Master Thesis

Optimizing Planar Arrays of Magnetic Sensors for Medical Robot Localization

Xiaowei Lin

Advised by

Prof. Dr. Pierre E. Dupont

Prof. Dr. Giovanni Pittiglio

Dupont Lab

Boston Children's Hospital, Harvard Medical School

Prof. Dr. Bradley J. Nelson

Multi-Scale Robotics Lab (MSRL)

Swiss Federal Institute of Technology Zurich (ETH)

October 2024

Acknowledgments

I would like to express my deepest gratitude to Prof. Pierre E. Dupont for giving me the opportunity to conduct my Master's thesis in his lab and for being an outstanding mentor. He's always easy to approach, even during the busiest time. His invaluable feedback and guidance were instrumental throughout the project.

My heartfelt thanks also go to my Co-Supervisor, Prof. Giovanni Pittiglio, for his unwavering support and insightful feedback. Our engaging discussions were vital to the project's success.

I am profoundly grateful to my ETH supervisor, Prof. Bradley J. Nelson, for enabling me to pursue my thesis abroad and always being available for guidance.

I would also like to thank my colleagues Phillip Tran, Max McCandless, Yuan Wang, Abhijit Mondal, and Namrata U. Nayar for their warm welcome and for making my experience at Boston Children's Hospital so enjoyable.

Lastly, I am deeply thankful to my friends and family for their unwavering support throughout this journey.

This thesis marks the culmination of my master's studies at ETH Zurich. Over the past three years, I've experienced profound growth, both academically and personally. It has been a transformative and vibrant chapter in my life. With the mindset and experiences I've gained in Zurich, the Bay Area, and Boston, I feel fully prepared and confident to embrace the next chapter of my journey.

Abstract

Robotic catheter systems have been widely used in the medical field, such as heart ablation. These systems aim to enhance navigation, increase catheter stability, and potentially reduce radiation exposure during procedures. To achieve accurate control of the catheter, shape information of the catheter is often required to close the control loop. A lightweight, non-invasive, yet reliable method to obtain the shape information of the catheter is to use permanent magnets. The magnet is usually put in the body of the catheter, e.g. the tip, and is tracked by an array of magnetic sensors. By tracking the position and orientation of the magnetic, we can reconstruct the shape of the catheter using geometry, e.g. a constant curvature model, a Bézier Curve, etc.

This thesis presents a new algorithm that tracks the magnet's position and orientation, proposes a strategy for optimizing the sensor placement for better algorithmic performance, and uses experiments to verify the outcome of the optimization. The optimized sensor array can be used to track the magnet's configuration with low latency and high accuracy.

Contents

List of Figures

List of Tables

Acronyms and Abbreviations

TEER	Transcatheter Edge-to-Edge Repair
TAVR	Transcatheter Aortic Valve Replacement
FBG	Fiber Bragg Gratings
DOF	Degree of Freedom
KKT conditions	Karush-Kuhn-Tucker conditions
BFGS algorithm	Broyden–Fletcher–Goldfarb–Shanno algorithm
SVD	Singular Value Decomposition
i.i.d.	Independent and Identically Distributed
PDF	Probability Density Function
CDF	Cumulative Distribution Function
GA	Genetic Algorithm

Chapter 1

Introduction

1.1 Motivation and problem definition

Robotic catheter systems are becoming increasingly advanced in the field of cardiac surgery, particularly in minimally invasive procedures. Compared to manual catheter systems, they provide surgeons with greater precision, flexibility, and control during delicate operations. Various kinds of heart surgery can be performed by a robotic catheter system. Here are some examples:

- Ablation for Arrhythmias [1]: Robotic catheters are used to navigate inside the heart to treat conditions like atrial fibrillation or other arrhythmias. They deliver energy (usually radiofrequency or cryotherapy) to specific heart tissues to create scars that block abnormal electrical signals.
- Mitral Valve Repair (Transcatheter Edge-to-Edge Repair - TEER) [2]: Robotic catheters can be used to repair leaking mitral valves by clipping the leaflets of the valve together to restore proper closure.
- Transcatheter Aortic Valve Replacement (TAVR) [3]: In TAVR, robotic catheters can help replace a diseased aortic valve without open-heart surgery. A new valve is delivered via a catheter and deployed at the site of the old valve.

Figure 1.1 visualizes a robotic catheter operating in a patient's heart while the patient is lying on the operating bed [4].

To achieve good control and navigation of the catheter, it is often beneficial to have the shape information of the catheter that closes the control loop. This can be achieved by fiber optic technologies, e.g. Fiber Bragg Gratings (FBGs), which insert an optical fiber into the catheter and use light deflection to sense the shape [5]. But this approach is very expensive. Another approach can be putting a permanent magnet at the tip of the catheter. By tracking the permanent magnet, we can reconstruct the shape of the catheter using some modeling techniques, e.g. the constant curvature model, Quadratic Bézier curve, etc [6]. There are several advantages of using the permanent magnet to provide the shape information of the robotic catheter:

- Simplified design: Permanent magnet systems operate passively, meaning they do not need embedded electronics or power sources inside the catheter.

Figure 1.1: A robotic catheter operating in human's heart

- Robustness and durability: Permanent magnets are highly durable and not as sensitive to bending, twisting, or mechanical stress as fiber optics.
- Fewer calibration and alignment challenges: Magnetic systems are generally easier to align and calibrate compared to fiber optics.
- Lower cost: Permanent magnets and magnetic sensors are generally much cheaper than a fiber optic shape sensing system.

Given the advantages, we would like to focus on using a permanent magnet to perform shape sensing of a robotic catheter. In particular, we are interested in using an array of magnetic sensors to track a permanent magnet operating in the heart of a patient, who is lying on the operating bed. The magnetic sensors are placed on the operating bed. The setup is illustrated in Figure 1.2.

To further define the problem, we model the heart workspace as a sphere with a 10cm diameter and the bottom of the sphere is 10cm away from the operating bed, where the sensors are placed. The setup is illustrated in Figure 1.3. In the example setup, there are four magnetic sensors placed on the operating bed, represented by the four dark pillars.

Take a closer look at the defined workspace in Figure 1.4. The permanent magnet operating in the spherical workspace is modeled as a magnetic dipole and generates the magnetic field. There is a local frame attached to the magnet, representing its configuration, i.e. position p and orientation R. The z-axis of the local frame aligns with the magnet direction, pointing from the south to the north pole. The magnetic field is captured by a number of magnetic sensors placed on the operating bed. Their coordinates frames are aligned with the world frame, so we can express every quantity with respect to the world frame. Without special notice, the quantities are expressed in the world frame.

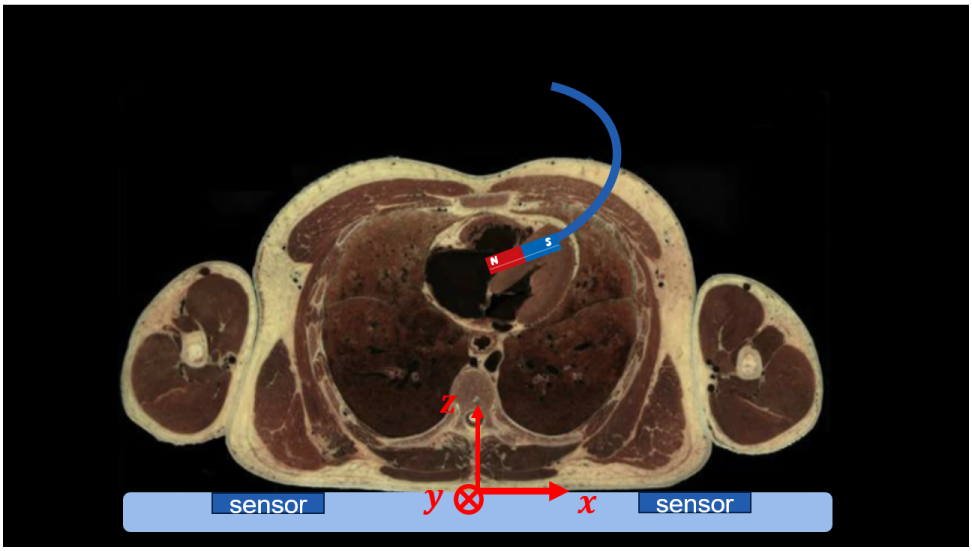

Figure 1.2: A permanent magnet mounted on the tip of a robotic catheter operating in the patient's heart; The magnet is tracked by an array of magnetic sensors on the operating bed

