orientation (Fig. B.3)

Appendix C Magnetic Field Model

In this part of the appendix we first discuss the spherical scalar potential, commonly found in literature on electrodynamics [5], and derive a Cartesian scalar potential that can be used to fit an accurate magnetic field model to measurements of a magnetic field from an azimuthally symmetric magnet. This is followed by a discussion on how the magnetic field model is fit. Finally a Matlab script is provided that contains the fitting algorithm used in this work.

C.1 Deriving the Scalar Potential

A scalar potential is defined as a scalar function which is a solution to the Laplace equation, also called a harmonic scalar function. Considering the Maxwell equations of the magnetic field generated by an electromagnet with current I, at position $\boldsymbol{p} = [x, y, z]^T$ in a current-free workspace,

$$\begin{aligned} \nabla \cdot \boldsymbol{B}(\boldsymbol{p}, I) &= 0 \\ \nabla \times \boldsymbol{B}(\boldsymbol{p}, I) &= 0 \end{aligned} \tag{C.1}$$

the second, curl-free, property of the magnetic field allows expressing the field as the gradient of a scalar function $V(\boldsymbol{p}, I)$, and when we assume the field to be linear with current,

$$\boldsymbol{B}(\boldsymbol{p}, I) = \nabla V(\boldsymbol{p}) \cdot I, \tag{C.2}$$

such that,

$$\nabla \times \nabla V(\boldsymbol{p}) \cdot I = 0, \tag{C.3}$$

regardless of V. The first, divergence-free, property of the magnetic field constrains V to be a harmonic scalar function, since

$$\nabla \cdot (\nabla V(\boldsymbol{p}) \cdot I) = \nabla^2 V(\boldsymbol{p}) \cdot I = 0. \tag{C.4}$$

The general solution of $V(\boldsymbol{p})$, for an azimuthally symmetric magnetic field source, is given in electrodynamics textbooks as a sum of Legendre polynomials in spherical coordinates,

$$V(r, \cos\theta) = \sum_{k=1}^{\infty} \left(A_k \cdot r^k + B_k \cdot \frac{1}{r^{k+1}} \right) P_k(\cos\theta), \tag{C.5}$$

where $r = \|\boldsymbol{p}\|$ and θ is the angle between the magnet north-pole direction and position vector $\boldsymbol{p}$, shown in Fig. C.1, $A_k, B_k \in \mathbb{R}$ are unknown constants, and $P_k(\cos\theta)$ is the k^{th} Legendre polynomial in the variable $\cos\theta$. We can split Eqn. (C.5) into

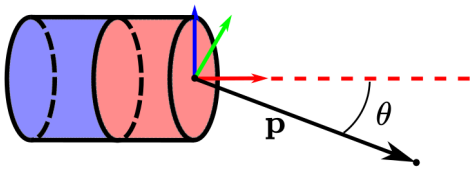

Figure C.1: A magnet with magnetic dipole moment along local $\hat{\boldsymbol{x}}$ (red) and position vector $\boldsymbol{p} \in \mathbb{R}^3$ with angular distance ($\theta \in \mathbb{R}$) from the dipole moment axis.

$$\begin{aligned} V(r, \cos\theta) = &\sum_{k=1}^{\infty} A_k \cdot r^k \cdot P_k(\cos\theta) + \\ &\sum_{k=1}^{\infty} B_k \cdot \frac{1}{r^{k+1}} \cdot P_k(\cos\theta) \\ = &V^A(r, \cos\theta) + V^B(r, \cos\theta). \end{aligned} \tag{C.6}$$

Next we evaluate the two terms in Eqn. (C.6). To start, the first five Legendre polynomials, excluding the zeroth, are given by

$$\begin{aligned} P_1(\cos\theta) &= \cos\theta, \\ P_2(\cos\theta) &= (3\cos^2\theta - 1)/2, \\ P_3(\cos\theta) &= (5\cos^3\theta - 3\cos\theta)/2, \\ P_4(\cos\theta) &= (35\cos^4\theta - 30\cos^2\theta + 3)/8, \\ P_5(\cos\theta) &= (63\cos^5\theta - 70\cos^3\theta + 15\cos\theta)/8. \end{aligned}$$

Then, considering that for a local x-directed dipole we get $x = r \cdot \cos\theta$, the first (simplified) five terms of $V^A(\boldsymbol{p})$ are given as

$$\begin{aligned} V_1^A(\boldsymbol{p}) &= A_1 \cdot x \\ V_2^A(\boldsymbol{p}) &= \frac{A_2}{2} \cdot (2x^2 - (y^2 + z^2)) \\ V_3^A(\boldsymbol{p}) &= \frac{A_3}{2} \cdot (2x^3 - 3x(y^2 + z^2)) \\ V_4^A(\boldsymbol{p}) &= \frac{A_4}{8} \cdot (8x^4 - 24x^2(y^2 + z^2) + 3(y^2 + z^2)^2) \\ V_5^A(\boldsymbol{p}) &= \frac{A_5}{8} \cdot (8x^5 - 40x^3(y^2 + z^2) + 15x(y^2 + z^2)^2), \end{aligned}$$

which follows the pattern

$$\begin{aligned} V_k^A(\boldsymbol{p}) &= \tilde{A}_k \left(\|\boldsymbol{p}\|^{2k+1} \frac{\partial}{\partial x} V_{k-1}^A(\boldsymbol{p}) \right) \\ &= \tilde{A}_k \left(\|\boldsymbol{p}\|^{2k+1} \frac{\partial^k}{\partial x^k} \frac{1}{\|\boldsymbol{p}\|} \right), \end{aligned} \tag{C.7}$$

where $\tilde{A}_k$ is a scaled version of A_k. In a similar fashion

we can get the first five terms of $V^B(\boldsymbol{p})$ as

$$
\begin{aligned}
V_1^B(\boldsymbol{p}) &= B_1 \cdot \frac{x}{\|\boldsymbol{p}\|^3} \\
V_2^B(\boldsymbol{p}) &= \frac{B_2}{2} \cdot \frac{2x^2 - (y^2+z^2)}{\|\boldsymbol{p}\|^5} \\
V_3^B(\boldsymbol{p}) &= \frac{B_3}{2} \cdot \frac{2x^3 - 3x(y^2+z^2)}{\|\boldsymbol{p}\|^7} \\
V_4^B(\boldsymbol{p}) &= \frac{B_4}{8} \cdot \frac{8x^4 - 24x^2(y^2+z^2) + 3(y^2+z^2)^2}{\|\boldsymbol{p}\|^9} \\
V_5^B(\boldsymbol{p}) &= \frac{B_5}{8} \cdot \frac{8x^5 - 40x^3(y^2+z^2) + 15x(y^2+z^2)}{\|\boldsymbol{p}\|^{11}},
\end{aligned}
$$

which follows the pattern

$$
\begin{aligned}
V_k^B(\boldsymbol{p}) &= \tilde{B}_k \left(\frac{\partial}{\partial x} V_{k-1}^B(\boldsymbol{p}) \right) \\
&= \tilde{B}_k \left(\frac{\partial^k}{\partial x^k} \frac{1}{\|\boldsymbol{p}\|} \right),
\end{aligned}
\tag{C.8}
$$

where $\tilde{B}_k$ is a scaled version of B_k. We observe that the numerators of V_k^B and V_k^A are differently scaled versions of themselves. In Matlab the divergence of the gradients of both V_k^A and V_k^B, for $k = 1, \ldots, 5$, was confirmed to be zero for arbitrary $\boldsymbol{p}$.

With the scalar potential terms in Eqn. (C.6) rewritten in Cartesian coordinates, we want to find a model of the magnetic field from the scalar potential in Eqn. (C.2). When an x-directed magnetic field source is at the origin we consider that the field goes to zero at infinity, i.e.

$$
\lim_{x \to \infty} \boldsymbol{B}(\boldsymbol{p}, I) = \boldsymbol{0}.
$$

Because the scalar terms in the multipole expansion go exponentially to infinity and zero,

$$
\lim_{x \to \infty} V_k^A(\boldsymbol{p}) = \pm\infty, \qquad \lim_{x \to \infty} V_k^B(\boldsymbol{p}) = 0,
$$

we set all coefficients A_k to zero, which would correspond to a field source away from the origin. Therefore our scalar potential is chosen as

$$
V(\boldsymbol{p}) = V^B(\boldsymbol{p}) = \sum_{k=1}^{\infty} \tilde{B}_k \left(\frac{\partial^k}{\partial x^k} \frac{1}{\|\boldsymbol{p}\|} \right), \tag{C.9}
$$

where the first term ($k = 1$) represents the well known analytical point dipole model as a first-order approximation of the field. To show this, consider a magnet with magnetic dipole moment $\boldsymbol{\mu} = \mu \cdot \hat{\boldsymbol{x}}$ as shown in Fig. C.2. The point-dipole model of the field is given by

$$
\boldsymbol{B}_1(\boldsymbol{p}, \boldsymbol{\mu}) = \frac{\mu_0}{4\pi} \cdot \frac{1}{\|\boldsymbol{p}\|^5} \left(3\boldsymbol{p}\boldsymbol{p}^T - \|\boldsymbol{p}\|^2 \boldsymbol{I}_3 \right) \boldsymbol{\mu},
$$

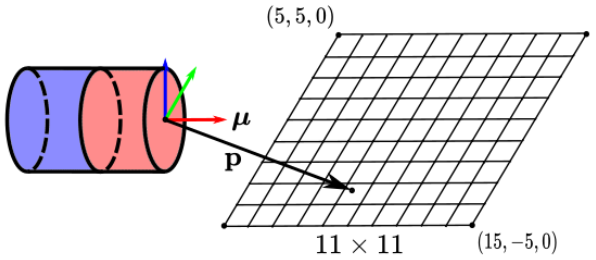

Figure C.2: A magnet with dipole moment ($\boldsymbol{\mu} \in \mathbb{R}^3$) along local $\hat{\boldsymbol{x}}$ (red) and a position vector ($\boldsymbol{p} \in \mathbb{R}^3$) to a point on an 11×11 grid where measurements of the generated magnetic field were taken for fitting of a magnetic field map.

where μ_0 is the permeability of free space. Substituting $\boldsymbol{\mu} = \mu\hat{\boldsymbol{x}}$ and $\boldsymbol{p} = [x, y, z]^T$ gives, after simplifying,

$$
\frac{\mu_0}{4\pi} \int \boldsymbol{B}_1(\boldsymbol{p}, \boldsymbol{\mu}) \cdot dx = -\frac{\mu_0}{4\pi} \cdot \frac{x}{\|\boldsymbol{p}\|} = \tilde{B}_k \cdot \frac{\partial}{\partial x} \frac{1}{\|\boldsymbol{p}\|}.
$$

The point dipole model of the field is sufficiently accurate for positions far away from the field source, but results in large errors close to it. For a more accurate description of the field, we incorporate the quadrupole, octupole, etc, terms from Eqn. (C.9), and will henceforth be called the multipole model of the magnetic field. The values of $\tilde{B}_k$ are unknown constants which will be found by measurement informed linear least squares regression. BigMag has two types of coils: air-cored and iron-cored. The fitting procedure is performed for each type.

C.2 Fitting the Multipole Model

Due to azimuthal symmetry of the field generated by each of the electromagnets in BigMag, it is sufficient to perform measurements in a local plane in front of the electromagnet. Figure C.2 shows a grid on the local xy-plane for $x \in [5, 15]$ cm and $y \in [-5, 5]$ cm. Let $\boldsymbol{b}_{ij} \equiv \boldsymbol{b}(\boldsymbol{p}_{ij}, I_f) \in \mathbb{R}^3$ denote a field measurement at position $\boldsymbol{p}_{ij} = [x_i, y_j, 0]^T$, $i = 1, \ldots, m$ and $j = 1, \ldots, n$, at a fitting current of I_f A. To avoid model errors caused by round-off errors from field measurement equipment, measurements were performed at a fitting current of $I_f = 4$ A. The multipole model in Eqn. (C.9) is expanded to a form suitable for linear least squares regression (in the paper, $\tilde{B}_k = a_k$),

$$
\begin{aligned}
\mathcal{V}(\boldsymbol{p}) &= \sum_{k=1}^{N} \tilde{B}_k \left(\frac{\partial^k}{\partial x^k} \frac{1}{\|\boldsymbol{p}\|} \right) \\
&= \tilde{B}_1 \cdot \mathcal{V}_1 + \ldots + \tilde{B}_N \cdot \mathcal{V}_N,
\end{aligned}
\tag{C.10}
$$

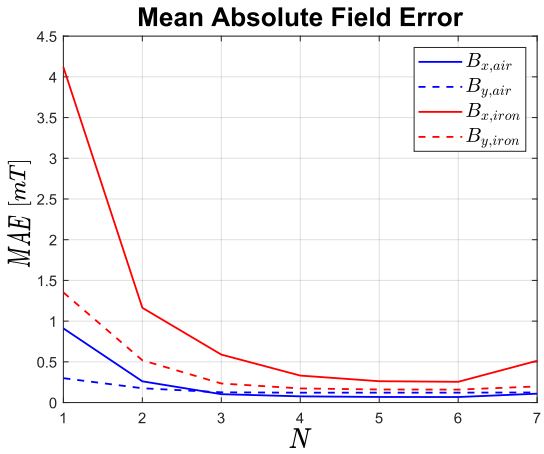

Figure C.3: Mean absolute error in milliTesla for x- and y-components of the magnetic field (B_x, B_y), for air-cored and iron-cored electromagnets of BigMag, against the order (N) of the Cartesian multipole expansion (C.10). The point-dipole model corresponds to $N = 1$.

from which the expanded multipole model follows,

$$\boldsymbol{B}(\boldsymbol{p}, I_f) = \tilde{B}_1 \cdot \nabla \mathcal{V}_1 + \ldots + \tilde{B}_N \cdot \nabla \mathcal{V}_N. \qquad \text{(C.11)}$$

The coefficients are thereafter computed from

$$\begin{bmatrix} \tilde{B}_1 \\ \vdots \\ \tilde{B}_N \end{bmatrix} = \begin{bmatrix} \nabla \mathcal{V}_1(\boldsymbol{p}_{11}) & \ldots & \nabla \mathcal{V}_N(\boldsymbol{p}_{11}) \\ & \vdots & \\ \nabla \mathcal{V}_1(\boldsymbol{p}_{m1}) & \ldots & \nabla \mathcal{V}_N(\boldsymbol{p}_{m1}) \\ \nabla \mathcal{V}_1(\boldsymbol{p}_{12}) & \ldots & \nabla \mathcal{V}_N(\boldsymbol{p}_{12}) \\ & \vdots & \\ \nabla \mathcal{V}_1(\boldsymbol{p}_{mn}) & \ldots & \nabla \mathcal{V}_N(\boldsymbol{p}_{mn}) \end{bmatrix}^{\dagger} \begin{bmatrix} \boldsymbol{b}_{11} \\ \vdots \\ \boldsymbol{b}_{m1} \\ \boldsymbol{b}_{12} \\ \vdots \\ \boldsymbol{b}_{mn} \end{bmatrix}. \qquad \text{(C.12)}$$

The multipole model order (N) was determined from the mean absolute error (MAE) between the measurements and the model predictions, defined as

$$MAE = \frac{1}{m \cdot n} \sum_{i=1}^{m} \sum_{j=1}^{n} |\boldsymbol{b}_{ij} - \boldsymbol{B}_{ij}|. \qquad \text{(C.13)}$$

The MAE in milliTesla for the x- and y-components of the magnetic field (B_x and B_y), for both the air-cored and iron-cored electromagnets in BigMag, against the multipole model order (N) are shown in Fig. C.3. The MAE is maximum at $N = 1$, which only includes the dipole term, and minimum at $N = 6$ for both types of electromagnets.

A 3D visual comparison of the measured and modelled field for the air-cored electromagnets in BigMag is shown in Fig. C.4. The xy-plane covers the measurement area ($x \in [5, 15]$ and $y \in [-5, 5]$ cm). The z-axis

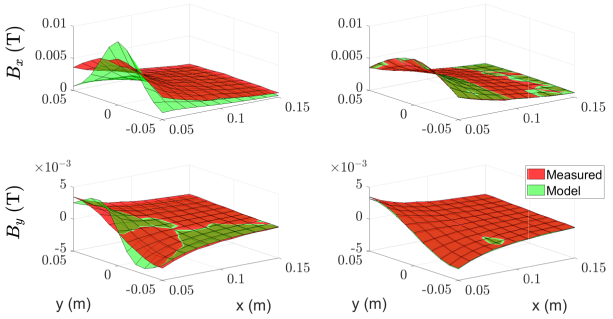

Figure C.4: Surface plots of the x- and y-components of the magnetic field (B_x and B_y) obtained from measurements (red) and model (green) of the magnetic fields in the xy-plane of an air-cored electromagnet body frame of BigMag, comparing a least-squares fit of the point dipole model (left column) to our field model with $N = 6$ (right column).

shows the xy-components of the field ($\boldsymbol{B}(\boldsymbol{p}_{ij}, I_f)$) in Tesla. The red and green surfaces represent the measurement and model, respectively. The green surface in the left column is a result of a least squares fit of the point dipole model. The green surface in the right column is a result of a least squares fit of a sixth order multipole model.

The multipole model in Fig. C.4 is fit on measurements conducted at a fitting current $I_f = 4$ A, giving the multipole field model $\boldsymbol{B}(\boldsymbol{p}, I_f)$. Instead we want a unit-current description of the magnetic field ($\beta(\boldsymbol{p}) \in \mathbb{R}^3$) and spatial magnetic field gradients ($\beta_\nabla(\boldsymbol{p}) \in \mathbb{R}^{3\times3}$), obtained from

$$\begin{aligned} \beta(\boldsymbol{p}) &= \frac{1}{I_f} \cdot \boldsymbol{B}(\boldsymbol{p}, I_f), \\ \beta_\nabla(\boldsymbol{p}) &= \beta(\boldsymbol{p}) \nabla^T, \end{aligned} \qquad \text{(C.14)}$$

such that the magnetic field and spatial magnetic field gradients of an electromagnet running a current I is given by

$$\begin{aligned} \boldsymbol{B}(\boldsymbol{p}, I) &= \beta(\boldsymbol{p}) \cdot I, \\ \boldsymbol{B}_\nabla(\boldsymbol{p}, I) &= \beta_\nabla(\boldsymbol{p}) \cdot I. \end{aligned}$$

On the following page an example of the fitting algorithm in Matlab R2019b is given; appropriate ordering of measurement data for linear least squares regression not included. The Matlab Coder toolbox was used to export function handles for $\beta(\boldsymbol{p})$ and $\beta_\nabla(\boldsymbol{p})$ to C++.

C.3 Fitting Matlab Script

```
%% Load Measurement Data
load('dat_Field_Measurements'); % b, vector of field measurements -- b = [b_11 ; b_12; ...; b_1n; b_21; ...; b_mn]  unit: Tesla
load('dat_x_positions'); % X, vector of x-positions, X(1) corresponds to b_11, X(2) to b_12, etc. unit: meter
load('dat_y_positions'); % Y, vector of y-positions unit: meter
load('dat_z_positions'); % Z, vector of z-positions unit: meter
I_f = 4; % Fitting current, unit: ampere
a = 0;
b = 0;
c = 0;

%% Fitting
%{ Make a symbolic position vector and define a monopole scalar potential term V. For the chosen order differentiate V and store the symbolic gradients in the columns of matrix B. Make a matlab function to compute BB numerically. %}
syms x y z
p = [x;y;z];
N = 6;
V = 1/((x+a)^2+(y+b)^2+(z+c)^2)^(1/2);
B = NaN(3,N);
for i = 1:N
    V = diff(V,x); % Note -- if local magnetization axis is y (fig. 2.1) interchange x with y
    B(:,i) = gradient(V,p);
end
B_func = matlabFunction(B);

%{ Fill in the regression matrix and compute the coefficients 'a' with LLSQ. %}
regression_matrix = [];
for i = 1:3:size(b,1)
    px = X(i);
    py = Y(i);
    pz = 0;
    regression_matrix = [regression_matrix; B_func(px,py,pz)];
end
a = pinv(regression_matrix)*b;

%{ Compute symbolic unit-current field map and gradient tensor %}
beta = (1/I_f)*(B*a);

beta_x = beta(1);  beta_nabla_x = gradient(beta_x,p);
beta_y = beta(2);  beta_nabla_y = gradient(beta_y,p);
beta_z = beta(3);  beta_nabla_z = gradient(beta_z,p);

beta_nabla = [beta_nabla_x'; beta_nabla_y'; beta_nabla_z'];

%{ Export the symbolic expressions for field and gradient to file %}
matlabFunction(beta, 'File', 'Beta', 'Vars', {p});
matlabFunction(beta_nabla, 'File', 'Beta_nabla', 'Vars', {p});
```

Appendix D Cosserat Rod Theory

In this part of the appendix we describe Cosserat rod theory. Cosserat rod theory is a continuum mechanics representation of the continuum manipulator and is used to compute the statics [31]. We first take a look at the governing equations of Cosserat Rod theory, including kinematics, balance laws, and constitutive relations. This is followed by an implementation of the theory in solving the statics of a discretely magnetized continuum manipulator.

D.1 Directors

The reference configuration of a manipulator may be a region surrounding a curve $s \in [s_1, s_2] \in \mathbb{R}$ described by the base curve $\boldsymbol{p}(s) \in \mathbb{R}^3$. In our case s_1 may be zero and s_2 equal to the rod length. The parameter s of the manipulator axis identifies the material cross section. In addition, each material point described by $\boldsymbol{p}(s)$ has a right-handed orthonormal material frame attached to it, described by unit vectors $\boldsymbol{d}_i(s) \in \mathbb{R}^3$, $i \in \{1, 2, 3\}$, called directors [31].

The motion of a Cosserat rod is defined by 3 vector functions

$$\boldsymbol{p}(s,t), \qquad \boldsymbol{d}_1(s,t), \qquad \boldsymbol{d}_2(s,t), \tag{D.1}$$

and by definition of an orthonormal coordinate frame a third director $\boldsymbol{d}_3$ is defined as

$$\boldsymbol{d}_3 = \boldsymbol{d}_1 \times \boldsymbol{d}_3, \tag{D.2}$$

defined in the axial direction of the Cosserat rod.

The triad $\{\boldsymbol{d}_1(s,t), \boldsymbol{d}_2(s,t), \boldsymbol{d}_3(s,t)\}$ form a right handed orthonormal basis at each material centerline point $\boldsymbol{p}(s,t)$. The directors are able to rotate along s. In addition, note that the material frame $\{\boldsymbol{d}_i\}$ is defined externally to $\boldsymbol{p}(s,t)$. This means that information about curvature and stretch of the manipulator is contained in the description of the material frame, and is thus independent of the centerline curve [31].

D.2 Kinematics

The kinematics of a Cosserat rod describe the evolution of the centerline curve $\boldsymbol{p}(s,t)$ and its associated material frame $\{\boldsymbol{d}_i(s,t)\}$. Their evolution along s is given by the respective rates of change with respect to s, denoted by $\boldsymbol{p}' \in \mathbb{R}^3$ and $\boldsymbol{d}_i' \in \mathbb{R}^3$. The explicit dependence on s is omitted for brevity.

The evolution of the directors ($\boldsymbol{d}_i'$) is used to describe material extension and material bending. To do so, vectors $\boldsymbol{v}(s) \in \mathbb{R}^3$ and $\boldsymbol{u}(s) \in \mathbb{R}^3$ are defined the represent the amount of extension/bending along/about the directors, and are therefore expressed in terms of the directors

$$\begin{aligned} \boldsymbol{v} &= v_1\boldsymbol{d}_1 + v_2\boldsymbol{d}_2 + v_3\boldsymbol{d}_3, \\ \boldsymbol{u} &= u_1\boldsymbol{d}_1 + u_2\boldsymbol{d}_2 + u_3\boldsymbol{d}_3. \end{aligned} \tag{D.3}$$

These vectors ($\boldsymbol{v}$ and $\boldsymbol{u}$) are also called the strain variables, which will become clear shortly. Because the strain variables describe the material extension and bending, they are used to describe the Cosserat rod kinematic equations as

$$\begin{aligned} \boldsymbol{v}(s) &\equiv \boldsymbol{r}'(s), \\ \boldsymbol{d}_i(s)' &= \boldsymbol{u}(s) \times \boldsymbol{d}_i(s), \end{aligned} \tag{D.4}$$

where $\boldsymbol{r}'(s) \equiv \boldsymbol{p}'(s)$ expressed in the local director frame $\{\boldsymbol{d}_i\}$. Substituting the definition of $\boldsymbol{u}$ from Eqn. D.3 expands the second expression to

$$\boldsymbol{d}_i' = u_1(\boldsymbol{d}_1 \times \boldsymbol{d}_i) + u_2(\boldsymbol{d}_2 \times \boldsymbol{d}_i) + u_3(\boldsymbol{d}_3 \times \boldsymbol{d}_i), \tag{D.5}$$

such that

$$\begin{aligned} \boldsymbol{d}_1' &= -u_2\boldsymbol{d}_3 + u_3\boldsymbol{d}_2, \\ \boldsymbol{d}_2' &= u_1\boldsymbol{d}_3 - u_3\boldsymbol{d}_1, \\ \boldsymbol{d}_3' &= -u_1\boldsymbol{d}_2 + u_3\boldsymbol{d}_1. \end{aligned} \tag{D.6}$$

The directors and derivatives in Eqn. (D.6) are shown in Fig. D.1. The directors $\{\boldsymbol{d}_1, \boldsymbol{d}_2\}$ are oriented parallel to the material cross-section, and $\boldsymbol{d}_3$ is perpendicular to the material cross-section. The values of u_1 and u_2 determine the amount of rod bending about $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, respectively. The value of u_3 determines the amount of rod torsion about $\boldsymbol{d}_3$. From the definition of $\boldsymbol{v}$ in Eqn (D.3) the values of v_1 and v_2 determine the shear of the rod material along $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$, respectively. The value of v_3 determines the strain of the rod material along $\boldsymbol{d}_3$. Hence why $\boldsymbol{v}(s)$ and $\boldsymbol{u}(s)$ are called the strain variables [31].

The cause of material strain and bending are internal forces and moments, which are caused by external forces and torques. Their relation are formulated in the balance laws.

D.3 Balance Laws

The balance laws of a Cosserat rod describe the force and torque balance attributed to a material cross-section of infinitesimal thickness $s \in [s^-, s^+]$. Let $\boldsymbol{n}(s) \in \mathbb{R}^3$ and $\boldsymbol{m}(s) \in \mathbb{R}^3$ denote the internal force and bending moment, respectively, at a material cross-section characterized by centerline parameter s. Later

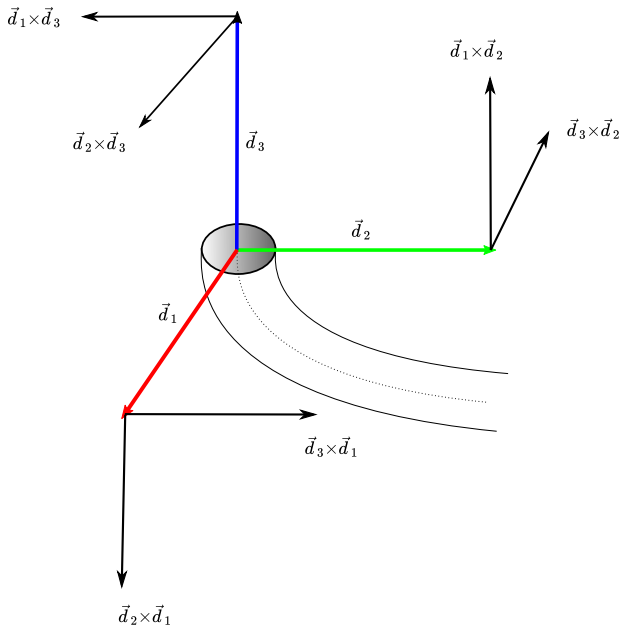

Figure D.1: Any material cross-section along the body of a continuum manipulator is associated with a local material frame defined with the triad $\{\boldsymbol{d}_1, \boldsymbol{d}_2, \boldsymbol{d}_3\}$, called the directors. The directors are orthonormal vectors. The directors $\boldsymbol{d}_1 \in \mathbb{R}^3$ and $\boldsymbol{d}_2 \in \mathbb{R}^3$ represent the principal axes of the material cross section. The director $\boldsymbol{d}_3 \in \mathbb{R}^3$ is directer perpendicular to the material cross-section and represents the local axial direction of the continuum manipulator. Each director can be expressed as a cross-product of the others [31]

on we will see that $\boldsymbol{n}(s)$ and $\boldsymbol{m}(s)$ determine the strain variables $\boldsymbol{v}(s)$ and $\boldsymbol{u}(s)$, repsectively. It is therefore important to know how $\boldsymbol{n}(s)$ and $\boldsymbol{m}(s)$ evolve along s, that is we wish to know $\boldsymbol{n}'$ and $\boldsymbol{m}'$.

Consider a rod cross-section at centerline position s. The proximal side of this cross-section is at position s^- and the distal side at position s^+. The internal forces and bending moments on each side are therefore noted as $\{\boldsymbol{n}(s^-), \boldsymbol{m}(s^-)\}$ and $\{\boldsymbol{n}(s^+), \boldsymbol{m}(s^+)\}$. In addition, consider an external acting distributed force $\boldsymbol{f}(s) \in \mathbb{R}^3$ and torque $\boldsymbol{\tau}(s) \in \mathbb{R}^3$ acting on the cross-section, as shown in Fig. D.2. The equilibrium equations are then given by

$$\begin{aligned}
&\boldsymbol{n}(s^+) - \boldsymbol{n}(s^-) + \int_{s^-}^{s^+} \boldsymbol{f}(s)ds = 0,\\
&\boldsymbol{m}(s^+) - \boldsymbol{m}(s^-) + \boldsymbol{p}(s) \times \boldsymbol{n}(s^+) - \boldsymbol{p}(s) \times \boldsymbol{n}(s^-) +\\
&\int_{s^-}^{s^+} \boldsymbol{p}(s) \times \boldsymbol{f}(s) + \boldsymbol{\tau}(s)ds = 0.
\end{aligned}$$

Differentiating the above equilibrium equations with

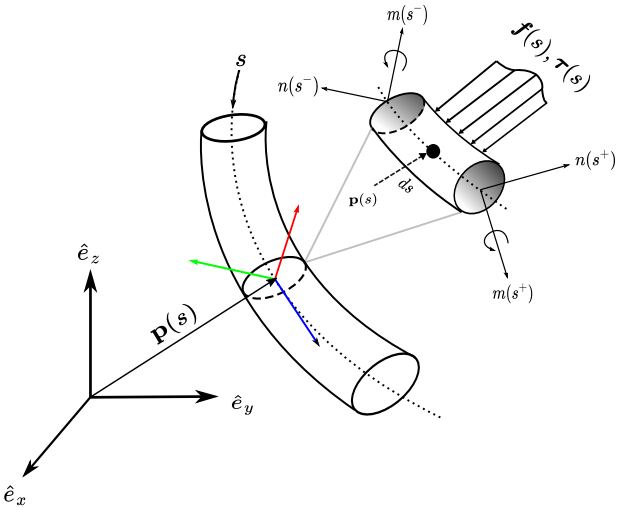

Figure D.2: A segment of a continuum manipulator with a centerline characterized by centerline parameter $s \in \mathbb{R}$. A material cross-section at position $\boldsymbol{p}(s) \in \mathbb{R}^3$ is visualized as a subsegment of the continuum manipulator with infinitesimal length ds. The proximal and distal cross-sectional area of the subsegment occurs at s^- and s^+, respectively. The subsegment is subject to an external force ($\boldsymbol{f}(s) \in \mathbb{R}^3$) and torque ($\boldsymbol{\tau}(s) \in \mathbb{R}^3$), and internal forces ($\boldsymbol{n}(s) \in \mathbb{R}^3$) and moments $\boldsymbol{m}(s) \in \mathbb{R}^3$ at its proximal and distal end from prior and subsequent infinitesimal subsegments of the continuum manipulator.

respect to s gives

$$\begin{aligned}
\boldsymbol{n}'(s) + \boldsymbol{f}(s) = 0,\\
\boldsymbol{m}'(s) + \boldsymbol{p}(s)' \times \boldsymbol{n}(s) +\\
\boldsymbol{p}(s) \times \boldsymbol{n}'(s) + \boldsymbol{p}(s) \times \boldsymbol{f}(s) + \boldsymbol{\tau}(s) = 0,
\end{aligned}$$

where

$$\begin{aligned}
\boldsymbol{n}'(s) &= \frac{d}{ds}(\boldsymbol{n}(s^+) - \boldsymbol{n}(s^-)),\\
\boldsymbol{m}'(s) &= \frac{d}{ds}(\boldsymbol{m}(s^+) - \boldsymbol{m}(s^-)).
\end{aligned}$$

Substituting $\boldsymbol{f}(s) = -\boldsymbol{n}'(s)$ simplifies to the final balance equations to

$$\begin{aligned}
\boldsymbol{n}'(s) + \boldsymbol{f}(s) = 0\\
\boldsymbol{m}'(s) + \boldsymbol{p}(s)' \times \boldsymbol{n}(s) + \boldsymbol{\tau}(s) = 0,
\end{aligned} \tag{D.7}$$

for a Cosserat rod in static equilibrium [31]. Now that the evolution of internal forces and moments along the Cosserat rod is known, we want to know how these quantities relate to the strain variables, which determine material extension and bending. These relations are called the constitutive relations.