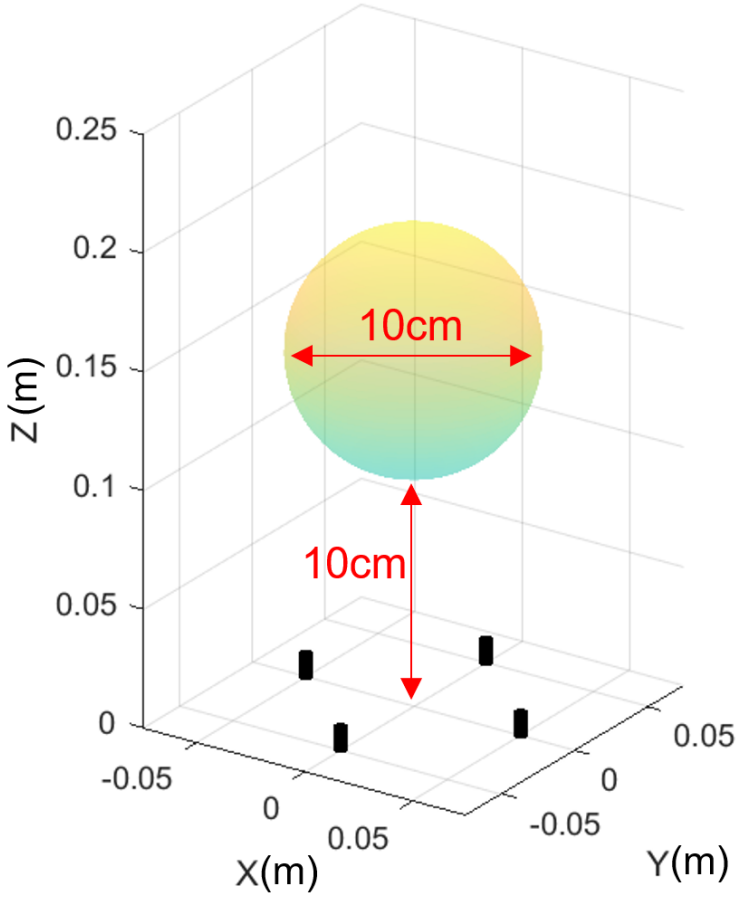

Figure 1.3: The heart is modeled as a 10cm diameter sphere whose bottom is 10cm away from the sensor array

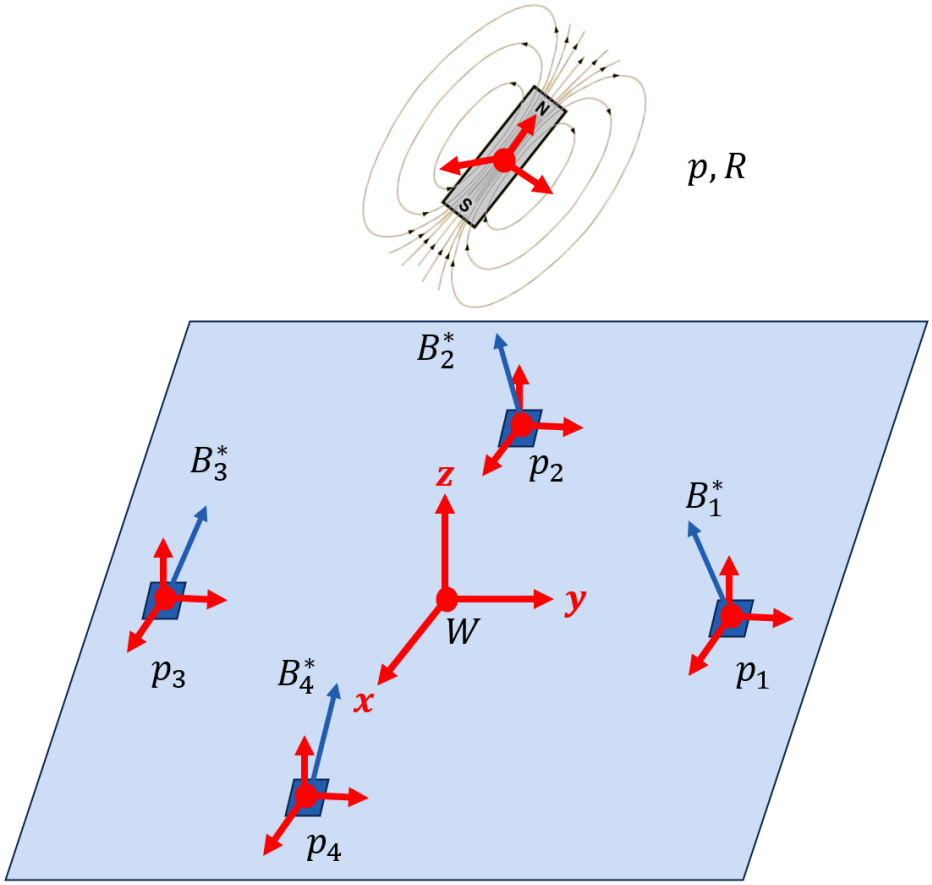

Figure 1.4: Detailed workspace illustration

We are interested in solving the magnet's configuration, p and R, from which we can reconstruct the shape of the catheter where the magnet is mounted. To do this, we need to measure the magnetic field generated by the magnet using m sensors on the operating bed and use it to solve for the magnet's configuration:

$$B^* = \begin{bmatrix} B_1^* \\ B_2^* \\ \vdots \\ B_m^* \end{bmatrix} \in \mathbb{R}^{3m} \rightarrow p, R \tag{1.1}$$

The magnet is a 5-DOF system because rotating around the magnet's z-axis does not change the magnetic field. So this DOF is not observable from the magnetic field. To solve a 5-DOF system, at least 5 measurements of the magnetic field are required. Using 3-axis magnetic sensors, which give 3 measurements individually, at least 2 3-axis sensors are required.

Researchers have been working on using a permanent magnet to perform shape-sensing of robotic catheters. In the next sections, various existing methods are introduced and in the last section of this Chapter, the contribution and the structure of this thesis are introduced.

Another interesting problem is to detect the small motion of the magnet from the change of magnetic field. For the same amount of small motion, we often want the change in magnetic field to be bigger than the noise level such that we can detect the change of magnet configuration from the noisy measurement.

1.2 Existing methods

Researchers have been focusing on solving the magnet's configuration from the magnetic field it generates. The magnetic field is usually measured by the sensor array structured as a grid. By tracking the configuration of the magnet on the catheter, the shape of the catheter can be reconstructed.

Changchun Zhang et al., 2017 [7] put one permanent magnet on the tip of the catheter. They used 8 sensors placed on a square to track the magnet. They represent the position using Euclidean coordinates and the orientation using the unit vector aligned with the body's z-axis. By tracking the magnet's configuration, the shape of the catheter is reconstructed using the Bézier Curve.

Jie Wang et al., 2017 [8] put two permanent magnets on the catheter and track the two magnets simultaneously using a sensor array of 16 sensors placed on a 4×4 grid. Like Changchun Zhang et al., they use Euclidean coordinates to represent the position and the unit vector that aligns with the body z-axis to represent its orientation. The magnets' configurations are solved using the Levenberg–Marquardt algorithm with the constraint that the orientation vector has to be norm 1. With the configurations of the two magnets, a Bézier Curve is used to reconstruct the shape of the catheter.

Cameron R. Taylor et al., 2019 [9] proposed a method to track multiple magnets. They use 16 sensors placed on a 4×4 grid to track the magnets. They used Euclidean coordinates to represent the position and two Euler angles to represent the 2-DOF orientation. The Jacobians of the magnetic field with respect to the position and orientation are analytically

derived, which is used in the Levenberg–Marquardt algorithm that solves the magnet's configuration.

Nordine Sebkhi et al., 2019 [10] developed a neural network that tracks the magnet based on magnetic field measurement. They used a grid of 24 magnetic sensors to track a permanent magnet, whose position is represented by Euclidean coordinates and orientation represented by two Euler angles.

As we can see from the previous research, researchers only focus on solving the magnet's configuration from the magnetic field captured by the sensor array, which is usually structured as a grid. They often used existing algorithms to solve the magnet's configuration, where the position is parametrized using the Euclidean coordinates, and the orientation is also parametrized in the Euclidean space, using the constrained unit vector or the Euler angles. In this thesis, we aim to develop the algorithm that solves the inverse problem from the magnetic field to the magnet configuration and optimize the sensor array configuration for better algorithmic performance, in terms of accuracy and speed. Analyzing the magnetic field model, we also understand where the estimation error comes from.

1.3 Contribution

We will see in the following chapter that the algorithms that solve the inverse problem require linearization of the magnetic field model with respect to the parametrization of position and orientation. From the previous section, we can see that when researchers parameterize the configuration of the magnet, they parameterize the magnet's orientation in Euclidean space, e.g. the constrained unit vector or the Euler angles. We found that these parametrizations may be problematic because the Euclidean space is not isomorphic to the space of orientation, i.e. the Special Orthogonal group $SO(3)$. The linearized magnetic field model also allows us to analyze how the magnetic field behaves locally around the linearization point, enabling the analysis for the estimation error. Thus, a proper linearization of the magnetic field model is required. The linearized model is further used in developing a new algorithm that solves the inverse problem, which leads to the requirement of optimizing the sensor array configuration for a better performance of the algorithm.