D.4 Constitutive Relations

The Cosserat rod constitutive relations are a set of equations that relate the internal force and bending moments $\boldsymbol{n}(s)$ and $\boldsymbol{m}(s)$ to the strain variables $\boldsymbol{v}(s)$ and $\boldsymbol{u}(s)$. Because the definitions are similiar, e.g. $m_i(s)$ a bending moment about $\boldsymbol{d}_i(s)$ and $u_i(s)$ the evolution $\boldsymbol{d}_i'(s)$, the easiest choice is a linear, diagonal, relationship

$$\begin{aligned} m_i(s) &= K_{b,i} \cdot (u_i(s) - \hat{u}_i(s)), \\ n_i(s) &= K_{s,i} \cdot (v_i(s) - \hat{v}_i(s)), \end{aligned} \tag{D.8}$$

where $K_{b,i}, K_{s,i} \in \mathbb{R}^+$ represent constant bending and shear stiffnesses (they can be dependent on s) [26].

D.5 Model Equations

The previous sections introduced the notation used in Cosserat rod theory with directors, balance laws, and constitutive relations expressed in a local material frame at position $\boldsymbol{p}(s)$. In this section we translate the notation to a representation with respect to a reference coordinate frame.

As we have seen before, the centerline curve is characterized by $\boldsymbol{p}(s) \in \mathbb{R}^3$ with an attached material frame

$$\boldsymbol{R}(s) = \begin{bmatrix} \boldsymbol{d}_1(s) & \boldsymbol{d}_2(s) & \boldsymbol{d}_3(s) \end{bmatrix} \in SO(3).$$

The Cosserat rod can therefore be described by a homogeneous transformation $\boldsymbol{H}(s) \in SE(3)$

$$\boldsymbol{H}(s) = \begin{bmatrix} \boldsymbol{R}(s) & \boldsymbol{p}(s) \\ \boldsymbol{0}^T & 1 \end{bmatrix}. \tag{D.9}$$

The evolution of $\boldsymbol{H}(s)$ along s is again defined by the strain variables

$$\boldsymbol{R}'(s) = \boldsymbol{R}(s)[\boldsymbol{u}(s)]_\times, \qquad \boldsymbol{p}'(s) = \boldsymbol{R}(s)\boldsymbol{v}(s), \tag{D.10}$$

with $[\boldsymbol{u}(s)]_\times \in \mathbb{R}^{3\times 3}$ a skew-symmetric matrix representation of $\boldsymbol{u}(s)$. Similarly the global representation of the internal force and bending moment is obtained from the constitutive relations

$$\begin{aligned} \boldsymbol{n}(s) &= \boldsymbol{R}(s)\boldsymbol{K}_s(s)(\boldsymbol{v}(s) - \hat{\boldsymbol{v}}(s)), \\ \boldsymbol{m}(s) &= \boldsymbol{R}(s)\boldsymbol{K}_b(s)(\boldsymbol{u}(s) - \hat{\boldsymbol{u}}(s)), \end{aligned} \tag{D.11}$$

with

$$\begin{aligned} \boldsymbol{K}_s(s) &= diag(GA(s), GA(s), EA(s)), \\ \boldsymbol{K}_b(s) &= diag(EI_x(s), EI_y(s), GI_z(s)), \end{aligned} \tag{D.12}$$

where $A(s) \in \mathbb{R}$ is the cross-sectional area of the rod, $E(s) \in \mathbb{R}$ the material elasticity modulus, $G(s) \in \mathbb{R}$ the material shear modulus, $I_x \in \mathbb{R}$ and $I_y \in \mathbb{R}$ the second moments of area of the cross-section about the principal axes, and $I_z \in \mathbb{R}$ the polar moment of inertia of the cross-section about $\boldsymbol{d}_3$. In addition $\hat{\boldsymbol{v}}(s)$ and $\hat{\boldsymbol{u}}(s)$ are the reference strain variables. For an unstressed reference state of the rod, the reference strains satisfy

$$\hat{\boldsymbol{v}} = [0, 0, 1]^T, \qquad \hat{\boldsymbol{u}} = [0, 0, 0]^T,$$

such that the unextended manipulator length L is unchanged,

$$\int_0^L \boldsymbol{v}(s) ds = [0, 0, L]^T, \qquad \boldsymbol{v}(s) = \hat{\boldsymbol{v}}(s).$$

Combining the kinematic equations in Eqn. (D.10), balance laws in Eqn. (D.7), and constitutive relations in Eqn. (D.11), gives the set of Cosserat ordinary differential equations (ODEs),

$$\begin{aligned} \boldsymbol{p}' &= \boldsymbol{R}\boldsymbol{v}, \qquad \boldsymbol{v} = \boldsymbol{K}_s^{-1}\boldsymbol{R}^T\boldsymbol{n} + \hat{\boldsymbol{v}}, \\ \boldsymbol{R}' &= \boldsymbol{R}[\boldsymbol{u}]_\times, \qquad \boldsymbol{u} = \boldsymbol{K}_b^{-1}\boldsymbol{R}^T\boldsymbol{m} + \hat{\boldsymbol{u}}, \\ \boldsymbol{n}' &= -\boldsymbol{f}, \\ \boldsymbol{m}' &= -\boldsymbol{p}' \times \boldsymbol{n} - \boldsymbol{\tau}, \end{aligned} \tag{D.13}$$

where the explicit dependence on s is omitted for brevity [26].

We now want to rewrite Eqn. (D.13) to an orientation quaternion representation, because the parameterization of $\boldsymbol{R}(s)$ in Eqn. (D.13) is subject to singularities. We define an orientation quaternion $\boldsymbol{q}(s)=(q_r, \boldsymbol{q}_i) \in \mathbb{H}$, with $\boldsymbol{q}_i=q_i\boldsymbol{i}+q_j\boldsymbol{j}+q_k\boldsymbol{k}$ ($\|\boldsymbol{q}\|=1$). The Cosserat ODEs in the quaternion representation contain $\boldsymbol{q}'(s)$ instead of $\boldsymbol{R}'(s)$. To find the expression for $\boldsymbol{q}'(s)$, first consider the time derivative of an orientation quaternion

$$\dot{\boldsymbol{q}} = \frac{1}{2} \begin{bmatrix} -\boldsymbol{q}_i^T \\ q_r\mathcal{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix} \boldsymbol{\omega},$$

with $\boldsymbol{\omega} \in \mathbb{R}^3$ an angular velocity vector represented in a global coordinate frame (Appendix F.3.7). The spatial alternative to $\boldsymbol{\omega}$ is the transformed strain variable $\boldsymbol{R}(\boldsymbol{q})\boldsymbol{u}$. The expression for $\boldsymbol{q}'$ is therefore given by

$$\boldsymbol{q}' = \frac{1}{2} \begin{bmatrix} -\boldsymbol{q}_i^T \\ q_r\boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix} \boldsymbol{R}(\boldsymbol{q})\boldsymbol{u}, \tag{D.14}$$

where $\boldsymbol{R}(\boldsymbol{q}) \in SO(3)$ is given by Eqn. (F.16). The Cosserat ODEs in Eqn. (D.13), in quaternion repre-

sentation [27], are then given by

$$\begin{aligned}
\boldsymbol{p}' &= \boldsymbol{R}(\boldsymbol{q})\boldsymbol{v}, \\
\boldsymbol{v} &= \boldsymbol{K}_s^{-1}\boldsymbol{R}(\boldsymbol{q})^T\boldsymbol{n} + \hat{\boldsymbol{v}}, \\
\boldsymbol{q}' &= \frac{1}{2}\begin{bmatrix} -\boldsymbol{q}_i^T \\ q_r\boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix}\boldsymbol{R}(\boldsymbol{q})\boldsymbol{u}, \\
\boldsymbol{u} &= \boldsymbol{K}_b^{-1}\boldsymbol{R}(\boldsymbol{q})^T\boldsymbol{m} + \hat{\boldsymbol{u}}, \\
\boldsymbol{n}' &= -\boldsymbol{f}, \\
\boldsymbol{m}' &= -\boldsymbol{p}' \times \boldsymbol{n} - \boldsymbol{\tau},
\end{aligned} \tag{D.15}$$

which are combined in a Cosserat rod state gradient vector $\boldsymbol{y}'(s)$,

$$\boldsymbol{y}'(s) = \begin{bmatrix} \boldsymbol{p}'(s)^T & \boldsymbol{q}'(s)^T & \boldsymbol{n}'(s)^T & \boldsymbol{m}'(s)^T \end{bmatrix}^T \in \mathbb{R}^{13}, \tag{D.16}$$

later used to solve the statics of the Cosserat rod.

D.6 Statics Solution

The Cosserat rod, from here on out called the continuum manipulator (CM), deforms as a result of external distributed forces and torques acting on its body. In this section we want to find the CM shape when in static equilibrium, using the Cosserat rod ODEs in Eqn. (D.15).

The CM statics are solved as a boundary value problem (BVP). We consider the manipulator material state parameters $\boldsymbol{y}(s) \in \mathbb{R}^{13}$, $s \in [0, L]$, as

$$\boldsymbol{y}(s) = \begin{bmatrix} \boldsymbol{p}(s)^T & \boldsymbol{q}(s)^T & \boldsymbol{n}(s)^T & \boldsymbol{m}(s)^T \end{bmatrix}^T \in \mathbb{R}^{13}, \tag{D.17}$$

whose evolution along s was described by the set of thirteen ODEs in Eqn. (D.16). From this point onward we adopt the notation $\boldsymbol{p}_s \equiv \boldsymbol{p}(s)$. The BVP includes a proximal ($s = 0$) and distal ($s = L$) boundary condition, denoted as $\mathbb{P}(\boldsymbol{y}_0)$ and $\mathbb{D}(\boldsymbol{y}_L)$, where $\mathbb{P}$ and $\mathbb{D}$ are sets that contains a subset of the CM state parameters that depend on the CM configuration.

A schematic of a proximally-fixed and distally-free CM carrying two permanent magnets at discrete positions along s is shown in Fig. D.3 (also found in the paper). The proximal end of the CM, due to being fixed, has a constant position and orientation, and therefore the proximal boundary condition is given as

$$\mathbb{P}(\boldsymbol{y}_0) = \{\boldsymbol{p}_0, \boldsymbol{q}_0\}.$$

Similarly, the distal end of the CM, due to being free floating, has a constant internal force and moment, and therefore the distal boundary condition is given as

$$\mathbb{P}(\boldsymbol{y}_L) = \{\boldsymbol{n}_L, \boldsymbol{m}_L\} = \{\boldsymbol{0}, \boldsymbol{0}\}.$$

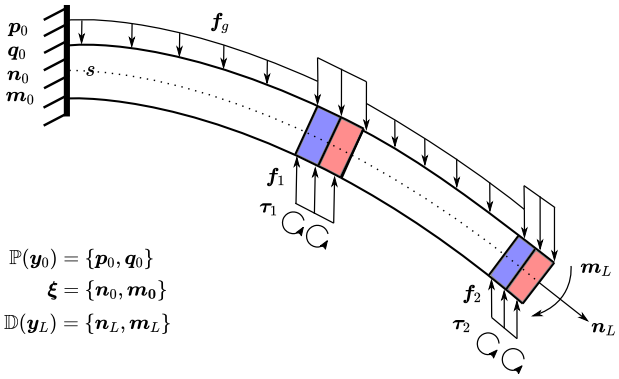

Figure D.3: Discretely magnetized continuum manipulator subject to a distributed gravitational force $\boldsymbol{f}_g$ along its full length, and distributed magnetic torques $\boldsymbol{\tau}_m$ and forces $\boldsymbol{f}_m$ along its magnets, where $m = 1, 2$ denotes the magnet index. The manipulator shape is described by position $\boldsymbol{p}(s)$, orientation quaternion $\boldsymbol{q}(s)$, internal force $\boldsymbol{n}(s)$, and internal moment $\boldsymbol{m}(s)$, grouped into a state vector $\boldsymbol{y}(s) \equiv \boldsymbol{y}_s$, where $s \in [0, L]$ is the centerline parameter. The manipulator is proximally fixed, with a constant proximal boundary condition $\mathbb{P}(\boldsymbol{y}_0)$, and distally free with distal boundary condition $\mathbb{D}(\boldsymbol{y}_L)$. Unknown proximal material state parameters are grouped into a vector $\boldsymbol{\xi}$ and optimized with convex optimization and forward integration.

The CM body is subject to a distributed gravitational force $\boldsymbol{f}_g \in \mathbb{R}^3$, and distributed magnetically exerted forces $\boldsymbol{f}_m \in \mathbb{R}^3$ and torques $\boldsymbol{\tau}_m \in \mathbb{R}^3$, where $m = 1, 2$ denotes the magnet index as counted from the proximal end of the CM.

With the above defined proximal and distal boundary conditions we can solve the BVP. The method we use to find a solution is forward integration with an explicit Runge-Kutta fourth order method and convex optimization with Levenberg-Marquardt. These algorithms are discussed below.

D.6.1 Forward Integration

Consider that the CM in Fig. D.3 is divided into $N \in \mathbb{N}$ subsections, and let $\boldsymbol{y}_N$ be its distal state parameters (different from $\boldsymbol{y}_L$). The forward integration method computes $\boldsymbol{y}_N$ from guessed proximal states $\boldsymbol{y}_0$. In Fig. D.3 the proximal position and orientation are known, but the internal force and moment must be guessed. The guessed proximal states are contained in the vector $\boldsymbol{\xi} = [\boldsymbol{n}_0^T, \boldsymbol{m}_0^T]^T \in \mathbb{R}^6$. For forward integration we must first properly discretize the CM.

Figure D.4 shows a discretized version of the CM. For a nonuniform CM we must also choose nonuniform discretizations such that material transitions, that is flexible-rigid and rigid-flexible, are fully cap-

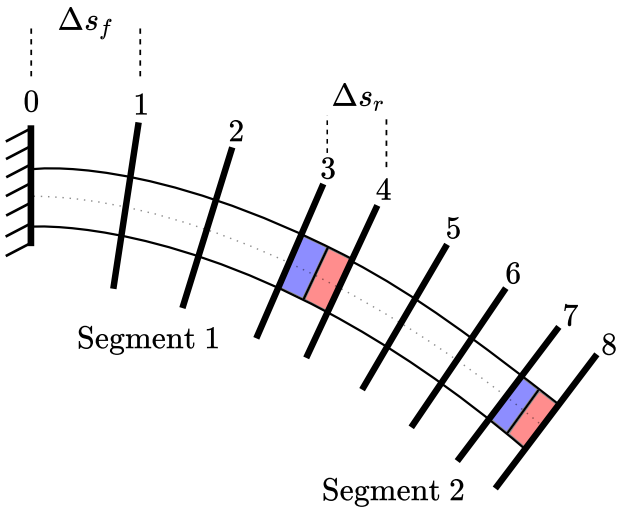

Figure D.4: A discretely magnetized continuum manipulators can be considered as a series of segments with a tip magnet. A solution to the statics of a continuum manipulator (with Cosserat rod theory) is found with forward integration. Forward integration requires discretization of the manipulator centerline ($s \in [0, L]$) into smaller parts Δs. To capture the transition between materials during forward integration, discretization of flexible nonmagnetic and rigid magnetic parts of a subsegment must be discretized separately as Δs_f and Δs_r, respectively.

tured. This is necessary such that forward integration fully covers each flexible and rigid part. To do so, first we split the CM into two segments, each containing proximal flexible and distal rigid (magnet) material. Then we define ${}^{i}L_f \in \mathbb{R}^+$ and ${}^{i}L_r \in \mathbb{R}^+$ as their respective lengths for segment $i = 1, 2$. We discretize each length into ${}^{i}N_f \in \mathbb{N}_1$ and ${}^{i}N_r \in \mathbb{N}_1$ parts. Finally, integration steps are defined as

$$ {}^{i}\Delta s = \begin{cases} {}^{i}\Delta s_f = \frac{{}^{i}L_f}{{}^{i}N_f} & \text{If flexible} \\ {}^{i}\Delta s_r = \frac{{}^{i}L_r}{{}^{i}N_r} & \text{If rigid} \end{cases}. $$

Forward integration is performed with an explicit Runge-Kutta fourth order method. This method is a family of formulae that advances a state $\boldsymbol{y}_n$ to $\boldsymbol{y}_{n+1}$, using a step size Δs and state gradient $\boldsymbol{y}'(s, \boldsymbol{y})$

$$ \boldsymbol{y}_{n+1} = \boldsymbol{y}_n + \frac{1}{6}\left(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4\right), $$

where the definitions of $\boldsymbol{k}_j$, $j \in \{1, 2, 3, 4\}$, is given by

$$ \begin{aligned} \boldsymbol{k}_1 &= \Delta s \cdot \boldsymbol{y}'(s_n, \boldsymbol{y}_n), \\ \boldsymbol{k}_2 &= \Delta s \cdot \boldsymbol{y}'(s_n + \frac{\Delta s}{2}, \boldsymbol{y}_n + \frac{\boldsymbol{k}_1}{2}), \\ \boldsymbol{k}_3 &= \Delta s \cdot \boldsymbol{y}'(s_n + \frac{\Delta s}{2}, \boldsymbol{y}_n + \frac{\boldsymbol{k}_2}{2}), \\ \boldsymbol{k}_4 &= \Delta s \cdot \boldsymbol{y}'(s_n + \Delta s, \boldsymbol{y}_n + \boldsymbol{k}_3). \end{aligned} $$

All $\boldsymbol{k}_j$ assume linearity of some gradient between $s \in [s_n, s_n + \Delta s]$. The value of $\boldsymbol{k}_1$ predicts the change in state parameters from the known state parameters $\boldsymbol{y}_n$. The value of $\boldsymbol{k}_2$ predicts the change in state parameters from the predicted gradient half way along Δs, based on state parameters predicted from $\boldsymbol{k}_1$. The value of $\boldsymbol{k}_3$ also predicts the change in state parameters from the predicted gradient half way along Δs, but based on state parameters predicted from $\boldsymbol{k}_2$. Finally, $\boldsymbol{k}_4$ predicts the change in state parameters from the predicted gradient at the end of Δs. Then, the final solution ($\boldsymbol{y}_N$) is the weighted average of all four predictions. A matrix of manipulator state parameters at discrete points along the manipulator with $S \in \mathbb{N}_1$ segments, after forward integration from a unique $\boldsymbol{y}_0$, is obtained

$$ \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{y}_0 & \boldsymbol{y}_1 & \dots & \boldsymbol{y}_N \end{bmatrix} \in \mathbb{R}^{13 \times N}, $$

where

$$ N = 1 + \sum_{i=1}^{S} {}^{i}N_f + {}^{i}N_r. $$

The forward integration algorithm is shown in Algorithm 2. Since we perform forward integration from the proximal end of the CM, the proximal boundary condition is automatically satisfied, but the distal boundary condition is not. To this end a residual $r \in \mathbb{R}^+$ is defined as the 2-norm of the error between the desired distal boundary conditions and distal boundary states obtained from forward integration,

$$ \begin{aligned} \boldsymbol{r} &= \mathbb{D}(\boldsymbol{y}_L) - \mathbb{D}(\boldsymbol{y}_N), \\ r &= \|\boldsymbol{r}\|. \end{aligned} $$

If the residual is large, the obtained shape ($\boldsymbol{Y}$) is not a valid solution to the CM statics. In that case we must adapt the unknown proximal state parameters ($\boldsymbol{\xi}$) and try again. We do so with convex optimization, using Levenberg-Marquardt.

D.6.2 Convex Optimization

A solution to the statics of the CM means finding the proximal state parameters in $\boldsymbol{\xi}$ for which

$$ \arg\min_{\boldsymbol{\xi}} \|\boldsymbol{r}\|. $$

In practice this means setting an arbitrary threshold $E \in \mathbb{R}^+$ and finding a solution out of a set of possible solutions, i.e.

$$ \boldsymbol{\xi} \in \{\tilde{\boldsymbol{\xi}} \mid \|\boldsymbol{r}(\tilde{\boldsymbol{\xi}})\| \leq E\}. $$

To find a suitable $\boldsymbol{\xi}$, we iteratively optimize its values with a shift vector $\Delta\boldsymbol{\xi}$ computed with Levenberg-Marquardt (LM),

$$\Delta\boldsymbol{\xi} = -\underbrace{\left(\boldsymbol{J}^T\boldsymbol{J} + \lambda \cdot \operatorname{diag}\left(\boldsymbol{J}^T\boldsymbol{J}\right)\right)^{-1}\boldsymbol{J}^T}_{\boldsymbol{J}^\dagger}\boldsymbol{r}, \quad \text{(D.18)}$$

where $\lambda \in \mathbb{R}^3$ is called the Marquardt parameter and $\boldsymbol{J} \in \mathbb{R}^{m\times n}$, when $\boldsymbol{r} \in \mathbb{R}^m$ and $\boldsymbol{\xi} \in \mathbb{R}^n$, is a numerically computed Jacobian defined as

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial r_1}{\partial \xi_1} & \cdots & \frac{\partial r_1}{\partial \xi_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial \xi_1} & \cdots & \frac{\partial r_m}{\partial \xi_n} \end{bmatrix}. \quad \text{(D.19)}$$

Eqn. (D.18) is derived in appendix E, but contains an additional minus sign. This is because not $\boldsymbol{J}$ but $-\boldsymbol{J}$ is used in the computation, reversing the gradients towards the descending direction of $\boldsymbol{r}$. The Marquardt parameter shifts Eqn. (D.18) towards a Gauss-Newton algorithm when relatively low, and towards a Gradient-Descent algorithm when relatively high,

$$\Delta\boldsymbol{\xi} = \begin{cases} -\left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\boldsymbol{J}^T\boldsymbol{r}, & (\lambda \ll 1) \\ -\frac{1}{\lambda}\boldsymbol{J}^T\boldsymbol{r}, & (\lambda \gg 1) \end{cases}.$$

The advantage of this is that the Gradient-Descent algorithm is fast when the guess $\boldsymbol{\xi}$ is far from the 'actual' solution, while the Gauss-Newton algorithm is faster when $\boldsymbol{\xi}$ is close to the actual solution. The term 'diag $\left(\boldsymbol{J}^T\boldsymbol{J}\right)$' depicts a diagonal matrix with the diagonal values of $\boldsymbol{J}^T\boldsymbol{J}$, which is an adaptation of the conventional LM method to make the inversion scale invariant [33, 34]. Furthermore, the addition of the Marquardt parameter (λ) makes the right hand side term in Eqn. (D.18) a (left) damped Moore-Penrose pseudoinverse, i.e. $\boldsymbol{J}^\dagger\boldsymbol{J} \simeq \boldsymbol{I}$.

The algorithms for the numerical computation of theJacobian and for the convex optimization with LM algorithm are given in Algorithms 3 and 4, respectively. To compute $\boldsymbol{J}$, the values $\boldsymbol{\xi}_j$, $j \in \{1, \ldots, n\}$, are iteratively incremented and decremented, the new residual $\boldsymbol{r}^+$ computed, and the gradient stored in a column of $\boldsymbol{J}$. In the LM algorithm the Jacobian is used to compute the shift vector $\Delta\boldsymbol{\xi}$ until the magnitude of the residuals $\|\boldsymbol{r}\|$ fall below the threshold E. There exists a trade-off between computation time and solution accuracy. Computation time for finding a suitable $\boldsymbol{\xi}$ is lower for higher E, but the statics solution ($\boldsymbol{Y}$) is less accurate; alternatively computation time is higher for lower E, where the statics solution is more accurate. The algorithm contains two while-loops, in the first the Marquardt parameter λ is decremented by a factor 3, in the second λ is increased by a factor 2. The first, therefore, moves the computation of the shift vector $\Delta\boldsymbol{\xi}$ towards the Gauss-Newton method, and, the latter, towards the Gradient-descent method.

Algorithm 2: Forward Integration with Runge-Kutta

Input : Proximal state parameters, $\boldsymbol{y}_0(\boldsymbol{\xi})$
Output : Shape Solution, $\boldsymbol{Y}$

$\boldsymbol{Y} = \boldsymbol{0} \in \mathbb{R}^{13\times N}$
$s = 0$
$j = 1$
for $i \leftarrow 1$ **to** S **do**
  ${}^i\Delta s_f = {}^iL_f/{}^iN_f$
  ${}^i\Delta s_r = {}^iL_r/{}^iN_r$
  /* Flexible Non-Magnetic Region */
  for $n \leftarrow 1$ **to** ${}^iN_f$ **do**
    $\boldsymbol{k}_1 = {}^i\Delta s_f \cdot \boldsymbol{y}'(s, \boldsymbol{Y}_{j-1})$
    $\vdots$
    $\boldsymbol{k}_4 = {}^i\Delta s_f \cdot \boldsymbol{y}'(s + {}^i\Delta s_f, \boldsymbol{Y}_{j-1} + \boldsymbol{k}_3)$
    $\boldsymbol{Y}_j = \boldsymbol{Y}_{j-1} + \frac{1}{6}(\boldsymbol{k}_1 + 2\boldsymbol{k}_2 + 2\boldsymbol{k}_3 + \boldsymbol{k}_4)$
    $s = s + {}^i\Delta s_f$
    $j = j + 1$
  end for
  /* Rigid Magnetic Region */
  for $m \leftarrow 1$ **to** ${}^iN_r$ **do**
    $\boldsymbol{k}_1 = {}^i\Delta s_r \cdot \boldsymbol{y}'(s, \boldsymbol{Y}_{j-1}, \boldsymbol{f}_i, \boldsymbol{\tau}_m)$
    $\vdots$
    $j = j + 1$
  end for
end for
return $\boldsymbol{Y}$

Algorithm 3: getJacobian

Input : Current residuals $\boldsymbol{r}$
Proximal Boundary Condition $\mathbb{P}(\boldsymbol{y}_0)$
Guessed states $\boldsymbol{\xi}$
Output : Jacobian $\boldsymbol{J}$

$\boldsymbol{J} = \boldsymbol{0} \in \mathbb{R}^{m \times n}$ $\delta\xi = 1e-10$
for $j \leftarrow 1$ **to** n **do**
  $\boldsymbol{\xi}_j = \boldsymbol{\xi}_j + \delta\xi$
  $\boldsymbol{y}_0 \leftarrow \mathbb{P}(\boldsymbol{y}_0), \boldsymbol{\xi}$
  $\boldsymbol{Y} = \text{RungeKutta}(\boldsymbol{y}_0)$
  $\mathbb{D}(\boldsymbol{y}_N) \leftarrow \mathbb{D}(\boldsymbol{Y}_N)$
  $\boldsymbol{r}^+ = \mathbb{D}(\boldsymbol{y}_L) - \mathbb{D}(\boldsymbol{Y}_N)$
  $\boldsymbol{J}_j = (\boldsymbol{r}^+ - \boldsymbol{r})/\delta\xi$
  $\boldsymbol{\xi}_j = \boldsymbol{\xi}_j - \delta\xi$
end for
return $\boldsymbol{J}$

Algorithm 4: Convex Optimization

Input : Proximal Boundary Condition $\mathbb{P}(\boldsymbol{y}_0)$
Initial guessed states $\boldsymbol{\xi}$
Distal Boundary Condition $\mathbb{D}(\boldsymbol{y}_L)$
Output : Statics Solution, $\boldsymbol{Y}$

$\boldsymbol{y}_0 \leftarrow \mathbb{P}(\boldsymbol{y}_0), \boldsymbol{\xi}$
$\boldsymbol{Y} = \text{RungeKutta}(\boldsymbol{y}_0)$
$\mathbb{D}(\boldsymbol{y}_N) \leftarrow \mathbb{D}(\boldsymbol{Y}_N)$
$\boldsymbol{r} = \mathbb{D}(\boldsymbol{y}_L) - \mathbb{D}(\boldsymbol{y}_N)$
$E^- = \|\boldsymbol{r}\|$
$\lambda = 1e-3$
/* Toward Gauss-Newton */
while $E^- > E$ **do**
  $\boldsymbol{J} = \text{getJacobian}(\boldsymbol{r}, \boldsymbol{\xi})$
  $\lambda = \lambda/3$
  $\Delta\boldsymbol{\xi} = \left(\boldsymbol{J}^T\boldsymbol{J} + \lambda \cdot \text{diag}\left(\boldsymbol{J}^T\boldsymbol{J}\right)\right)^{-1}\boldsymbol{J}^T\boldsymbol{r}$
  $\boldsymbol{\xi} = \boldsymbol{\xi} + \Delta\boldsymbol{\xi}$
  $\boldsymbol{y}_0 \leftarrow \mathbb{P}(\boldsymbol{y}_0), \boldsymbol{\xi}$
  $\boldsymbol{Y} = \text{RungeKutta}(\boldsymbol{y}_0)$
  $\mathbb{D}(\boldsymbol{y}_N) \leftarrow \mathbb{D}(\boldsymbol{Y}_N)$
  $\boldsymbol{r} = \mathbb{D}(\boldsymbol{y}_L) - \mathbb{D}(\boldsymbol{y}_N)$
  $E^+ = \|\boldsymbol{r}\|$
  $\boldsymbol{\xi} = \boldsymbol{\xi} - \Delta\boldsymbol{\xi}$
  if $E^+ < E^-$ **then**
    $\boldsymbol{\xi} = \boldsymbol{\xi} + \Delta\boldsymbol{\xi}$
    $E^- = E^+$
  end if
  else
    /* Toward Gradient-Descent */
    while $E^+ \geq E^-$ **do**
      $\lambda = 2 \cdot \lambda$
      $\Delta\boldsymbol{\xi} = \ldots$
      $\vdots$
      $E^+ = \|\boldsymbol{r}\|$
    end while
    $\boldsymbol{\xi} = \boldsymbol{\xi} + \Delta\boldsymbol{\xi}$
    $E^- = E^+$
  end if
end while
return $\boldsymbol{Y}$

Appendix E Nonlinear Least Squares

The problem of nonlinear least squares is to minimize the sum of squared errors between dependent variables ($\boldsymbol{y} \in \mathbb{R}^n$) and a model function ($\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{\xi})$). The model function consists of independent parameters ($\boldsymbol{x}$) nonlinear with respect to unknown parameters ($\boldsymbol{\xi} \in \mathbb{R}^m$). Considering the BVPs used for solving the continuum manipulator statics, the dependent variables ($\boldsymbol{y}$) are distal boundary conditions, independent variables ($\boldsymbol{x}$) are proximal boundary conditions, and unknown parameters ($\boldsymbol{\xi}$) contain all parameters that remain. The contrast with regular linear least squares is that iterative optimization algorithms need to be used to compute $\boldsymbol{\xi}$, opposed to simple matrix inversion.

Below we describe the theory on nonlinear least squares, which linearizes a nonlinear least squares problem using a first order Taylor expansion [34].