The rest of the thesis is structured as follows:

- In Chapter 2, the classic magnetic dipole model is introduced and its linearization with respect to position and orientation is introduced. Local parametrization of the orientation is introduced to account for the fact that the orientation parametrization in the Euclidean space is not isomorphic to the orientation in the $SO(3)$. The properly derived linearized model allows us to analyze how the change in position and orientation affects the change in the magnetic field locally.
- In Chapter 3, the linearized magnetic field model is used in developing the least square algorithm that solves the magnet configuration from the measured magnetic field. The classic algorithm is first introduced with the limitation that it assumes the unknown parameters lie in the space that is isomorphic to the Euclidean space, which is not the case for orientation. The classic algorithm is then adapted for orientation. The adapted algorithm is then compared to another existing approach that solves the non-isomorphism issue, i.e. over-parametrization.
- In Chapter 4, the sensor array configuration is optimized to achieve better perfor-

mance for the algorithm. Firstly, the criteria for optimization are derived, which are hypothesized to affect the algorithmic performance. Then, the sensor placement is optimized to improve the criteria.

- In Chapter 5, the different sensor configurations from the optimization are compared in experiments both in simulation and physically. It compares the performance of different sensor configurations with different levels of the hypothesized criteria. Based on the experiment results, conclusions on the hypotheses are drawn.
- In Chapter 6, the findings from previous chapters are summarized, emphasizing the contributions of the linearized magnetic dipole model, the adaptation of the least squares algorithm, and the optimization of sensor configurations. The impact of these developments on the performance of magnet localization algorithms is discussed, along with possible directions for future work.

Chapter 2

Magnetic Field Model

In this chapter, the magnetic dipole model is first introduced, which captures the characteristics of the magnetic field generated by a magnetic dipole. Then, the nonlinear dipole model is linearized with respect to position and orientation. In the end, we show how the linearized magnetic field model can be utilized to solve the inverse problem from the measured magnetic field to the magnet configuration and how it can be used to analyze the local behavior of the magnetic field.

2.1 Magnetic dipole model

In electromagnetism, a magnetic dipole is the limit of either a closed loop of electric current or a pair of poles as the size of the source is reduced to zero while keeping the magnetic moment constant. A magnet is an example of a pair of poles that generate a magnetic field and can be modeled as a magnetic dipole.

The magnetic dipole model describes the magnetic field at $p_i \in \mathbb{R}^3$ generated by the magnetic dipole at position $p \in \mathbb{R}^3$ with the dipole moment $m \in \mathbb{R}^3$:

$$B(r, m) = \frac{\mu_0}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)m \tag{2.1}$$

where r is the vector from p to p_i, i.e. $r = p_i - p$, $|r|$ (in m) is the length of the vector r, $\hat{r}$ is the normalized vector of r, i.e. $\hat{r} = \frac{r}{|r|}$, and $\mu_0 = 4\pi \times 10^{-7}$ H/m is the permeability of vacuum.

While the position of the magnetic dipole is explicitly expressed in the dipole model, its orientation is embedded in the magnetic moment vector m. The magnitude of the vector m is the magnet dipole moment $|m|$:

$$|m| = \frac{B_r V}{\mu_0} \tag{2.2}$$

where B_r (in T) is the magnetic flux density of the magnetic, which is related to its material; V (in m^3) is the volume of the magnet and $\mu_0 = 4\pi \times 10^{-7}$ H/m is the permeability of vacuum. (More clarification) As a result, the orientation of the magnet is a 2-DOF unit vector $\hat{m}$ that defines the direction the magnetic moment is pointing toward. Without loss

of generality, we align the z-axis unit vector e_z with the direction of the magnetic dipole. So the dipole's direction can be written as:

$$\hat{m} = Re_z \tag{2.3}$$

where R represents the orientation of the magnet. As a result, the magnetic field at p_i generated by a magnetic dipole at p with a dipole moment m whose direction is Re_z with $e_z = [0, 0, 1]^T$ can be re-written as:

$$B(r, m) = B(p, R) = \frac{B_r V}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)Re_z, r = p_i - p \tag{2.4}$$

2.2 The linear differential model

As introduced in 1.1, the two problems we are interested in solving are: 1. Given the magnetic field measured by the magnetic sensors, solve the inverse problem for the magnet position P and orientation R; 2. Enable the measurement of specified differential translation δp and orientation δR at a certain position and orientation of the magnet, p^* and R^*. Both of the tasks are difficult to perform or analyze because the magnetic dipole model is a nonlinear function. Thus, a proper approximation of the model is required.

For a nonlinear function $f(x)$, we can approximate the function at a certain point x_0 by a linear function using Taylor expansion:

$$f(x_0 + \delta x) \approx f(x_0) + f'(x_0)\delta x \tag{2.5}$$

This approximation is more accurate when the deviation from the point it's linearized around, δx, is smaller.

When a scalar function becomes a vector function, such as $B(p, R) \in \mathbb{R}^3$, and the variable is also a multi-dimensional vector, the derivative in 2.5 becomes the Jacobian and 2.5 becomes:

$$f(x_0 + \delta x) \approx f(x_0) + J(x_0)\delta x \tag{2.6}$$

where $f(x) \in \mathbb{R}^m$, $x \in \mathbb{R}^n$ and the Jacobian is defined as:

$$J(x) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \in \mathbb{R}^{m \times n} \tag{2.7}$$

There are several reasons for choosing to approximate the model with a linear model:

- It allows us to use the knowledge of linear algebra to perform analysis.
- The numerical method for solving the inverse problem introduced in 3 uses linearization for the objective function to update the guess for the variables. The linearized

dipole model fits naturally into the numerical method.

2.2.1 Linearization with respect to position

The Jacobian of the magnetic field with respect to the magnet's position can be derived using the chain rule [11]:

$$\frac{\partial B(p,R)}{\partial p} = \frac{\partial B(p,R)}{\partial r}\frac{\partial r}{\partial p} = -\frac{3B_rV}{4\pi|r|^4}((I-5\hat{r}\hat{r}^T)(\hat{r}^T\hat{m}) + \hat{m}\hat{r}^T + \hat{r}\hat{m}^T) \tag{2.8}$$

where $r = p_i - p$ and $\frac{\partial r}{\partial p} = -I$, $\hat{m} = Re_z$. When linearizing the model at a certain configuration, p_0 and R_0, the Jacobian 2.8 needs to be evaluated at p_0 and R_0:

$$\left.\frac{\partial B(p,R)}{\partial p}\right|_{p_0,R_0} \tag{2.9}$$

2.2.2 Linearization with respect to orientation

To obtain the linearization formula as 2.5 for the numerical algorithm or analysis of the linearized model, some parametrization of the rotation matrices and its corresponding mapping from the parameter space to $SO(3)$ is required. There are several ways to parameterize a rotation, such as the exponential coordinates with the exponential map, that maps the vectors in the Euclidean space $\mathbb{R}^3$ onto the three-dimension manifold $SO(3)$. The following discussion is based on the fact that rotations are represented using the exponential coordinates.

For a vector $\omega = [\omega_x, \omega_y, \omega_z]^T \in \mathbb{R}^3$, there exists a mapping that first maps the vector $w \in \mathbb{R}^3$ into the skew-symmetric matrix group $so(3)$ and then maps the element in $so(3)$ into the rotation matrix group $SO(3)$. This is the well-known Rodrigues formula:

$$R(\omega) = exp([\omega]_\times) = I + \frac{\sin|\omega|}{|\omega|}[\omega]_\times + \frac{1-\cos|\omega|}{|\omega|^2}[\omega]_\times^2 \tag{2.10}$$

where $[\omega]_\times$ is the skew-symmetric operator $[\cdot]_\times : \mathbb{R}^3 \to so(3)$ given by:

$$[\omega]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2.11}$$

The result from the Rodrigues formula $R(\omega)$ is a rotation matrix in $SO(3)$ representing a rotation along the axis $\frac{\omega}{|\omega|}$ with the angle $|\omega|$.

However, the Lie group $SO(3)$ is not isomorphic to the Euclidean space $\mathbb{R}^3$ globally [12], which means that all of these global parameterizations will exhibit singularities or other anomalies at various points in the parameter space. An example is that if we use the exponential coordinates to parameterize the orientation:

$$B(p,\omega) = \frac{B_rV}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)R(\omega)e_z \tag{2.12}$$

And compute the Jacobian of B with respect to ω:

$$J_\omega = \frac{\partial B}{\partial \omega} \tag{2.13}$$

We expect a one-dimension null space for J_ω and the null space should represent the rotation around the body's z-axis because the rotation around the body's z-axis does not change the magnetic field. However, using the global parametrization, the null space of the Jacobian is not a rotation around the z-axis. It means that the z-axis rotation changes the magnetic field, which is not physically correct. It prevents us from using the linearized model to analyze the behavior of the magnetic field.

Taylor & Kriegman, 1994 [13], proposed a local parametrization of the rotation. That is, at every point R_0 on the manifold $SO(3)$, a continuous, differentiable mapping between a neighborhood of R_0 on the manifold and an open set in $\mathbb{R}^3$ is constructed:

$$R(w) = R_0 exp([\omega]_\times),\ \omega \in \mathbb{R}^3, |\omega| < \pi \tag{2.14}$$

Equation 2.14 provides a one-to-one mapping between vectors in the open ball $|\omega| < \pi$ and a local region of the manifold $SO(3)$ centered around the point R_0. It effectively moves the origin of the rotation parametrization to R_0, i.e. $R(\omega = [0,0,0]^T) = R_0$. As a result, when we are calculating the derivative of the rotation at R_0, using the local parametrization, we only need to evaluate the derivative at the origin $\omega = [0,0,0]^T$.

Now we need to calculate the derivative of the rotation matrix with respect to the parametrization ω to obtain the Jacobian. Inspired by Gallego & Yezzi, 2014 [14], the derivation follows:

$$\frac{\partial R(\omega)e_z}{\partial \omega} = \frac{\partial R_0 exp(\omega)e_z}{\partial \omega} = \begin{bmatrix} \frac{\partial R_0 exp(\omega)e_z}{\partial \omega_x} & \frac{\partial R_0 exp(\omega)e_z}{\partial \omega_y} & \frac{\partial R_0 exp(\omega)e_z}{\partial \omega_z} \end{bmatrix} \tag{2.15}$$

To obtain the Jacobian at R_0, we need to evaluate 2.15 at $\omega = [0,0,0]^T$. Each element in the matrix is given by:

$$\left.\frac{\partial R_0 exp(\omega)e_z}{\partial \omega_i}\right|_{\omega=0} = R_0[e_i]_\times e_z\ i \in x, y, z \tag{2.16}$$

2.15 becomes:

$$\begin{bmatrix} R_0[e_x]_\times e_z & R_0[e_y]_\times e_z & R_0[e_z]_\times e_z \end{bmatrix} = R_0 \begin{bmatrix} [e_x]_\times e_z & [e_y]_\times e_z & [e_z]_\times e_z \end{bmatrix} = -R_0[e_z]_\times \tag{2.17}$$

We can obtain the Jacobian of the magnetic field with respect to the parametrization of the rotation at a certain rotation R_0. First, rewrite the rotation matrix using local parametrization:

$$B(p, R(\omega)) = \frac{B_r V}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)R_0 exp([\omega]_\times)e_z \tag{2.18}$$

Calculate the derivative with respect to ω and evaluate the outcome at $\omega = [0,0,0]^T$:

$$\frac{\partial B(p, R(\omega))}{\omega}\bigg|_{\omega=[0,0,0]^T} = -\frac{B_r V}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)R_0[e_z]_\times \tag{2.19}$$

When linearizing the model at a certain configuration, p_0 and R_0, the Jacobian 2.8 needs to be evaluated at p_0 and R_0, i.e. p_0 and $\omega = [0,0,0]^T$:

$$\frac{\partial B(p, R(\omega))}{\omega}\bigg|_{p_0,\omega=[0,0,0]^T} \tag{2.20}$$

Depending on the rotation convention, 2.14 and also be written as:

$$R(w) = exp([\omega]_\times)R_0. \tag{2.21}$$

In the former case, the new rotation $exp([\omega]_\times)$ is post-multiplied to the original rotation R_0, meaning it follows the intrinsic rotation convention; in the latter case, the rotation is pre-multiplied to the original rotation R_0, which follows the extrinsic rotation convention.

Notice that the rank of the Jacobian is 2 because there is one DOF of the magnet that cannot be observed, which is the rotation around the body's z-axis. According to the dipole model, this rotation will not change the magnetic field. And the null space of the Jacobian is the same as the null space of the matrix $[e_z]_\times$:

$$N(\frac{\partial B(p, R(\omega))}{\omega}\bigg|_{p_0,\omega=[0,0,0]^T}) = N([e_z]_\times) = \begin{bmatrix}0\\0\\1\end{bmatrix} \tag{2.22}$$

The null space corresponds to the rotation around the body's z-axis expressed in the local frame, which is valid for the intrinsic rotation. If the local parametrization is rewritten as 2.21, the null space of the Jacobian becomes the rotation around the body's z-axis expressed in the global frame. Since there is always a 1-dimensional null space, the Jacobian is always rank-deficient. It verifies the fact that the magnetic dipole is a 5-DOF system where the rotation around the body's z-axis will not change the magnetic field it generates.

2.2.3 Summary

With both Jacobian matrices derived, we can obtain the counterpart of 2.5 for the magnetic dipole model 2.4. When 2.4 is approximated by a linear model at p_0, R_0 and the rotation is locally parametrized with exponential coordinates $R(\omega) = R_0 exp([\omega]_\times)$, the expression becomes:

$$\begin{aligned} B(p_0 + \delta p, R(0 + \delta\omega)) = B(p_0, R_0) &+ \frac{\partial B(p, R(\omega))}{\partial p}\bigg|_{p_0,\omega=[0,0,0]^T} \delta p \\ &+ \frac{\partial B(p, R(\omega))}{\partial \omega}\bigg|_{p_0,\omega=[0,0,0]^T} \delta\omega \end{aligned} \tag{2.23}$$

Define:

$$J_p(p_0, R_0) = \left.\frac{\partial B(p, R(\omega))}{\partial p}\right|_{p_0, \omega=[0,0,0]^T} \tag{2.24}$$

$$J_R(p_0, R_0) = \left.\frac{\partial B(p, R(\omega))}{\partial \omega}\right|_{p_0, \omega=[0,0,0]^T} \tag{2.25}$$

2.23 can be rewritten as:

$$\begin{aligned} B(p_0 + \delta p, R(\delta\omega)) &= B(p_0, R_0) + J_p(p_0, R_0)\delta p + J_R(p_0, R_0)\delta\omega \\ &= B(p_0, R_0) + \underbrace{\begin{bmatrix} J_p(p_0, R_0) & J_R(p_0, R_0) \end{bmatrix}}_{J(p_0, R_0)} \begin{bmatrix} \delta p \\ \delta\omega \end{bmatrix} \end{aligned} \tag{2.26}$$

Similar to 2.5, this linear approximation is more accurate when δp and $\delta\omega$ are small.

2.3 Linearized Problem Definitions

With the linear differential model introduced, we can see how this model can be applied to the problems we are interested in solving: 1. Solve for p^* and R^* of a magnet given the magnetic field it generates, measured by the magnetic sensors, B^*. 2. Distinguish small motion δp and δR of the magnet happens at the neighborhood of p^* and R^*. Both require using the linear differential model.

2.3.1 The inverse problem

A magnet with the configuration p^* and R^* generates a ground truth magnetic field at different positions, which can be captured by m 3-axis magnetic sensors located at those positions, obtaining the ground truth magnetic field vector:

$$B^* = \begin{bmatrix} B_1^* \\ B_2^* \\ \vdots \\ B_m^* \end{bmatrix} \in \mathbb{R}^{3m} \tag{2.27}$$

The inverse problem is trying to obtain an estimation of $\hat{p}$ and $\hat{R}$ that is as close to p^* and R^* as possible such that the error between the magnetic field generated by $\hat{p}$ and $\hat{R}$ and the ground truth magnetic field B^* is minimized. Namely, the inverse problem is trying to solve the following minimization problem:

$$\hat{p}, \hat{R} = \min_{p,R} \left\| \begin{bmatrix} B_1^* \\ B_2^* \\ \vdots \\ B_m^* \end{bmatrix} - \begin{bmatrix} B_1(p, R) \\ B_2(p, R) \\ \vdots \\ B_m(p, R) \end{bmatrix} \right\|^2 \tag{2.28}$$

where B_j is calculated from the p and R, and the sensor position p_j through the magnetic dipole model 2.4.

Various types of least square methods can be used to solve this kind of problem, e.g. Newton's method, Gauss-Newton method, etc. Most of the algorithms require an initial guess of the independent variables, and updating the guess iteratively. At every step of the iteration, the algorithms require an approximation of the loss function $L(p, R)$, e.g. quadratic approximation for Newton's method and linear approximation for the Gauss-Newton method and use the approximated loss function to inform the new step.

Both of the aforementioned algorithms require approximating the loss function to at least the linear level. Notice that the loss function is the subtraction between a vector of magnetic field calculated from the magnetic dipole model and a constant ground truth magnetic field vector. It means that linearizing the loss function requires linearizing the magnetic dipole model, which is calculated in 2.26.

More details in different algorithms and how they can be adjusted to estimate both the position, which is in the Euclidean space, and orientation, which is in the Lie group of $SO(3)$, will be discussed in detail in 3.