The residuals of a data set are defined as the error between the dependent variable $\boldsymbol{y}$ and its prediction by a model function $f(\boldsymbol{x}, \boldsymbol{\xi})$

$$\boldsymbol{r} = \boldsymbol{y} - f(\boldsymbol{x}, \boldsymbol{\xi}). \tag{E.1}$$

The sum of squared residuals S is thereafter defined as

$$S = \sum_{i=1}^{n} r_i^2, \tag{E.2}$$

with $r_i \in \boldsymbol{r}$, or equivalent $S = \boldsymbol{r}^T \boldsymbol{r}$. We have a perfect model $f(\boldsymbol{x}, \boldsymbol{\xi})$ when $S = 0$, which occurs at $\boldsymbol{\xi}$ for which the partial derivatives of S with respect to $\xi_j \in \boldsymbol{\xi}$ are zero,

$$\frac{\partial S}{\partial \xi_j} = 2\boldsymbol{r}^T \frac{\partial \boldsymbol{r}}{\partial \xi_j} = 0, \qquad j \in \{1, 2, ..., m\}. \tag{E.3}$$

For nonlinear systems, the model function ($\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{\xi})$) is a nonlinear function of its arguments. Therefore, so are the residuals ($r_i \; i \in \{1, 2, ..., n\}$) and their partial derivatives ($\frac{\partial r_i}{\partial \xi_j}$), generally. The solution to Eqn. (E.3) is most likely not an analytical one, and the parameters $\boldsymbol{\xi}$ are optimized iteratively to minimize S, according to

$$\xi_j^{k+1} = \xi_j^k + \Delta\xi_j^k,$$

with k an iteration number and $\Delta\xi_j^k$ the shift vector of the parameter ξ_j. The model function $f(\boldsymbol{x}, \boldsymbol{\xi}^{k+1})$ at iteration step k+1 can be approximated by a first-order Taylor expansion,

$$\begin{aligned} f(\boldsymbol{x}, \boldsymbol{\xi}^{k+1}) \approx & f(\boldsymbol{x}, \boldsymbol{\xi}^k) + \sum_{j=1}^{m} \frac{\partial f(\boldsymbol{x}, \boldsymbol{\xi}^k)}{\partial \xi_j} (\xi_j^{k+1} - \xi_j^k), \\ \approx & f(\boldsymbol{x}, \boldsymbol{\xi}^k) + \sum_{i=j}^{m} J_j^k \Delta\xi_j^k. \end{aligned} \tag{E.4}$$

Where the Jacobian J_j^k represents the partial derivative of $f(\boldsymbol{x}, \boldsymbol{\xi})$ with respect to $\xi_j \in \boldsymbol{\xi}$, evaluated at $\boldsymbol{x}$ and $\boldsymbol{\xi}^k$. Similarly, the residuals at each iteration can be approximated with a first-order Taylor expansion,

$$\boldsymbol{r}^{k+1} = \boldsymbol{r}^k + \sum_{j=1}^{m} \frac{\partial \boldsymbol{r}^k}{\partial \xi_j} \Delta\xi_j^k. \tag{E.5}$$

The partial derivative term in Eqn. (E.5) can be written in terms of the model function. Using Eqn. (E.1), with $\boldsymbol{y}$ considered as a constant, the partial derivative term is written as

$$\frac{\partial \boldsymbol{r}^k}{\partial \xi_j} = -\frac{\partial f(\boldsymbol{x}, \boldsymbol{\xi}^k)}{\partial \xi_j} = -J_j^k. \tag{E.6}$$

Substituting Eqn. (E.6) into Eqn. (E.5) gives the result

$$\boldsymbol{r}^{k+1} = \boldsymbol{r}^k - \sum_{i=j}^{m} J_j^k \Delta\xi_j^k. \tag{E.7}$$

We now return to Eqn. (E.3) and explicitly write this equation in terms of the iteration number k,

$$\frac{\partial S^{k+1}}{\partial \xi_j} = 2\left(\boldsymbol{r}^{k+1}\right)^T \frac{\partial \boldsymbol{r}^{k+1}}{\partial \xi_j} = 0. \tag{E.8}$$

The partial derivative term on the right hand side is written, with help from Eqns. (E.4) and (E.6), and ignoring the higher order terms, as

$$\begin{aligned} \frac{\partial \boldsymbol{r}^{k+1}}{\partial \xi_j} = & -\frac{\partial f(\boldsymbol{x}, \boldsymbol{\xi}^{k+1})}{\partial \xi_j}, \\ & \ll f(\boldsymbol{x}, \boldsymbol{\xi}^k) \\ = & -\frac{\partial}{\partial \xi_j}\Big(f(\boldsymbol{x}, \boldsymbol{\xi}^k) + \sum_{j=1}^{m} J_j^k \Delta\xi_j^k\Big), \\ \approx & -\frac{\partial f(\boldsymbol{x}, \boldsymbol{\xi}^k)}{\partial \xi_j} = -J_j^k. \end{aligned} \tag{E.9}$$

Substituting Eqns. (E.7) and (E.9) into Eqn. (E.8) gives the result

$$\frac{\partial S^{k+1}}{\partial \xi_j} = -2\left(J_j^k\right)^T \Big(\boldsymbol{r}^k - \sum_{j=1}^{m} J_j^k \Delta\xi_j^k\Big) = 0.$$

Rearranging the terms and cancelling the coefficient 2 yields

$$\left(J_j^k\right)^T \sum_{l=1}^{m} J_l^k \Delta\xi_l^k = \left(J_j^k\right)^T \boldsymbol{r}^k, \tag{E.10}$$

where we used $j, l \in \{1, 2, \ldots, m\}$ for clarity. Then, $\forall j, l$, Eqn. (E.10) is written in complete matrix-vector form for as

$$\left(\boldsymbol{J}^T\boldsymbol{J}\right)\boldsymbol{\Delta\xi} = \boldsymbol{J}^T\boldsymbol{r}, \tag{E.11}$$

which is called the normal equation with

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial f(\boldsymbol{x},\boldsymbol{\xi})}{\partial \xi_1} & \frac{\partial f(\boldsymbol{x},\boldsymbol{\xi})}{\partial \xi_2} & \cdots & \frac{\partial f(\boldsymbol{x},\boldsymbol{\xi})}{\partial \xi_m} \end{bmatrix} \in \mathbb{R}^{n\times m},$$

where $\hat{\boldsymbol{y}} = f(\boldsymbol{x}, \boldsymbol{\xi}) \in \mathbb{R}^n$ and $\boldsymbol{\xi} \in \mathbb{R}^m$.

Now we want to iteratively improve the parameters $\boldsymbol{\xi}^k \to \boldsymbol{\xi}^{k+1}$ to minimize the residuals ($\boldsymbol{r}$). This is done with the normal Eqn. (E.11) by solving for the shift vector $\Delta\boldsymbol{\xi}$. We use the Levenberg-Marquardt algorithm, which is a combination of the Gauss-Newton and gradient descent methods.

E.1 Levenberg-Marquardt

The Levenberg-Marquardt approach extends the normal equation in Eqn. (E.11) with the Marquardt parameter λ as

$$\left(\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{I}_3\right)\boldsymbol{\Delta\xi} = \boldsymbol{J}^T\boldsymbol{r}.$$

The shift vector $\boldsymbol{\Delta\xi}$ is then computed as

$$\boldsymbol{\Delta\xi} = \left(\boldsymbol{J}^T\boldsymbol{J} + \lambda\boldsymbol{I}_3\right)^{-1}\boldsymbol{J}^T\boldsymbol{r}.$$

Because the effect of λ is relative to $\boldsymbol{J}^T\boldsymbol{J}$ the identity matrix $\boldsymbol{I}_3$ was replaced by Fletcher for the diagonal elements of $\boldsymbol{J}^T\boldsymbol{J}$, thereby making the solution scale invariant [33],

$$\boldsymbol{\Delta\xi} = \left(\boldsymbol{J}^T\boldsymbol{J} + \lambda\cdot\mathrm{diag}(\boldsymbol{J}^T\boldsymbol{J})\right)^{-1}\boldsymbol{J}^T\boldsymbol{r}. \tag{E.12}$$

If $\lambda \ll \boldsymbol{J}^T\boldsymbol{J}$ or $\lambda \gg \boldsymbol{J}^T\boldsymbol{J}$, the above equation shifts to the Gauss-Newton or gradient descent method, respectively,

$$\boldsymbol{\Delta\xi} = \begin{cases} \left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\boldsymbol{J}^T\boldsymbol{r}, & \left(\lambda \ll \boldsymbol{J}^T\boldsymbol{J}\right) \\ \frac{1}{\lambda}\boldsymbol{J}^T\boldsymbol{r}, & \left(\lambda \gg \boldsymbol{J}^T\boldsymbol{J}\right) \end{cases}.$$

The Gauss-Newton method (first case) uses the left pseudo-inverse $\left(\boldsymbol{J}^T\boldsymbol{J}\right)^{-1}\boldsymbol{J}^T$ of $\boldsymbol{J}$ to find the shift vector $\boldsymbol{\Delta\xi}$ to solve a linear least squares problem. The gradient descent method (second case) uses a fraction $1/\lambda$ of the steepest descent vector $\boldsymbol{J}^T\boldsymbol{r}$, noting with Eqn. (E.6) that $\boldsymbol{J}^T$ contains the gradients of $\boldsymbol{r}$ in the descending direction, to compute the shift vector $\boldsymbol{\Delta\xi}$ at each iteration. The Marquardt parameter λ is therefore capable of shifting between the Gauss-Newton and gradient descent methods in least squares problems. The gradient descent method is capable of fast convergence, but moves slowly when the local minimum is in a flat area, at which point it would be wise to switch to Gauss-Newton. Therefore the value of λ is initially relatively high, and is lowered progressively when $\boldsymbol{\Delta\xi}$ moves below a certain threshold [34].

Appendix F Representations of Orientation

In this part of the appendix we describe different representations of orientation that were used in this thesis. First we discuss the rotation matrix [30], second the axis-angle formulation [35], and finally orientation quaternions [35].

F.1 Rotation Matrix

An orientation of a body in 3D space is uniquely defined by a set of three orthonormal vectors, denoted here as the triad $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$. These vectors generally represent a global coordinate frame. In multi-body systems, there are often many more reference frames, called local coordinate frames, represented by a similar triad. To obtain a description from a local to a global description, a rotation matrix projects the local coordinate frame triad to the global. This is discussed below.

Let $\{O\}$ be an orthonormal global coordinate frame represented by the triad $\{\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}, \hat{\boldsymbol{z}}\}$. The position of a point $\boldsymbol{p}$ with respect to $\{O\}$ is expressed with components along each axis of $\{O\}$,

$$\boldsymbol{p} = p_x\hat{\boldsymbol{x}} + p_y\hat{\boldsymbol{y}} + p_z\hat{\boldsymbol{z}}.$$

At point $\boldsymbol{p}$ we have another orthonormal local coordinate frame $\{O'\}$ represented by the triad $\{\hat{\boldsymbol{x}}', \hat{\boldsymbol{y}}', \hat{\boldsymbol{z}}'\}$. To express the local triad with respect to the global coordinate frame, each local axis is projected onto the global axes,

$$\begin{aligned}
\hat{\boldsymbol{x}}' &= x'_x\hat{\boldsymbol{x}} + x'_y\hat{\boldsymbol{y}} + x'_z\hat{\boldsymbol{z}},\\
\hat{\boldsymbol{y}}' &= y'_x\hat{\boldsymbol{x}} + y'_y\hat{\boldsymbol{y}} + y'_z\hat{\boldsymbol{z}},\\
\hat{\boldsymbol{z}}' &= z'_x\hat{\boldsymbol{x}} + z'_y\hat{\boldsymbol{y}} + z'_z\hat{\boldsymbol{z}},
\end{aligned}$$

where x'_x, x'_y, x'_z are the lengths of the projected local axis $\hat{\boldsymbol{x}}'$ onto the global triad. The global description of the local triad can be stored into a 3×3 matrix,

$$\begin{bmatrix}\hat{\boldsymbol{x}}' & \hat{\boldsymbol{y}}' & \hat{\boldsymbol{z}}'\end{bmatrix} = \begin{bmatrix} x'_x & y'_x & z'_x \\ x'_y & y'_y & z'_y \\ x'_z & y'_z & z'_z \end{bmatrix}.$$

The lengths of the projected vectors are also termed the direction cosines between $\{O'\}$ and $\{O\}$.

If we know the angle (θ) between two axes, the direction cosines follow from the definition of the dot product between the axes. Consider the vector projection in figure F.1. The component x'_x is given by

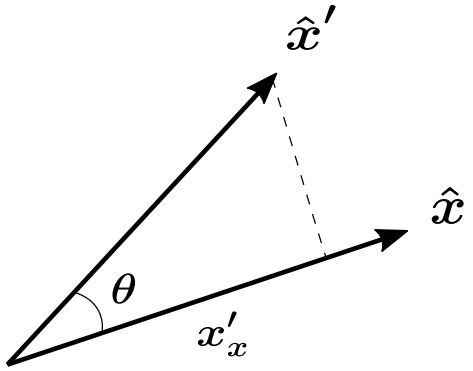

Figure F.1: The projection of a unit vector $\hat{\boldsymbol{x}}'$ onto a vector $\hat{\boldsymbol{x}}$ is obtained from the angle (θ) between the two vectors.

$$x'_x = \|\hat{\boldsymbol{x}}\|\|\hat{\boldsymbol{x}}'\|\cos\theta = \cos\theta,$$

which we can rewrite in terms of the global and local triad as

$$\cos\theta = \frac{\hat{\boldsymbol{x}}' \cdot \hat{\boldsymbol{x}}}{\|\hat{\boldsymbol{x}}'\|\|\hat{\boldsymbol{x}}\|} = \hat{\boldsymbol{x}}' \cdot \hat{\boldsymbol{x}} = \hat{\boldsymbol{x}}'^T \cdot \hat{\boldsymbol{x}}.$$

The 3×3 matrix of direction cosines can then be written as

$$\begin{bmatrix}\hat{\boldsymbol{x}}' & \hat{\boldsymbol{y}}' & \hat{\boldsymbol{z}}'\end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{x}} & \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{x}} & \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{x}} \\ \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{y}} & \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{y}} & \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{y}} \\ \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{z}} & \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{z}} & \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{z}} \end{bmatrix} = \boldsymbol{R},$$

where $\boldsymbol{R} \in SO(3)$ is the symbol for a rotation matrix. If we look at definition of the terms inside the rotation matrix, note that the columns represent the projection of the local triad of $\{O'\}$ onto the global triad of $\{O\}$, and the rows represent the projection of the global onto the local triad (and inverting the rotation is done by simply taking the transpose of $\boldsymbol{R}$). Therefore, in the matrix configuration $\boldsymbol{R}$, any point $\boldsymbol{p}' = p'_x\hat{\boldsymbol{x}}' + p'_y\hat{\boldsymbol{y}}' + p'_z\hat{\boldsymbol{z}}'$ written in terms of the unit vectors of the local frame $\{O'\}$ can be projected onto the unit vectors of the reference frame by performing multiplication with $\boldsymbol{R}$,

$$\boldsymbol{p} = \boldsymbol{R}\boldsymbol{p}'.$$

If we were to invert $\boldsymbol{R}$, with $\boldsymbol{R}^{-1} = \boldsymbol{R}^T$, then we would get

$$\begin{bmatrix}\hat{\boldsymbol{x}} & \hat{\boldsymbol{y}} & \hat{\boldsymbol{z}}\end{bmatrix} = \begin{bmatrix} \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{x}} & \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{y}} & \hat{\boldsymbol{x}}'^T\hat{\boldsymbol{z}} \\ \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{x}} & \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{y}} & \hat{\boldsymbol{y}}'^T\hat{\boldsymbol{z}} \\ \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{x}} & \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{y}} & \hat{\boldsymbol{z}}'^T\hat{\boldsymbol{z}} \end{bmatrix} = \boldsymbol{R}^T.$$

Thus if we have a vector $\boldsymbol{p} = p_x\hat{\boldsymbol{x}} + p_y\hat{\boldsymbol{y}} + p_z\hat{\boldsymbol{z}}$, the projection of this vector on the local reference axes is given by,

$$\boldsymbol{p}' = \boldsymbol{R}^T\boldsymbol{p}.$$

F.2 Axis-Angle of Rotation

The change in orientation of a triad $\{\boldsymbol{x}', \boldsymbol{y}', \boldsymbol{z}'\}$ to a triad $\{\boldsymbol{x}'', \boldsymbol{y}'', \boldsymbol{z}''\}$ can be represented with an angle of rotation (θ) and a unit axis of rotation ($\boldsymbol{k}$). In essence this is just the rotation of three vectors to three other vectors. There we discuss the axis-angle rotation for a single position vector below.