2.3.2 Distinguishing differential motion

We can rewrite 2.26 by rearranging the terms:

$$\underbrace{\hat{B}(p^* + \delta p, R(\delta\omega)) - \bar{B}(p^*, R^*)}_{\delta B = \hat{B} - \bar{B}} = \underbrace{\begin{bmatrix} J_p(p^*, R^*) & J_R(p^*, R^*) \end{bmatrix}}_{J} \underbrace{\begin{bmatrix} \delta p \\ \delta\omega \end{bmatrix}}_{\delta x} \tag{2.29}$$

where $B(p^*, R^*)$ represents the magnetic field measured when the magnet is at p^* and R^* and $B(p^*+\delta p, R(\delta\omega))$ represents the measured field after the configuration is changed for δp and $\delta\omega$. Ideally, we would like the change in the measured magnetic field, δB, to be greater than the noise level of the magnetic sensors to tell there is a change in the configuration. By analyzing the property of the Jacobian matrix $J = \begin{bmatrix} J_p(p^*, R^*) & J_R(p^*, R^*) \end{bmatrix}$ we can ensure the desired performance. More details will be introduced in the following section.

2.4 Summary

In this chapter, the local parametrization of the orientation is introduced to obtain a properly linearized magnetic dipole model. The linearized model can be used for analyzing the magnetic field behavior when the magnet moves around a certain configuration. In the next chapter, we will show that the linearized model can also be used in the development of the least square algorithm that solves the inverse problem from the measured magnetic field to the magnet's configuration.

Chapter 3

Least Square Method

The least-square algorithm is used to solve for the magnet's position and orientation given the measured magnetic field. To solve for the magnet's configuration given the measured magnetic field $B^* \in \mathbb{R}^{3m}$ we are effectively looking for the magnet's configuration that minimizes the residual between the measurement and the estimated field, in the least square sense:

$$\begin{aligned} \hat{p}, \hat{R} &= min_{p,R} ||r(p,R)||_2^2 \\ r(p,R) &= \begin{bmatrix} r_1(p,R) \\ r_2(p,R) \\ \vdots \\ r_{3m}(p,R) \end{bmatrix} \\ r_i(p,R) &= B^* - B(p,R) \end{aligned} \tag{3.1}$$

In this Chapter, the classic Gauss-Newton algorithm will be first introduced, which assumes the unknown parameters lying in a space that is isomorphic to the Euclidean space. This assumption does not hold for the orientation, which is in $SO(3)$. Thus, the classic Gauss-Newton algorithm needs to be adapted using the local parametrization for orientation, as introduced in Section 2.2.2. The singularity issue can also be solved using over-parametrization, e.g. quaternion, turning the minimization problem into a constrained one. The constrained problem can be solved using the method of Lagrangian and Newton-like algorithm, which is implemented in MATLAB. In the end, the adapted Gauss-Newton algorithm will be compared to the one implemented in MATLAB.

3.1 Classic Gauss-Newton algorithm

The Gauss-Newton algorithm is a method to solve unconstrained nonlinear least square problems [15]. In this section, the general Gauss-Newton algorithm is introduced.

Given m objective functions, $\boldsymbol{r} = \begin{bmatrix} r_1 & r_2 & \dots & r_m \end{bmatrix}^T$, which are often the residual functions of some ground truth values, $y^* = \begin{bmatrix} y_1^* & y_2^* & \dots & y_m^* \end{bmatrix}^T$, and the values obtained from the model $F(x) = \begin{bmatrix} f_1(x) & f_2(x) & \dots & f_m(x) \end{bmatrix}^T$:

$$r_i(\boldsymbol{x}) = y_i^* - f_i(x) r = y^* - F(x) \tag{3.2}$$

where $x = (x_1, x_2, \ldots, x_n)$ are the unknown parameter we want to estimate. Gauss-Newton algorithm solves for x that brings the values obtained by the model F as close to the ground truth value y^* as possible, which can be measured by the square of the norm of the residual vector $\boldsymbol{r}$:

$$R(\boldsymbol{x}) = ||r(x)||_2^2 = \sum_{i=1}^{m} (y_i^* - f_i(\boldsymbol{x}))^2 \tag{3.3}$$

Gauss-Newton algorithm solves the minimization problem iteratively. Starting with an initial guess $\boldsymbol{x}^{(0)}$, at each iteration step k, it linearizes $S(\boldsymbol{x})$ at $\boldsymbol{x}_k$:

$$R(\boldsymbol{x}) \approx R(\boldsymbol{x}_k) + J_r(\boldsymbol{x}_k)\Delta\boldsymbol{x} \tag{3.4}$$

where the entries of the Jacobian J_r are:

$$(J_r)_{ij} = \frac{\partial r_i(\boldsymbol{x}_k)}{\partial \boldsymbol{x}_j} \tag{3.5}$$

Solve the approximated linear least square problem by equating the linearized objective to 0 and obtain the update step:

$$\begin{aligned} R(\boldsymbol{x}_k) + J_r(\boldsymbol{x}_k)\Delta\boldsymbol{x} &\triangleq 0 \\ \Rightarrow \Delta\boldsymbol{x} &= -J_r(\boldsymbol{x}_k)^\dagger R(\boldsymbol{x_k}) \end{aligned} \tag{3.6}$$

Update the guess using $\Delta\boldsymbol{x}$:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \Delta\boldsymbol{x} = \boldsymbol{x}_k - J_r(\boldsymbol{x}_k)^\dagger R(\boldsymbol{x_k}) \tag{3.7}$$

When the residual function is written in the form of Equation 3.2, $J_f = -J_r$, the update step can be rewritten as:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + J_f(\boldsymbol{x}_k)^\dagger R(\boldsymbol{x_k}) \tag{3.8}$$

where J_f is the Jacobian of the model f with respect to the variables $\boldsymbol{x}$.

The general minimization procedure is summarized in Algorithm 1 in pseudocode.

The Gauss-Newton algorithm converges to the local minima and the guarantees of it converging to a local minima are introduced by Dennis and Schnabel, 1996 [16]. The convergence rate can approach quadratic [15]. However, the algorithm may converge slowly or not at all if the initial guess is far from the minimum or the matrix J_r is ill-conditioned. It poses the requirement of optimizing the Jacobians, which will be discussed in Chapter 4.

The classic Gauss-Newton algorithm assumes the unknown parameter lies in the space that is isomorphic to the Euclidean space. The update step is computed and carried out in the Euclidean space, as shown in Figure 3.1. It works naturally for the position because it is in the Euclidean space. However, for orientation, even though we can find some parametrization in the Euclidean space to represent the orientation, e.g. the exponential coordinates $\omega \in \mathbb{R}^3$ or the Euler angles in $\mathbb{R}^3$, the orientation lives in the Lie group $SO(3)$, which is not isomorphic to the Euclidean space. It means that all of the global

Algorithm 1 Gauss-Newton method

Require: y^* ▷ Ground truth
x_0 ▷ Initial guess
c ▷ Exit condition
n ▷ Maximum iteration

$k \leftarrow 0$ ▷ Initialization
$R(x) \leftarrow y^* - f(x)$ ▷ The residual function $R(x)$
while true **do**
  $R(x) \leftarrow R(x_k) - J_f(x_k)\Delta x$ ▷ Linear approximation of $R(x)$
  $\Delta x \leftarrow J_f(x_k)^\dagger R(x_k)$ ▷ Obtain the update step

  if $\Delta x < c$ **then break** ▷ Exit the loop if the update step is small enough
  else
    $x_{k+1} \leftarrow x_k + \Delta x$ ▷ Update x_k
  end if

  $k \leftarrow k + 1$
  if $i > n$ **then**
    break ▷ Exit the loop if maximum iteration is reached
  end if
end while

parameterizations in the Euclidean space will suffer from singularities or other kinds of anomalies. One example is the null space issue introduced in Section 2.2.2. The property of not being isomorphic is also crucial for the minimization procedure because the descent direction computed in the Euclidean space $\mathbb{R}^3$ is not necessarily the descent direction in $SO(3)$. As a result, the algorithm might not converge to the local minima. To solve this issue, the classic Gauss-Newton algorithm needs to be adapted to be used for estimating the orientation.