In this section we consider $\boldsymbol{v}$ be a position vector and $\boldsymbol{v}'$ the rotated position vector, expressed relative to an orthonormal global coordinate frame $\{O\}$. How we obtain the description of the rotated position vector from the original vector, an axis of rotation, and an angle of rotation is called the Rodrigues rotation formula,

$$\boldsymbol{v}' = \boldsymbol{v}\cos\theta + \boldsymbol{k}(\boldsymbol{v}\cdot\boldsymbol{k})(1-\cos\theta) + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta. \quad \text{(F.1)}$$

To derive the Rodrigues rotation formula in Eqn. F.1, first consider that the position vector ($\boldsymbol{v}$) can be decomposed into a parallel ($\boldsymbol{v}_\parallel$) and perpendicular ($\boldsymbol{v}_\perp$) component relative to $\boldsymbol{k}$,

$$\boldsymbol{v} = \boldsymbol{v}_\parallel + \boldsymbol{v}_\perp,$$

as is shown in Fig. F.2.

The length of the projected parallel component can be represented with vector projection (Fig. F.1) as

$$\left.\begin{aligned} \boldsymbol{v}_\parallel &= \|\boldsymbol{v}_\parallel\|\boldsymbol{k} \\ \|\boldsymbol{v}_\parallel\| &= \|\boldsymbol{v}\|\cos\theta \\ \cos\theta &= \frac{\boldsymbol{v}\cdot\boldsymbol{k}}{\|\boldsymbol{v}\|\|\boldsymbol{k}\|} \end{aligned}\right\} \quad\Rightarrow\quad \|\boldsymbol{v}_\parallel\| = \frac{\boldsymbol{v}\cdot\boldsymbol{k}}{\|\boldsymbol{k}\|} = \boldsymbol{v}\cdot\boldsymbol{k},$$

such that,

$$\boldsymbol{v}_\parallel = (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k}.$$

The perpendicular component, also called the vector rejection, follows from taking cross-products between the axis of rotation ($\boldsymbol{k}$) and original vector ($\boldsymbol{v}$) as shown in Fig. F.2,

$$\begin{aligned} \boldsymbol{v}_\perp &= \boldsymbol{v} - \boldsymbol{v}_\parallel \\ &= \boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k}. \end{aligned}$$

and can be rewritten, using the vector identity $A\times(B\times C) = B(A\cdot C) - C(A\cdot B)$ by substituting $A = \boldsymbol{k}$, $B = \boldsymbol{k}$, and $C = \boldsymbol{v}$, as

$$\boldsymbol{v}_\perp = -\boldsymbol{k}\times(\boldsymbol{k}\times\boldsymbol{v})$$

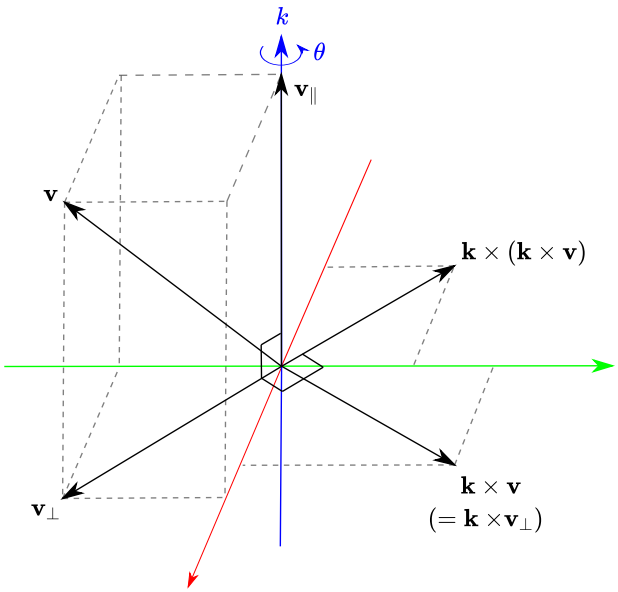

Figure F.2: A vector $\boldsymbol{v}$ that is rotated about an axis of rotation ($\boldsymbol{k}$) by an angle of rotation (θ) can be decomposed into a vector parallel to the axis of rotation ($\boldsymbol{v}_\parallel$) and perpendicular to the axis of rotation ($\boldsymbol{v}_\perp$). The perpendicular vector $\boldsymbol{v}_\perp$ can be expressed with cross-products between the axis ($\boldsymbol{k}$) and original vector ($\boldsymbol{v}$).

Now we have expressed the original vector ($\boldsymbol{v}$) in terms of a parallel and perpendicular component expressed with an axis of rotation ($\boldsymbol{k}$),

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{v}_\parallel + \boldsymbol{v}_\perp \\ &= (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k} - \boldsymbol{k}\times(\boldsymbol{k}\times\boldsymbol{v}). \end{aligned} \quad \text{(F.2)}$$

Now we want to rotate $\boldsymbol{v}$ by an angle θ about the rotation axis $\boldsymbol{k}$. The parallel component is unaffected by a rotation, i.e.

$$\boldsymbol{v}'_\parallel = \boldsymbol{v}_\parallel,$$

and the perpendicular component changes direction, but not magnitude, i.e.

$$\boldsymbol{v}'_\perp = \boldsymbol{v}_\perp\cos\theta + (\boldsymbol{k}\times\boldsymbol{v}_\perp)\sin\theta,$$

where $(\boldsymbol{k}\times\boldsymbol{v}_\perp)$ is at right angles with $\boldsymbol{k}$ and $\boldsymbol{v}_\perp$ according to the right hand rule. From Eqn. (F.2) we can see that $(\boldsymbol{k}\times\boldsymbol{v}_\perp) = (\boldsymbol{k}\times\boldsymbol{v})$, such that the above equation becomes

$$\boldsymbol{v}'_\perp = \boldsymbol{v}_\perp\cos\theta + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta,$$

which represents a simple 2D rotation. Now we obtain the Rodrigues rotation formula originally given in Eqn. (F.1) by summing the rotated parallel and

perpendicular components,

$$\begin{aligned}\boldsymbol{v}' &= \boldsymbol{v}'_{\parallel} + \boldsymbol{v}'_{\perp},\\ &= \boldsymbol{v}_{\parallel} + \boldsymbol{v}'_{\perp},\\ &= (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k} + \boldsymbol{v}_{\perp}\cos\theta + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta,\\ &= (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k} + \boldsymbol{v}\cos\theta - (\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k}\cos\theta + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta,\\ &= \boldsymbol{v}\cos\theta + \boldsymbol{k}(\boldsymbol{v}\cdot\boldsymbol{k})(1-\cos\theta) + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta.\end{aligned} \tag{F.3}$$

The axis-angle rotation and rotation by a rotation matrix are both able to rotate $\boldsymbol{v}$ to $\boldsymbol{v}'$ and therefore it is possible to obtain a rotation matrix from an axis and angle of rotation. We can linearise Eqn. (F.3) with respect to $\boldsymbol{v}$ to obtain a matrix-vector multiplication that is of the form $\boldsymbol{v}' = \boldsymbol{R}\boldsymbol{v}$.

F.2.1 Axis-Angle to Rotation Matrix

To get Eqn. (F.3) to a matrix-vector multiplication, consider that

$$\boldsymbol{k}\times\boldsymbol{v} = [\boldsymbol{k}]_{\times}\boldsymbol{v},$$

where $[\boldsymbol{k}]_{\times} : \mathbb{R}^3 \to \mathbb{R}^{3\times 3}$ represents a skew-symmetric matrix. Furthermore, substituting $A = \boldsymbol{k}$, $B = \boldsymbol{k}$, and $C = \boldsymbol{v}$ into the vector identity $A \times (B \times C) = B(A\cdot C) - C(A\cdot B)$ gives

$$\begin{aligned}\boldsymbol{k}(\boldsymbol{v}\cdot\boldsymbol{k}) &= \boldsymbol{k}\times(\boldsymbol{k}\times\boldsymbol{v}) + \boldsymbol{v},\\ &= [\boldsymbol{k}]^2_{\times}\boldsymbol{v} + \boldsymbol{v}.\end{aligned}$$

The rotation matrix $\boldsymbol{R}(\boldsymbol{k},\theta)$ associated with the axis $(\boldsymbol{k})$ and angle (θ) of rotation then follows as

$$\boldsymbol{R}(\boldsymbol{k},\theta) = \boldsymbol{I}_3 + \sin\theta[\boldsymbol{k}]_{\times} + (1-\cos\theta)[\boldsymbol{k}]^2_{\times}. \tag{F.4}$$

If we substitute $\boldsymbol{k} = [k_1, k_2, k_3]^T$ into Eqn. F.4 we can also obtain the axis-angle representation from a rotation matrix.

F.2.2 Rotation Matrix to Axis-Angle

Consider a rotation matrix $\boldsymbol{R}(\boldsymbol{k},\theta)$ for which the axis of rotation $\boldsymbol{k}$ and angle θ are unknown, and we want to know these values. First expand Eqn. (F.4) to Eqn. (F.5). The trace of $\boldsymbol{R}(\boldsymbol{k},\theta)$ is given by

$$\begin{aligned}\text{trace}(\boldsymbol{R}) &= 3 + 2(c_\theta - 1)(k_3^2 + k_2^2 + k_1^2),\\ &= 3 + 2(c_\theta - 1)(\boldsymbol{k}\cdot\boldsymbol{k})\\ &= 3 + 2(c_\theta - 1),\\ &= 1 + 2\cos\theta,\end{aligned}$$

such that the angle of rotation is given by

$$\theta = \cos^{-1}\left(\frac{\text{trace}(\boldsymbol{R}) - 1}{2}\right). \tag{F.6}$$

The axis of rotation $(\boldsymbol{k})$ can be found from the off-diagonal terms. It can quite easily be seen that $\boldsymbol{k}$ is given by

$$\boldsymbol{k} = \frac{1}{2\sin\theta}\begin{bmatrix} R_{32} - R_{23}\\ R_{13} - R_{31}\\ R_{21} - R_{12}\end{bmatrix}, \tag{F.7}$$

where R_{ij} represents the element (i,j) of $\boldsymbol{R}$. We observe that Eqn. (F.7) has numerical problems for small rotations and when $\theta = \pi$. Below we discuss how then to compute the axis of rotation.

When $\theta = \pi$, Eqn. (F.4) reduces to

$$\boldsymbol{R}(\boldsymbol{k},\theta) = \boldsymbol{I}_3 + 2[\boldsymbol{k}]^2_{\times}, \qquad \theta = \pi$$

Furthermore, when $\|\boldsymbol{k}\| = 1$, we have the relationship $[\boldsymbol{k}]^2_{\times} = \boldsymbol{k}\boldsymbol{k}^T - \boldsymbol{I}_3$. First consider $\boldsymbol{k}\boldsymbol{k}^T$,

$$\boldsymbol{k}\boldsymbol{k}^T = \begin{bmatrix} k_1^2 & k_1k_2 & k_1k_3\\ k_2k_1 & k_2^2 & k_2k_3\\ k_3k_1 & k_2k_3 & k_3^2\end{bmatrix}.$$

The off-diagonal terms are the same as for $[\boldsymbol{k}]^2_{\times}$. The diagonal terms are different, but we have the relationship for unit-vectors that $k_1^2 + k_2^2 + k_3^2 = 1$. Therefore, the diagonal elements have the equality $k_1^2 - 1 = -(k_2^2 + k_3^2)$. Therefore, for $\theta = \pi$, we can rewrite

$$\boldsymbol{R} = \boldsymbol{I}_3 + 2[\boldsymbol{k}]^2_{\times} = 2\boldsymbol{k}\boldsymbol{k}^T - \boldsymbol{I}_3,$$

and further rewrite

$$\boldsymbol{k}\boldsymbol{k}^T = \frac{\boldsymbol{R} + \boldsymbol{I}_3}{2}, \qquad \theta = \pi.$$

Then the components of the axis of rotation $(\boldsymbol{k})$ are found by taking the square root of the individual diagonal terms. Their signs can be deduced from the values of the off-diagonal terms. Always two solutions exist at $\theta = \pi$, because $\boldsymbol{R}(\pi) = \boldsymbol{R}(-\pi)$, and so the solutions are $\boldsymbol{k}$ and -$\boldsymbol{k}$.

$$\boldsymbol{R}(\boldsymbol{k},\theta) = \begin{bmatrix} 1 + (c_\theta - 1)(k_3^2 + k_2^2) & k_2k_1(1-c_\theta) - k_3s_\theta & k_3k_1(1-c_\theta) + k_2s_\theta\\ k_2k_1(1-c_\theta) + k_3s_\theta & 1 + (c_\theta - 1)(k_3^2 + k_1^2) & k_3k_2(1-c_\theta) - k_1s_\theta\\ k_1k_3(1-c_\theta) - k_2s_\theta & k_3k_2(1-c_\theta) + k_1s_\theta & 1 + (c_\theta - 1)(k_2^2 + k_1^2)\end{bmatrix} \tag{F.5}$$

F.3 Quaternions

Quaternions consist of one real and three complex numbers. If the magnitude of the quaternion equals one, it is called an orientation quaternion. Orientation quaternions are closely related to the axis-angle representation, and give more numerical stability compared to rotation matrices. Below an introduction to quaternions is given, and how orientation quaternions can be used to get the Rodrigues rotation formula derive in Eqn. (F.3).

A quaternion $\boldsymbol{q} \in \mathbb{H}$ is expressed as

$$\boldsymbol{q} = q_r + q_i\boldsymbol{i} + q_j\boldsymbol{j} + q_k\boldsymbol{k},$$

where $\boldsymbol{i}$, $\boldsymbol{j}$, $\boldsymbol{k}$ are complex orthonormal complex vectors satisfying the following identities

$$\boldsymbol{i}^2 = \boldsymbol{j}^2 = \boldsymbol{k}^2 = \boldsymbol{ijk} = -1,$$

$$\begin{aligned}
&&\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{jk} = -\boldsymbol{i} &\Rightarrow \boldsymbol{ij} = \boldsymbol{k},\\
\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{jk} = -\boldsymbol{i} &\Rightarrow \boldsymbol{j} = -\boldsymbol{ik} &\Rightarrow \boldsymbol{ik} = -\boldsymbol{j},\\
\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{jk} = -\boldsymbol{i} &\Rightarrow \boldsymbol{k} = -\boldsymbol{ji} &\Rightarrow \boldsymbol{ji} = -\boldsymbol{k},\\
&&\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{jk} = -\boldsymbol{i} &\Rightarrow \boldsymbol{jk} = \boldsymbol{i},\\
\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{ij} = -\boldsymbol{k} &\Rightarrow \boldsymbol{i} = -\boldsymbol{kj} &\Rightarrow \boldsymbol{ki} = \boldsymbol{j},\\
\boldsymbol{ijk} = -1 &\Rightarrow -\boldsymbol{ij} = -\boldsymbol{k} &\Rightarrow \boldsymbol{i} = -\boldsymbol{kj} &\Rightarrow \boldsymbol{kj} = -\boldsymbol{i}.
\end{aligned}$$

These identities are used during quaternion multiplication, which is not commutative.

F.3.1 Quaternion Multiplication

Consider two quaternions $\boldsymbol{q}_1 = a + b\boldsymbol{i} + c\boldsymbol{j} + d\boldsymbol{k}$ and $\boldsymbol{q}_2 = e + f\boldsymbol{i} + g\boldsymbol{j} + h\boldsymbol{k}$. The quaternion product is found through componentwise scalar multiplication,

$$\begin{aligned}
\boldsymbol{q}_1\boldsymbol{q}_2 =&(ae - bf - cg - dh) + (af + be + ch - dg)\boldsymbol{i} +\\
&(ag - bh + ce + df)\boldsymbol{j} + (ah + bg - cf + de)\boldsymbol{k}.
\end{aligned}$$

The quaternion product is itself a quaternion, again consisting of a scalar plus vector part. Quaternions are therefore often written in the following form, divided into a scalar and vector part,

$$\boldsymbol{q} = (q_r, \boldsymbol{q}_i),$$

where $q_r \in \mathbb{R}$ and $\boldsymbol{q}_i \in \mathbb{C}^3$. A quaternion product is similarly expressed as

$$(a, \boldsymbol{q}_{i1})(e, \boldsymbol{q}_{i2}) = ae + a\boldsymbol{q}_{i2} + e\boldsymbol{q}_{i1} + \boldsymbol{q}_{i1}\boldsymbol{q}_{i2},$$

where the imaginary product follows from

$$\boldsymbol{q}_{i1}\boldsymbol{q}_{i2} = \boldsymbol{q}_{i1} \times \boldsymbol{q}_{i2} - \boldsymbol{q}_{i1} \cdot \boldsymbol{q}_{i2}. \tag{F.8}$$

The imaginary vector dot product $(\boldsymbol{q}_{i1} \cdot \boldsymbol{q}_{i2})$ returns a real scalar, and thus a quaternion product gives

$$(a, \boldsymbol{q}_{i1})(e, \boldsymbol{q}_{i2}) = (ae - \boldsymbol{q}_{i1} \cdot \boldsymbol{q}_{i2},\ \ a\boldsymbol{q}_{i2} + e\boldsymbol{q}_{i1} + \boldsymbol{q}_{i1} \times \boldsymbol{q}_{i2}). \tag{F.9}$$

If a quaternion is of unit length, $\|\boldsymbol{q}\| = 1$, it is also called an orientation quaternion. In that case a quaternion multiplication of orientation quaternions represents consecutive rotations.

F.3.2 Orientation Quaternion

An orientation quaternion is a quaternion of unit length. It is possible to define an orientation quaternion from an axis-angle representation that, when applied to rotate a vector, results in the Rodrigues orientation formula in Eqn. (F.3).

Consider an angle of rotation (θ) and a unit axis of rotation $(\boldsymbol{k})$. Then an orientation quaternion is defined as

$$\boldsymbol{q} = \left(\cos\left(\frac{\theta}{2}\right),\ \ \boldsymbol{k}\sin\left(\frac{\theta}{2}\right)\right), \tag{F.10}$$

which is of unit length,

$$\begin{aligned}
\|\boldsymbol{q}\| =&\sqrt{\cos^2\left(\frac{\theta}{2}\right) + (k_1^2 + k_2^2 + k_3^2)\sin^2\left(\frac{\theta}{2}\right)}\\
=&\sqrt{\cos^2\left(\frac{\theta}{2}\right) + \sin^2\left(\frac{\theta}{2}\right)} = 1
\end{aligned}$$

Similar to the inverse of a rotation matrix, there also exists an orientation quaternion inverse. An orientation quaternion multiplied by its inverse gives a unit-quaternion with zero vector part, i.e.

$$\boldsymbol{q}\boldsymbol{q}^{-1} = (1, \boldsymbol{0})$$

The quaternion inverse $(\boldsymbol{q}^{-1})$ is equal to the complex conjugate of the quaternion,

$$\boldsymbol{q}\boldsymbol{q}^* = \|\boldsymbol{q}\|^2 \rightarrow \frac{\boldsymbol{q}\boldsymbol{q}^*}{\|\boldsymbol{q}\|^2} = (1, \boldsymbol{0}) \Rightarrow \boldsymbol{q}^{-1} = \frac{\boldsymbol{q}^*}{\|\boldsymbol{q}\|^2} = \boldsymbol{q}^*.$$

Thus, the quaternion inverse is found by simply negating the imaginary vector component, such that with an orientation quaternion defined in Eqn. (F.10) the inverse is given by

$$\boldsymbol{q}^{-1} = \left(\cos\left(\frac{\theta}{2}\right),\ \ -\boldsymbol{k}\sin\left(\frac{\theta}{2}\right)\right). \tag{F.11}$$

The orientation quaternion and orientation quaternion inverse are used in quaternion rotation.

F.3.3 Orientation Quaternion Rotation

In the sections on rotation matrices and axis-angle representations we considered 3D Cartesian position vectors. If we consider a vector $(\boldsymbol{v})$, expressed in

Cartesian coordinates, that we want to rotate to $\boldsymbol{v}'$, we must define $\boldsymbol{v}$ as the imaginary vector part of a quaternion,

$$\boldsymbol{v} \rightarrow (0, \boldsymbol{v}_i).$$

The quaternion rotation with orientation quaternion $\boldsymbol{q}$ is then given by

$$\boldsymbol{v}' \leftarrow \boldsymbol{q}\boldsymbol{v}\boldsymbol{q}^{-1}. \tag{F.12}$$

It turns out that expanding Eqn. (F.12) results in the Rodrigues rotation formula, where use the following notation $\boldsymbol{v} \equiv \boldsymbol{v}_i$ and $\boldsymbol{k} \equiv \boldsymbol{k}_i$ (axis of rotation) for easy of notation,

$$\begin{aligned}
\boldsymbol{v}' =& \boldsymbol{q}(0, \boldsymbol{v})\boldsymbol{q}^{-1} \\
\boldsymbol{v}' =& \left(\cos\frac{\theta}{2} + \boldsymbol{k}\sin\frac{\theta}{2}\right)\boldsymbol{v}\left(\cos\frac{\theta}{2} - \boldsymbol{k}\sin\frac{\theta}{2}\right), \\
=& \boldsymbol{v}\cos^2\frac{\theta}{2} + (\boldsymbol{k}\boldsymbol{v} - \boldsymbol{v}\boldsymbol{k})\sin\frac{\theta}{2}\cos\frac{\theta}{2} - \boldsymbol{k}\boldsymbol{v}\boldsymbol{k}\sin^2\frac{\theta}{2}, \\
=& \boldsymbol{v}\cos^2\frac{\theta}{2} + 2(\boldsymbol{k}\times\boldsymbol{v})\sin\frac{\theta}{2}\cos\frac{\theta}{2} \\
& - (\boldsymbol{v} - 2\boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v}))\sin^2\frac{\theta}{2}, \\
=& \boldsymbol{v}\left(\cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2}\right) + 2(\boldsymbol{k}\times\boldsymbol{v})\sin\frac{\theta}{2}\cos\frac{\theta}{2} \\
& + 2\boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v}))\sin^2\frac{\theta}{2}, \\
=& \boldsymbol{v}\cos\theta + (\boldsymbol{k}\times\boldsymbol{v})\sin\theta + \boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v})(1-\cos\theta),
\end{aligned} \tag{F.13}$$

which is equal to the Rodrigues rotation formula in Eqn. (F.1). We have used the following quaternion and trigonometric identities,

- Imaginary vector multiplication and subtraction

$$\begin{aligned}
\boldsymbol{k}\boldsymbol{v} - \boldsymbol{v}\boldsymbol{k} &= (-\boldsymbol{k}\cdot\boldsymbol{v}, \boldsymbol{k}\times\boldsymbol{v}) - (-\boldsymbol{v}\cdot\boldsymbol{k}, \boldsymbol{v}\times\boldsymbol{k}) \\
&= (0, 2\boldsymbol{k}\times\boldsymbol{v})
\end{aligned}$$

- Imaginary vector quaternion multiplications

$$\begin{aligned}
\boldsymbol{k}\boldsymbol{v}\boldsymbol{k} &= \boldsymbol{k}(-\boldsymbol{v}\cdot\boldsymbol{k}, \boldsymbol{v}\times\boldsymbol{k}) \\
&= (-\boldsymbol{k}\cdot(\boldsymbol{v}\times\boldsymbol{k}), -(\boldsymbol{v}\cdot\boldsymbol{k})\boldsymbol{k} + \boldsymbol{k}\times(\boldsymbol{v}\times\boldsymbol{k})) \\
&= (0, -\boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v}) + \boldsymbol{v} - \boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v})) \\
&= (0, \boldsymbol{v} - 2\boldsymbol{k}(\boldsymbol{k}\cdot\boldsymbol{v}))
\end{aligned}$$

- Trigonometric Identities

$$\begin{aligned}
2\sin^2\frac{\theta}{2} &= 1 - \cos\theta \\
2\sin\frac{\theta}{2}\cos\frac{\theta}{2} &= \sin\theta \\
\cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2} &= \cos\theta
\end{aligned}$$

Similar to how we can rewrite a rotation matrix to an angle-axis rotation and vice versa, we can do the same with orientation quaternions.

F.3.4 Orientation Quaternion to Rotation Matrix

To obtain the rotation matrix associated with an orientation quaternion ($\boldsymbol{R}(\boldsymbol{q})$) we should expand the quaternion rotation in Eqn. (F.12) such that we get an expression linear in the original vector ($\boldsymbol{v}$) and in terms of the components of the orientation quaternion ($\boldsymbol{q}$). First, rewrite the quaternion rotation in Eqn. (F.12) to explicit real and imaginary components,

$$(0, \boldsymbol{v}') = (q_r, \boldsymbol{q}_i)(0, \boldsymbol{v})(q_r, -\boldsymbol{q}_i),$$

which is expanded below $(q_r, \boldsymbol{q}_i)(0, \boldsymbol{v})$ yields

$$\begin{aligned}
(q_r, \boldsymbol{q}_i)(0, \boldsymbol{v}) &= (-\boldsymbol{q}_i\cdot\boldsymbol{v},\ \ q_r\boldsymbol{v} + \boldsymbol{q}_i\times\boldsymbol{v}), \\
(q_r, \boldsymbol{q}_i)(0, \boldsymbol{v})(q_r, -\boldsymbol{q}_i) &= \Big(q_r(-\boldsymbol{q}_i\cdot\boldsymbol{v}) - \\
& \quad (q_r\boldsymbol{v} + \boldsymbol{q}_i\times\boldsymbol{v}))\cdot(-\boldsymbol{q}_i), \\
& \quad -\boldsymbol{q}_i(-\boldsymbol{q}_i\cdot\boldsymbol{v}) + \\
& \quad q_r^2\boldsymbol{v} + q_r(\boldsymbol{q}_i\times\boldsymbol{v}) + \\
& \quad (q_r\boldsymbol{v} + \boldsymbol{q}_i\times\boldsymbol{v})\times -\boldsymbol{q}_i\Big).
\end{aligned}$$

We can simplify the expanded quaternion rotation by considering that the dot product between orthogonal vectors is zero,

$$(\boldsymbol{q}_i\times\boldsymbol{v})\cdot(-\boldsymbol{q}_i) = 0,$$

and

$$\boldsymbol{q}_i(\boldsymbol{q}_i\cdot\boldsymbol{v}) = \boldsymbol{q}_i\boldsymbol{q}_i^T\boldsymbol{v},$$

such that the quaternion rotation is given by

$$\begin{aligned}
&(q_r, \boldsymbol{q}_i)(0, \boldsymbol{v})(q_r, -\boldsymbol{q}_i) = \\
&\Big(0,\ \ q_r^2\boldsymbol{v} + \boldsymbol{q}_i\boldsymbol{q}_i^T\boldsymbol{v} + 2q_r[\boldsymbol{q}_i]_\times\boldsymbol{v} + [\boldsymbol{q}_i]_\times^2\boldsymbol{v}\Big),
\end{aligned} \tag{F.14}$$

and we can see that the rotation matrix $\boldsymbol{R}(\boldsymbol{q})$ is given by

$$\boldsymbol{R}(q) = q_r^2\boldsymbol{I}_3 + \boldsymbol{q}_i\boldsymbol{q}_i^T + 2q_r[\boldsymbol{q}_i]_\times + [\boldsymbol{q}_i]_\times^2, \tag{F.15}$$

which expands to Eqn. (F.16).

F.3.5 Orientation Quaternion to Axis-Angle

We have seen that an orientation quaternion can be constructed from a unit axis of rotation ($\boldsymbol{k}$) and angle of rotation (θ) in Eqn. (F.10). Thus we can quite easily obtain an axis and angle of rotation from an orientation quaternion as well.

$$\boldsymbol{R}(\boldsymbol{q}) = \begin{bmatrix} q_r^2 + q_i^2 - q_j^2 - q_k^2 & 2(q_i q_j - q_r q_k) & 2(q_i q_k + q_r q_j) \\ 2(q_j q_i + q_r q_k) & q_r^2 - q_i^2 + q_j^2 - q_k^2 & 2(q_j q_k - q_r q_i) \\ 2(q_k q_i - q_r q_j) & 2(q_k q_j + q_r q_i) & q_r^2 - q_i^2 - q_2^2 + q_k^2 \end{bmatrix} \quad \text{(F.16)}$$

Given an orientation quaternion $\boldsymbol{q} = q_r + q_i\boldsymbol{i} + q_j\boldsymbol{j} + q_k\boldsymbol{k}$, the angle and axis of rotation is found by

$$q = \begin{bmatrix} q_r \\ q_i \\ q_j \\ q_k \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ k_1 \sin(\theta/2) \\ k_2 \sin(\theta/2) \\ k_3 \sin(\theta/2) \end{bmatrix} \Rightarrow \begin{cases} \theta = 2\cos^{-1}(q_r) \\ k_1 = q_i / \sin(\theta/2) \\ k_2 = q_j / \sin(\theta/2) \\ k_3 = q_k / \sin(\theta/2) \end{cases} \quad \text{(F.17)}$$

which only gives a problem when $\theta = 0$. Therefore one should always do a zero check on θ before computing $\boldsymbol{k}$.

F.3.6 Orientation Quaternion Error

When using orientation quaternions (or any other representation) to represent and, most importantly, steer, the orientation of an orthonormal coordinate frame to some desired orientation, there will exist a discrepancy between the real and desired orientation. This is the orientation error which we want to quantify. We first start with a short description of the orientation error in terms of rotation matrices because it is easier to visualize. Then we follow up with the orientation quaternion error.

First, consider two local orthonormal coordinate frames $\{A\}$ and $\{B\}$, defined relative to a global coordinate frame $\{O\}$. The orientations of the local frames with respect to the global frame, using rotation matrices, are denoted as ${}^O_A\boldsymbol{R}$ and ${}^0_B\boldsymbol{R}$. Now consider ${}^O_B\boldsymbol{R}$ to be a desired orientation and ${}^O_A\boldsymbol{R}$ the current orientation. The difference in orientation is then denoted as by $\tilde{\boldsymbol{R}}$, and given by

$$\tilde{\boldsymbol{R}}\,{}^O_A\boldsymbol{R} = {}^O_B\boldsymbol{R} \Rightarrow \tilde{\boldsymbol{R}} = {}^O_B\boldsymbol{R}\,{}^O_A\boldsymbol{R}^T.$$

When controlling the orientation of frame $\{A\}$ to a desired frame $\{B\}$, the final orientation error should ideally be zero, that is $\tilde{\boldsymbol{R}} \approx \boldsymbol{I}_3$.

Now consider the orientation quaternions

$$\begin{aligned} {}^O_A\boldsymbol{q} &= ({}^O_A\eta, {}^O_A\boldsymbol{q}_i) \\ {}^O_B\boldsymbol{q} &= ({}^O_B\eta, {}^O_B\boldsymbol{q}_i), \end{aligned}$$

which differ by a relative orientation quaternion $\tilde{\boldsymbol{q}} = (\delta\eta, \delta\boldsymbol{q}_i)$,

$$\tilde{\boldsymbol{q}}\,{}^O_A\boldsymbol{q} = {}^O_B\boldsymbol{q} \Rightarrow \tilde{\boldsymbol{q}} = {}^O_B\boldsymbol{q}\,{}^O_A\boldsymbol{q}^{-1},$$

where the orientation quaternion inverse is its complex conjugate (Eqn. (F.11)). Then expanding the orientation quaternion multiplication on the right hand side gives

$$\begin{aligned} \tilde{\boldsymbol{q}} &= ({}^O_B\eta, {}^O_B\boldsymbol{q}_i)({}^O_A\eta, -{}^O_A\boldsymbol{q}_i) \\ &= ({}^O_A\eta\,{}^O_B\eta - {}^O_B\boldsymbol{q}_i \cdot -{}^O_A\boldsymbol{q}_i,\ {}^O_A\eta\,{}^O_B\boldsymbol{q}_i - {}^O_B\eta\,{}^O_A\boldsymbol{q}_i + {}^O_B\boldsymbol{q}_i \times -{}^O_A\boldsymbol{q}_i) \\ &= ({}^O_A\eta\,{}^O_B\eta + {}^O_B\boldsymbol{q}_i \cdot {}^O_A\boldsymbol{q}_i,\ {}^O_A\eta\,{}^O_B\boldsymbol{q}_i - {}^O_B\eta\,{}^O_A\boldsymbol{q}_i + {}^O_A\boldsymbol{q}_i \times {}^O_B\boldsymbol{q}_i) \end{aligned}$$

The real scalar and imaginary vector parts of the orientation quaternion difference, taking ${}^O_A\boldsymbol{q}$ and ${}^O_B\boldsymbol{q}$ as the current and desired orientation quaternions, respectively, are thus given by

$$\begin{aligned} \delta\eta &= {}^O_A\eta\,{}^O_B\eta + {}^O_A\boldsymbol{q}_i{}^T\,{}^O_B\boldsymbol{q}_i, \\ \delta\boldsymbol{q} &= {}^O_A\eta\,{}^O_B\boldsymbol{q}_i - {}^O_B\eta\,{}^O_A\boldsymbol{q}_i + {}^O_A\boldsymbol{q}_i \times {}^O_B\boldsymbol{q}_i. \end{aligned} \quad \text{(F.18)}$$

When ${}^O_A\boldsymbol{q} = {}^O_B\boldsymbol{q}$ then $\delta\eta = 1$ (simply the dot-product with itself) and $\delta\boldsymbol{q}_i = 0$, equivalent to $\boldsymbol{R}(\tilde{\boldsymbol{q}}) = \boldsymbol{I}_3$ in Eqn. (F.16).