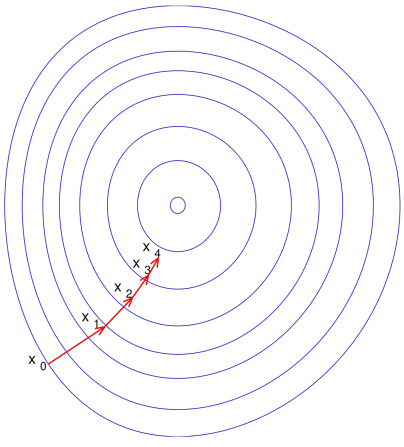

Figure 3.1: Error minimization in the Euclidean space

3.2 Gauss-Newton on manifold

To solve the issue of singularity due to the non-isomorphism, Taylor & Kriegman, 1994 [13]'s proposed solution for the orientation is to forgo the global parameterization in favor of an atlas of local parameterizations, derived from the exponential coordinates $\omega \in \mathbb{R}^3$. That is, at every point R_k on the manifold $SO(3)$, construct a continuous, invertible differentiable mapping between an open set in $\mathbb{R}^3$ and a neighborhood of R_k on the manifold:

$$R(w) = R_k exp([\omega]_\times),\ \omega \in \mathbb{R}^3, |\omega| < \pi \tag{3.9}$$

While the example algorithm provided by Taylor & Kriegman, 1994 [13] is the Newton's method, it can be adapted to the Gauss-Newton algorithm without computing the Hessian in every iteration. The following description of the algorithm will try to find both position p and orientation R of the magnet from the measured magnetic field.

Given the magnetic field measured by m 3-axis sensors, $B^* = \begin{bmatrix} B_1^* & B_2^* & \dots & B_m^* \end{bmatrix} \in \mathbb{R}^{3m}$, we would like to find p^* and R^* of the magnet solve the following minimization problem:

$$min_{p,R} ||r(p,R)||_2^2 \tag{3.10}$$

where $\boldsymbol{r} = (r_1, r_2, \dots, r_m)$ and

$$r_i(p,R) = B_i^* - f_i(p,R) \tag{3.11}$$

where $f_i(p,R)$ is the magnetic dipole model that represents the magnetic field generated by a magnet at position p and orientation R at the sensor i located at p_i, as described in Equation 2.4.

The entire minimization procedure is outlined in Algorithm 2 in pseudocode. There are several differences between between the Gauss-Newton on manifold algorithm and the classic Gauss-Newton algorithm:

- In every iteration k, the Gauss-Newton on manifold algorithm is effectively approximating a different objective function:

$$r(p, R_k \exp([\omega]_\times)) \tag{3.12}$$

- In every iteration, the orientation Jacobian is evaluated at the origin $\omega = 0$.
- When updating the orientation, it updates the ω in the tangent space of R_k and remap the updated ω back to $SO(3)$ using the exponential map:

$$R_{k+1} = R_k \exp([\Delta\omega]_\times) \tag{3.13}$$

  The update procedure is illustrated in Figure 3.2.

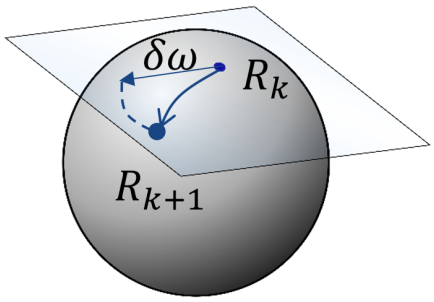

Figure 3.2: Error minimization in $SO(3)$

Algorithm 2 Gauss-Newton on manifold

Require: B^* ▷ Ground truth

Require: p_0, R_0 ▷ Initial guess
c_p, c_R ▷ Exit condition
n ▷ Maximum iteration

$k \leftarrow 0$ ▷ Initialization
$R(x) \leftarrow B^* - B(p, R(\omega)$ ▷ The residual function $R(x)$
while true **do**
$\boldsymbol{r}(p, R) \leftarrow \boldsymbol{r}(p, R_k \exp([\omega]_\times))$ ▷ Local parametrization of the orientation

$J_p(p_k, R_k) \leftarrow \frac{\partial \boldsymbol{r}}{\partial p}|_{p=p_k,\omega=0} = -\frac{\partial f}{\partial p}|_{p=p_k,\omega=0}$ ▷ 2.9
$J_R(p_k, R_k) \leftarrow \frac{\partial \boldsymbol{r}}{\partial \omega}|_{p=p_k,\omega=0} = -\frac{\partial f}{\partial \omega}|_{p=p_k,\omega=0}$ ▷ 2.21
$r \leftarrow r(p_k, R_k)$ ▷ Calculate the residual

$\Delta p \leftarrow -J_p^\dagger r$ ▷ Update step on p
$\Delta\omega \leftarrow -J_R^\dagger r$ ▷ Update step of R

if $\Delta p < c_p$ and $\Delta\omega < c_R$ **then**
break ▷ Exit the loop if the update step is small enough
else
$p_{k+1} \leftarrow p_k + \Delta p$ ▷ Update p_0
$R_{k+1} \leftarrow R_k \exp([\Delta\omega]_\times)$ ▷ Update R_0
end if

$k \leftarrow k + 1$ ▷ Count the iteration
if $k > n$ **then**
break ▷ Exit the loop if maximum iteration is reached
end if
end while

3.3 Method of Lagrange multipliers and Newton's method

Another approach to solve the singularity issue parametrizing the orientation using a vector in $\mathbb{R}^3$ is over-parametrization. For example, unit quaternion $q \in S^3$ that has 4 variables can be used to parameterize the orientation. However, due to the over-parametrization, further constraints need to be satisfied, e.g. $||q||_2 = 1$ for the unit quaternion. If this is the case, directly applying Equation 3.8 to update the parameter breaks the constraints. To overcome this problem, we can turn the unconstrained optimization problem into a constrained optimization problem and solve the new optimization problem:

$$\begin{aligned} \min_x ||R(x)||_2^2,\ x = \begin{bmatrix} p \\ q \end{bmatrix} \in \mathbb{R}^7 \\ \text{subject to } ||q||_2 = 1 \end{aligned} \tag{3.14}$$

The idea to solve this constrained optimization problem is to turn it into an unconstrained problem and use the methods of solving unconstrained problems, e.g. Quasi-Newton methods, to solve it. The problem described in Equation 3.14 can be turned into an unconstrained problem using the Lagrangian multipliers. Using the Lagrangian multipliers, the Lagrangian function of Problem 3.14:

$$\mathcal{L}(x, \lambda) = ||r(x)||_2^2 + \lambda ceq(x) \tag{3.15}$$

where λ is the Lagrangian multiplier for the equality constraint and the constrained is re-written as:

$$ceq(x) = ||q||_2 - 1 = 0 \tag{3.16}$$

To find the optima, i.e. necessarily the stationary point of $||r(x)||_2^2$, the gradient of the Lagrangian is taken:

$$\nabla_x \mathcal{L}(x, \lambda) = 2J(x)^T r(x) + J_E^T(x)\lambda \tag{3.17}$$

where J(x) is the Jacobian of the objective function $r(x)$:

$$J_{i,j}(x) = \frac{\partial r_i(x)}{\partial x_j} \tag{3.18}$$

and $J_E(x)$ is the Jacobian of the equality constraint function $ceq(x)$.

The Karush-Kuhn-Tucker (KKT) conditions provide a necessary condition for a point x in the variable space to be the optima. And the KKT conditions for the problem 3.14 are:

$$\begin{aligned} \nabla_x \mathcal{L}(x, \lambda) = 2J(x)^T r(x) + J_E^T(x)\lambda = 0 \\ ceq(x) = 0 \end{aligned} \tag{3.19}$$

We can use either Newton's method or the Gauss-Newton method to solve Equation 3.19. The interior point method provided by MATLAB uses the Gauss-Newton method together with the BFGS Hessian approximation to solve Equation 3.19. In the iteration, the Hessian of the Lagrangian function 3.15 is required but evaluating the Hessian is computationally expensive. Thus, the BFGS Hessian approximation is used to approximate the Hessian. It also starts with an initial guess for the Hessian, which is usually chosen to be the identity

matrix, and updates the guess iteratively. For the readers interested in the details of MATLAB's algorithm, please refer to the MATLAB'S documentation [17].

The method of Lagrange multipliers provides a way to solve an optimization problem with constraint, which can be applied to the problem of solving the inverse problem described in Section 2.3.1. To apply this method, the orientation of the magnet can be parametrized using the 4-dimension quaternion with a unitary constraint. MATLAB provides the optimization toolbox that implements this algorithm.

3.4 Comparison

Finally, we compare the performance of MATLAB's algorithm and the Gauss-Newton on manifold method. For both methods, four sensors are placed on the periphery of a circle of 25cm.

3.4.1 Test settings

The magnet is placed within the spherical workspace described in Chapter 1. Spheres of different radii are considered within the 5cm-radius sphere, i.e. $r \in \{1, 2, 3, 4, 5\}$cm. On each sphere, 42 positions are sampled as shown in Figure 3.3. In total, $5 \times 42 = 210$ positions are considered.

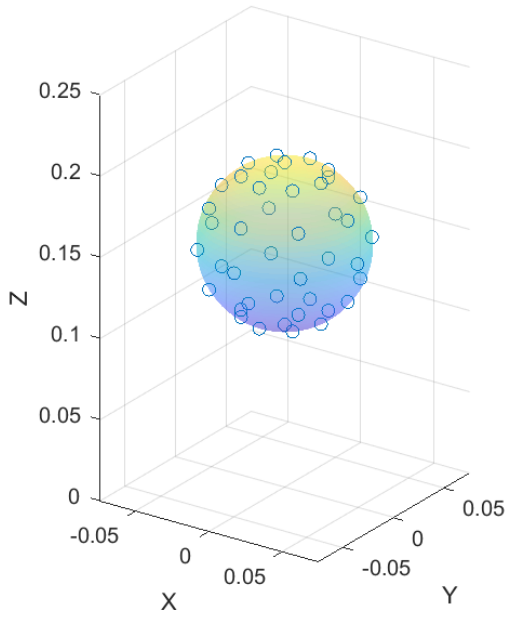

Figure 3.3: Nominal cone of various orientations

At different positions, various orientations are considered. Since both algorithms converge to the optima only when the initial guess is close enough to it, the orientations at each position are chosen to be only varying from the initial guess by at most 60°. In other words, various orientations at a certain position form a cone-like configuration space, as shown in Figure 3.4 with the center vector indicated in red. The cone's orientation can also be represented by the center vector. The cone is rotated around x or y for multiples

of 45° to cover different orientation workspace, with examples shown in Figure 3.5 and 3.6.

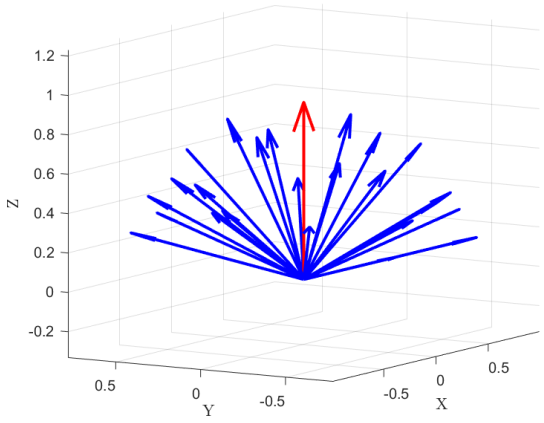

Figure 3.4: Nominal cone of various orientations

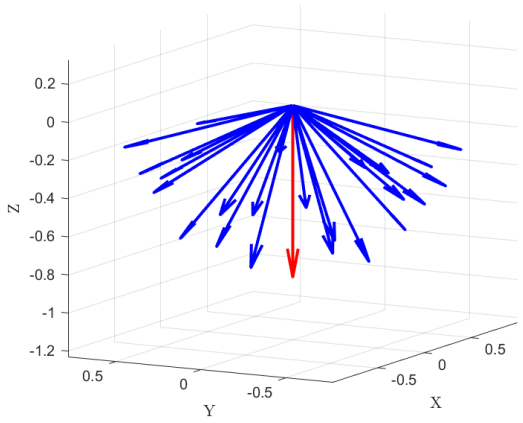

Figure 3.5: Cone rotated 180° around the x-axis

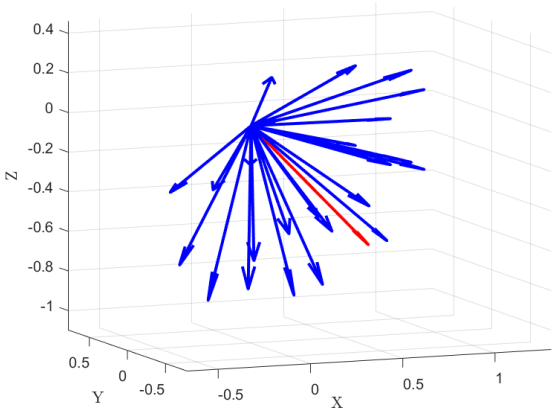

Figure 3.6: Cone rotated 135° around the y-axis

Each cone direction defines one set of test orientations on the selected positions. There are 25 orientations chosen in each cone at different positions, making it in total $25 \times 210 = 5250$ configurations in one test set. Various cone orientations are chosen to cover the whole workspace. There are 8 independent x-axis rotations and 8 independent y-axis rotations, resulting in 16 different test sets. For all the test sets, the initial guess for the position is the center of the spherical workspace. The initial guess for the orientation is chosen to be the center vector of the cone, which is different for each test set.

For each test set, both algorithms are run to solve for all of the 5250 test configurations, and the total time for solving all the configurations is recorded.

Results

The average time for solving one test set of 5250 configurations is recorded for both algorithms and summarized in Table.

	MATLAB	Gauss-Newton on manifold
Average time for solving 5250 configurations	180.39	1.68

Table 3.1: Comparison of average time for solving 5250 configurations for MATLAB's algorithm and Gauss-Newton on manifold

From the data, we can observe that using the Gauss-Newton on manifold method, the average time for solving 5250 configurations is 1.68s, which is 0.32ms for one configuration and the refresh rate is $\sim$ 3000Hz. MATLAB's algorithm using the Lagrangian multiplier and quasi-Newton method takes 180s to solve the 5250 configurations, which is 34ms for one configuration and the refresh rate is $\sim$ 30Hz. The Gauss-Newton on manifold method is $\sim$ 100 times faster than MATLAB's algorithm, which is beneficial for real-time application.

3.5 Summary

In this chapter, the classic Gauss-Newton algorithm is first introduced with its limitation on estimating the unknown parameter that is not in the Euclidean space, e.g. the orientation in $SO(3)$. To estimate the orientation of the magnet, the algorithm has been adapted with the local parametrization of the orientation, which uses the linearized magnetic field model derived in Chapter 2. Comparing the adapted Gauss-Newton on manifold algorithm to the algorithm implemented in MATLAB, where the orientation can be parametrized using quaternions, the Gauss-Newton on manifold algorithm is found to be 100 times faster than the MATLAB algorithm. The comparison is done by naively putting the sensors on the circle of a 25cm radius. In the next chapter, the sensor configuration is optimized for better performance of the algorithm, in terms of its accuracy and speed.

Chapter 4

Sensor Array Optimization

With the algorithm for solving the magnet's configuration based on the sensor readings introduced, the next question is how to place the sensors to solve the magnet's configuration better or distinguish small motion from noise. These requirements lead to the optimization of the sensor placement. In this chapter, the optimization criteria are introduced and related to the algorithm's performance and small motion detection. Then a baseline configuration is introduced and compared to the ones obtained by the global optimization of MATLAB.

4.1 Criteria for optimization

As described in 2.3.1, both problems we are interested in solving require local linearization of the magnetic dipole model. The linear model, which consists of the Jacobian matrix, allows us to: 1. estimate the algorithm's performance that solves the inverse problem; 2. estimate the ability of the sensors to distinguish differential motion. By analyzing the properties of the Jacobian and optimizing it by placing the sensor wisely, we hope the inverse problem can be solved faster and more accurately, and finer differential motion can be distinguished from the noise of the magnetic sensors.

For a certain sensor configuration consisting of m sensors located at positions $p_j, j \in \{1, 2, \ldots, m\}$ and a certain magnet configuration p^* and R^* in the spherical magnet workspace, as shown in Chapter 1, each sensor captures the magnetic field generated by the magnet and the magnetic field vector can be obtained:

$$B = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_m \end{bmatrix} \in \mathbb{R}^{3m} \tag{4.1}$$

where B is a vector of $3m$ nonlinear functions consisting of m 3-dimension nonlinear functions $B_j, j \in 1, 2, \ldots, m$. Each B_j can be obtained from the magnetic dipole model:

$$B_j(p, R) = \frac{B_r V}{4\pi |r|^3}(3\hat{r}\hat{r}^T - I)Re_z, r = p_j - p \tag{4.2}$$

Linearizing the function vector B at p^* and R^* gives:

$$\begin{bmatrix} \hat{B_1}(p^* + \delta p, R(\delta\omega)) \\ \hat{B_2}(p^* + \delta p, R(\delta\omega)) \\ \vdots \\ \hat{B_m}(p^* + \delta p, R(\delta\omega)) \end{bmatrix} \approx \begin{bmatrix} \bar{B_1}(p^*, R^*) \\ \bar{B_2}(p^*, R^*) \\ \vdots \\ \bar{B_m}(p^*, R^*) \end{bmatrix} + \underbrace{\begin{bmatrix} \begin{bmatrix} J_{p1}(p^*, R^*) & J_{R1}(p^*, R^*) \end{bmatrix} \\ \begin{bmatrix} J_{p2}(p^*, R^*) & J_{R2}(p^*, R^*) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} J_{pm}(p^*, R^*) & J_{Rm}(p^*, R^*) \end{bmatrix} \end{bmatrix}}_{J(p^*, R^*)} \begin{bmatrix} \delta p \\ \delta\omega \end{bmatrix} \tag{4.3}$$

which is stacking the linearized magnetic dipole model derived in 2.26 for m sensors. And the Jacobian $J(p^*, R^*)$ stacks individual Jacobian in 2.26 for m sensors. Each Jacobian has the dimension $\begin{bmatrix} J_{pi}(p^*, R^*) & J_{Ri}(p^*, R^*) \end{bmatrix} \in \mathbb{R}^{3\times 6}$, resulting in the stacked Jacobian with dimension $J(p^*, R^*) \in \mathbb{R}^{3m\times 6}$.