F.3.7 Orientation Quaternion Derivative

The quaternion derivative is used to obtain the angular velocity of a coordinate frame (or vector) about an axis of rotation.

Consider the orientation quaternion $\boldsymbol{q} = (q_r, \boldsymbol{q}_i)$ and quaternion position vector $\boldsymbol{v} \equiv (0, \boldsymbol{v})$. For the derivation of the orientation quaternion derivative we must consider the following quaternion multiplications,

$$\begin{aligned} \boldsymbol{q}\boldsymbol{v} &= (q_r v_r - \boldsymbol{q}_i \cdot \boldsymbol{v}, q_r\boldsymbol{v} + v_r\boldsymbol{q}_i + \boldsymbol{q}_i \times \boldsymbol{v}), \\ \boldsymbol{v}\boldsymbol{q} &= (q_r v_r - \boldsymbol{v} \cdot \boldsymbol{q}_i, q_r\boldsymbol{v} + v_r\boldsymbol{q}_i + \boldsymbol{v} \times \boldsymbol{q}_i). \end{aligned} \quad \text{(F.19)}$$

The product of two quaternions $\boldsymbol{q}, \boldsymbol{v} \in \mathbb{H}$ gives a new quaternion. Equation (F.19) can be rewritten as a matrix-vector product, noting that the dot product between the quaternion's imaginary vector parts are commutative,

$$\begin{aligned} \boldsymbol{q}\boldsymbol{v} &= \underbrace{\begin{bmatrix} q_r & -\boldsymbol{q}_i^T \\ \boldsymbol{q}_i & q_r\boldsymbol{I}_3 + [\boldsymbol{q}_i]_\times \end{bmatrix}}_{Q} \begin{bmatrix} 0 \\ \boldsymbol{v} \end{bmatrix} \\ \boldsymbol{v}\boldsymbol{q} &= \underbrace{\begin{bmatrix} q_r & -\boldsymbol{q}_i{}^T \\ \boldsymbol{q}_i & q_r\boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix}}_{\bar{Q}} \begin{bmatrix} 0 \\ \boldsymbol{v} \end{bmatrix} \end{aligned} \quad \text{(F.20)}$$

The rotation of the position quaternion ($\boldsymbol{v}$) to $\boldsymbol{v}'$ with a rotation quaternion $\boldsymbol{q}$ was defined as

$$\boldsymbol{v}' = \boldsymbol{q}\boldsymbol{v}\boldsymbol{q}^{-1} \quad \Rightarrow \quad \boldsymbol{v} = \boldsymbol{q}^{-1}\boldsymbol{v}'\boldsymbol{q}$$

and differentiating $\boldsymbol{v}'$ with respect to time, using the product rule, gives

$$\dot{\boldsymbol{v}}' = \dot{\boldsymbol{q}}\boldsymbol{v}\boldsymbol{q}^{-1} + \boldsymbol{q}\boldsymbol{v}\dot{\boldsymbol{q}}^{-1}. \tag{F.21}$$

Now we substitute the expression for $\boldsymbol{v}$ to express $\dot{\boldsymbol{v}}'$ in terms of the rotated position vector ($\boldsymbol{v}'$) itself,

$$\begin{aligned} \dot{\boldsymbol{v}}' &= \dot{\boldsymbol{q}}\boldsymbol{q}^{-1}\boldsymbol{v}'\boldsymbol{q}\boldsymbol{q}^{-1} + \boldsymbol{q}\boldsymbol{q}^{-1}\boldsymbol{v}'\boldsymbol{q}\dot{\boldsymbol{q}}^{-1}, \\ &= \dot{\boldsymbol{q}}\boldsymbol{q}^{-1}\boldsymbol{v}' + \boldsymbol{v}'\boldsymbol{q}\dot{\boldsymbol{q}}^{-1}. \end{aligned} \tag{F.22}$$

Let's evaluate the expressions $\dot{\boldsymbol{q}}\boldsymbol{q}^{-1}$ and $\boldsymbol{q}\dot{\boldsymbol{q}}^{-1}$. We know that $\boldsymbol{q}\boldsymbol{q}^{-1} = (1, \boldsymbol{0})$, thus

$$\frac{d}{dt}\left(\boldsymbol{q}\boldsymbol{q}^{-1}\right) = \frac{d}{dt}(1, \boldsymbol{0}),$$

such that, with the product rule

$$\dot{\boldsymbol{q}}\boldsymbol{q}^{-1} + \boldsymbol{q}\dot{\boldsymbol{q}}^{-1} = (0, \boldsymbol{0}) \tag{F.23}$$

If we were to write out both expressions, we would obtain:

$$\begin{aligned} \dot{\boldsymbol{q}}\boldsymbol{q}^{-1} &= (\dot{q}_r q_r + \dot{\boldsymbol{q}}_i \cdot \boldsymbol{q}_i, \;\; -\dot{q}_r \boldsymbol{q}_i + q_r \dot{\boldsymbol{q}}_i - \dot{\boldsymbol{q}}_i \times \boldsymbol{q}_i) \\ \boldsymbol{q}\dot{\boldsymbol{q}}^{-1} &= (q_r \dot{q}_r + \boldsymbol{q}_i \cdot \dot{\boldsymbol{q}}_i, \;\; +\dot{q}_r \boldsymbol{q}_i - q_r \dot{\boldsymbol{q}}_i + \dot{\boldsymbol{q}}_i \times \boldsymbol{q}_i) \end{aligned}$$

Notice that the sum of the real scalar parts $Re(\dot{\boldsymbol{q}}\boldsymbol{q}^{-1}) + Re(\boldsymbol{q}\dot{\boldsymbol{q}}^{-1})$ does not automatically cancel to zero, thus both scalar parts must be zero for Eqn. (F.23) to always be true! In addition, the vector parts are equal and opposite, and thus do cancel to zero! Therefore both expressions can be written as

$$\begin{aligned} \dot{\boldsymbol{q}}\boldsymbol{q}^{-1} &= (0, \boldsymbol{w}), \\ \boldsymbol{q}\dot{\boldsymbol{q}}^{-1} &= (0, -\boldsymbol{w}). \end{aligned}$$

Now return to the expression of $\dot{\boldsymbol{v}}'$ in Eqn. (F.22), with $\boldsymbol{v}' = (0, \boldsymbol{v}')$, and substitute the above expressions to obtain

$$\begin{aligned} \dot{\boldsymbol{v}}' &= (0, \boldsymbol{w})(0, \boldsymbol{v}_i') + (0, \boldsymbol{v}_i')(0, -\boldsymbol{w}), \\ &= (0, 2\boldsymbol{w} \times \boldsymbol{v}_i'). \end{aligned} \tag{F.24}$$

Now $2\boldsymbol{w}$ represents an axis of rotation with magnitude equal to the angular velocity ($\boldsymbol{\omega}$). Therefore $\boldsymbol{\omega} = 2\boldsymbol{w}$, and making the angular velocity quaternion $\boldsymbol{\omega} \equiv (0, 2\boldsymbol{w})$ we can write

$$\boldsymbol{\omega} = 2\dot{\boldsymbol{q}}\boldsymbol{q}^{-1} \tag{F.25}$$

such that dividing by 2 and post-multiplying with $\boldsymbol{q}$ gives the quaternion derivative in terms of the angular velocity and the orientation quaternion itself,

$$\dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{\omega}\boldsymbol{q}, \tag{F.26}$$

which again can be written as a matrix-vector multiplication,

$$\begin{bmatrix} \dot{q}_0 \\ \dot{\boldsymbol{q}}_i \end{bmatrix} = \frac{1}{2}\begin{bmatrix} q_r & -\boldsymbol{q}_i{}^T \\ \boldsymbol{q}_i & q_r \boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix}\begin{bmatrix} 0 \\ \boldsymbol{\omega} \end{bmatrix}. \tag{F.27}$$

Or simplified,

$$\dot{\boldsymbol{q}} = \frac{1}{2}\begin{bmatrix} -\boldsymbol{q}_i{}^T \\ q_r \boldsymbol{I}_3 - [\boldsymbol{q}_i]_\times \end{bmatrix}\boldsymbol{\omega}. \tag{F.28}$$

References

[1] H. Tillander, "Magnetic guidance of a catheter with articulated steel tip," *Acta radiologica*, vol. 35, no. 1, pp. 62–64, 1951.

[2] D. C. Giancoli, *Physics for Scientists and Engineers with Modern Physics*. Pearson Education, 2008.

[3] J. J. Fahie, *A History of Electric Telegraphy, to the year 1837*. London: E. and FN Spon, 1884.

[4] D. Gooding, "Faraday, thomson, and the concept of the magnetic field," *The British Journal for the History of Science*, vol. 13, no. 2, pp. 91–120, 1980.

[5] D. J. Griffiths, *Introduction to Electrodynamics*. American Association of Physics Teachers, 2005.

[6] J. Sikorski, I. Dawson, A. Denasi, E. E. Hekman, and S. Misra, "Introducing bigmag—a novel system for 3d magnetic actuation of flexible surgical manipulators," pp. 3594–3599, 2017.

[7] L. Yang, X. Du, E. Yu, D. Jin, and L. Zhang, "Deltamag: An electromagnetic manipulation system with parallel mobile coils," pp. 9814–9820, 2019.

[8] M. P. Kummer, J. J. Abbott, B. E. Kratochvil, R. Borer, A. Sengul, and B. J. Nelson, "Octomag: An electromagnetic system for 5-dof wireless micromanipulation," *IEEE Transactions on Robotics*, vol. 26, no. 6, pp. 1006–1017, 2010.

[9] J. J. Abbott, E. Diller, and A. J. Petruska, "Magnetic methods in robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, 2020.

[10] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, "Continuum robots for medical applications: A survey," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1261–1280, 2015.

[11] T. da Veiga, J. H. Chandler, P. Lloyd, G. Pittiglio, N. J. Wilkinson, A. K. Hoshiar, R. A. Harris, and P. Valdastri, "Challenges of continuum robots in clinical context: A review," *Progress in Biomedical Engineering*, vol. 2, no. 3, 2020.

[12] T. L. Thomas, V. K. Venkiteswaran, G. Ananthasuresh, and S. Misra, "A monolithic compliant continuum manipulator: A proof-of-concept study," *Journal of Mechanisms and Robotics*, vol. 12, no. 6, 2020.

[13] J. Sikorski, S. Mohanty, and S. Misra, "Milimac: Flexible catheter with miniaturized electromagnets as a small-footprint system for microrobotic tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5260–5267, 2020.

[14] G. Pittiglio, L. Barducci, J. W. Martin, J. C. Norton, C. A. Avizzano, K. L. Obstein, and P. Valdastri, "Magnetic levitation for soft-tethered capsule colonoscopy actuated with a single permanent magnet: A dynamic control approach," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1224–1231, 2019.

[15] Y. Kim, G. A. Parada, S. Liu, and X. Zhao, "Ferromagnetic soft continuum robots," *Science Robotics*, vol. 4, no. 33, 2019.

[16] P. Lloyd, A. K. Hoshiar, T. da Veiga, A. Attanasio, N. Marahrens, J. H. Chandler, and P. Valdastri, "A learnt approach for the design of magnetically actuated shape forming soft tentacle robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3937–3944, 2020.

[17] A. J. Petruska and B. J. Nelson, "Minimum bounds on the number of electromagnets required for remote magnetic manipulation," *IEEE Transactions on Robotics*, vol. 31, no. 3, pp. 714–722, 2015.

[18] S. Salmanipour and E. Diller, "Eight-degrees-of-freedom remote actuation of small magnetic mechanisms," pp. 1–6, 2018.

[19] J. Sikorski, C. M. Heunis, F. Franco, and S. Misra, "The armm system: An optimized mobile electromagnetic coil for non-linear actuation of flexible surgical instruments," *IEEE Transactions on Magnetics*, vol. 55, no. 9, pp. 1–9, 2019.

[20] C. Heunis, J. Sikorski, and S. Misra, "Flexible instruments for endovascular interventions: Improved magnetic steering, actuation, and image-guided surgical instruments," *IEEE Robotics and Automation Magazine*, vol. 25, no. 3, pp. 71–82, 2018.

[21] F. Ongaro, C. M. Heunis, and S. Misra, "Precise model-free spline-based approach for magnetic field mapping," *IEEE Magnetics Letters*, vol. 10, pp. 1–5, 2018.

[22] A. J. Petruska and J. J. Abbott, "Optimal permanent-magnet geometries for dipole field approximation," *IEEE Transactions on Magnetics*, vol. 49, no. 2, pp. 811–819, 2012.

[23] A. J. Petruska, J. Edelmann, and B. J. Nelson, "Model-based calibration for magnetic manipulation," *IEEE Transactions on Magnetics*, vol. 53, no. 7, pp. 1–6, 2017.

[24] J. Sikorski, A. Denasi, G. Bucchi, S. Scheggi, and S. Misra, "Vision-based 3-d control of magnetically actuated catheter using bigmag—an array of mobile electromagnetic coils," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 2, pp. 505–516, 2019.

[25] V. K. Venkiteswaran, J. Sikorski, and S. Misra, "Shape and contact force estimation of continuum manipulators using pseudo rigid body models," *Mechanism and Machine Theory*, vol. 139, pp. 34–45, 2019.

[26] D. C. Rucker and R. J. Webster III, "Statics and dynamics of continuum robots with general tendon routing and external loading," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1033–1044, 2011.

[27] J. Edelmann, A. J. Petruska, and B. J. Nelson, "Magnetic control of continuum devices," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 68–85, 2017.

[28] V. N. Le, N. H. Nguyen, K. Alameh, R. Weerasooriya, and P. Pratten, "Accurate modeling and positioning of a magnetically controlled catheter tip," *Medical Physics*, vol. 43, no. 2, pp. 650–663, 2016.

[29] S. Jeon, A. K. Hoshiar, K. Kim, S. Lee, E. Kim, S. Lee, J.-y. Kim, B. J. Nelson, H.-J. Cha, B.-J. Yi, *et al.*, "A magnetically controlled soft microrobot steering a guidewire in a three-dimensional phantom vascular network," *Soft Robotics*, vol. 6, no. 1, pp. 54–68, 2019.

[30] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer Science and Business Media, 2010.

[31] S. Antman, *Nonlinear Problems of Elasticity*. Springer-Verlag, 1995.

[32] J. Till, V. Aloi, and C. Rucker, "Real-time dynamics of soft and continuum robots based on cosserat rod models," *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 723–746, 2019.

[33] R. Fletcher, "A modified marquardt subroutine for non-linear least squares," *United Kingdom Atomic Energy Research Establishment, Theoretical Physics Division, Harwell, Berkshire*, 1971.

[34] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*. SIAM, 1995.

[35] J. Diebel, "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, vol. 58, no. 15-16, pp. 1–35, 2006.