This Jacobian is the same one that is being pseudo-inversed in the Gauss-Newton method described in Section 3.2, whose property would affect the convergence quality of the algorithm. It also relates the differential motion δp and δR to the change of magnetic field measurement in m sensors, Equation 4.1, whose property allows us to estimate how differential motion would change the output. These facts make it valuable to study its properties and optimize the Jacobian for better performance of the sensors.

4.1.1 Singular value decomposition

Singular value decomposition is a factorization of a matrix $J \in \mathbb{R}^{m\times n}$ into the multiplication of three matrices: an orthonormal matrix $U \in \mathbb{R}^{m\times m}$, a diagonal matrix $S \in \mathbb{R}^{m\times n}$, and the transpose of another orthonormal matrix $V \in \mathbb{R}^{n\times n}$ [18]. The decomposition can be represented as:

$$J = USV^T \tag{4.4}$$

where $U^TU = I$, $V^TV = I$. The matrix $S \in \mathbb{R}^{m\times n}$ only has non-zero elements on its diagonal. So $S_{i,j} = 0$ if $i \neq j$. The remaining elements $\sigma_1 = S_{1,1}$, $\sigma_2 = S_{2,2}$, ..., $\sigma_r = S_{r,r}$ are known as the singular values and they have the property:

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 \tag{4.5}$$

where $r \leq \min\{m, n\}$ is the rank of the matrix J. So the number of non-zero elements on the diagonal of S represents the rank.

The columns of U and V are called the left and right singular vectors of A. They form an orthonormal basis for $\mathbb{R}^n$ and $\mathbb{R}^m$. In the case where J is rank-deficient, i.e. $r < \min\{m, n\}$, the first r columns of U spans the column space of J and the last $n - r$ columns of V spans the null space of J:

$$\begin{aligned} C(J) &= span\{u_1, u_2., ..., u_r\} \\ N(J) &= span\{v_{r+1}, v_{r+2}, ..., v_n\} \end{aligned} \tag{4.6}$$

v_{r+1}, v_{r+2}, ..., v_n represent the basis for the vector to collapse to 0 through the linear transformation Jv. The remaining columns of V, i.e. v_i $i \in 1, 2, \ldots, r$ represent the basis

for the vector to be stretched by the amount of the corresponding singular values, σ_i, through the linear transformation Jv.

4.1.2 The minimum non-zero singular value and sensor noise

We now consider the Jacobian, $J(p^*, R^*) \in \mathbb{R}^{3m\times 6}$, in the stacked linearized magnetic dipole model 4.3. As derived in 2.2.2, the individual J_{Ri} has a 1-dimension null space spanned by $[0, 0, 1]^T$ if the intrinsic rotation convention is used, which represents that the rotation around the body's z-axis will not change the magnetic field. It results in a 1-dimension null space for $J(p^*, R^*)$, spanned by $[0, 0, 0, 0, 0, 1]^T$. As a result, the Jacobian is rank 5 with 5 non-zero singular values $\{\sigma_1, \sigma_2, \ldots, \sigma_5\}$.

We are interested in the minimum non-zero singular value $\sigma_{min} = \sigma_5$ because it represents the minimum stretch applied to a vector that is in the orthogonal subspace of the null space.

Rewriting 4.3 as:

$$\begin{aligned} \hat{B} - \bar{B} &\approx J \begin{bmatrix} \delta p \\ \delta \omega \end{bmatrix} \\ \Rightarrow \delta B &\approx J \delta x \end{aligned} \tag{4.7}$$

where $\bar{B}$ is the magnetic field for $\bar{x}$ and $\hat{B}$ is the magnetic field after the configuration deviates for δx. We can obtain:

$$\sigma_{min} = \min_{\delta x \in N(J)^\perp} \frac{||J\delta x||}{||\delta x||} = \min_{\delta x \in N(J)^\perp} \frac{||\delta B||}{||\delta x||} \tag{4.8}$$

It allows us to estimate the least change in the magnetic field vector given a change in the configuration that does not consist of any z-axis rotation. Take $\delta x = v_5 \in N(J)^\perp$ for example. v_5 is the fifth right singular vector, whose norm is 1. It represents the motion that produces the minimum change of magnetic field, i.e. the differential motion that is the hardest to distinguish from noise. And the resulting δB by such a movement is:

$$||\delta B|| = \sigma_{min}||\delta x|| = \sigma_{min} * 1 = \sigma_{min} \tag{4.9}$$

The motion in the direction of v_5 for the amount of 1 is scaled down the most by the minimum singular value σ_{min}, resulting in the minimum change of the magnetic field. When there is noise in the measurement, what causes the change of magnetic field - the magnet's motion or noise, can be unclear. It leads to the first hypothesis:

Hypothesis 1 - v_{min} is the hardest DOF to distinguish when there is noise. When the algorithm tries to solve the magnet configuration from the noisy measurement, it will make the most error in the direction of v_{min}.

This hypothesis can be verified if we project the error vector on each of the singular vectors and compare their norm. The second hypothesis naturally follows:

Hypothesis 2 - If σ_{min} is larger, the same amount of motion in the direction of v_{min} will produce a larger $||\delta B||$, making it more robust to noise. As a result, the error of estimating this configuration from the noisy magnetic field measurement will be smaller.

To further analyze how the noise affects distinguishing small motion, we consider the case where the measurements are corrupted by noise, the measured change of magnetic field can be re-written as:

$$\delta\hat{B} = \begin{bmatrix} \hat{B_1} + \hat{\varepsilon_1} \\ \hat{B_2} + \hat{\varepsilon_2} \\ \vdots \\ \hat{B_m} + \hat{\varepsilon_m} \end{bmatrix} - \begin{bmatrix} \bar{B_1} + \bar{\varepsilon_1} \\ \bar{B_2} + \bar{\varepsilon_2} \\ \vdots \\ \bar{B_m} + \bar{\varepsilon_m} \end{bmatrix} = \begin{bmatrix} \hat{B_1} - \bar{B_1} \\ \hat{B_2} - \bar{B_2} \\ \vdots \\ \hat{B_m} - \bar{B_m} \end{bmatrix} + \begin{bmatrix} \hat{\varepsilon_1} - \bar{\varepsilon_1} \\ \hat{\varepsilon_2} - \bar{\varepsilon_2} \\ \vdots \\ \hat{\varepsilon_m} - \bar{\varepsilon_m} \end{bmatrix} = \delta B + E \in \mathbb{R}^{3m} \quad (4.10)$$

where $\delta B \approx J\delta x$, $\hat{\varepsilon_i}$ and $\bar{\varepsilon_i}$ is a vector consisting of three independent and identically distributed(i.i.d.) random variables representing the noise level on each axis of the sensor, following a Gaussian distribution of mean 0 and variance ξ^2:

$$\begin{aligned} \hat{\varepsilon_i} &= \begin{bmatrix} \hat{\epsilon}_{i,1} \\ \hat{\epsilon}_{i,2} \\ \hat{\epsilon}_{i,3} \end{bmatrix}, \ \hat{\epsilon_{i,j}} \sim N(0, \xi^2) \\ \bar{\varepsilon_i} &= \begin{bmatrix} \bar{\epsilon}_{i,1} \\ \bar{\epsilon}_{i,2} \\ \bar{\epsilon}_{i,3} \end{bmatrix}, \ \hat{\epsilon_{i,j}} \sim N(0, \xi^2) \end{aligned} \quad (4.11)$$

The i.i.d. assumption assumes each axis has the same noise level and each axis's measurement is independent to each other.

As a result, each element of the noise vector $E \in \mathbb{R}^{3m}$ is a subtraction of two i.i.d. random variables:

$$\epsilon_{i,j} = \hat{\epsilon}_{i,j} - \bar{\epsilon}_{i,j} \sim N(0, 2\xi^2), \ i \in \{1, 2, \ldots, m\}, \ j \in \{1, 2, 3\} \quad (4.12)$$

So the noise vector E is a $3m$ dimensional random vector whose elements are i.i.d. Gaussian variables with mean 0 and variance $2\xi^2$. Ideally, we want $||\delta B|| \gg ||E||$ such that the $||\delta B||$ generated by the small motion is distinguishable from the noise. We are interested in the norm of the noise vector $||E||$, which follows a scaled chi distribution, with cumulative distribution function can be analytically derived. For simplicity of notation, we substitute $2\xi^2$ with c and $3m$ with n for the derivation

$$||E||^2 = \sum_{i,j} \epsilon_{i,j}^2, \ \epsilon_{i,j} \sim N(0, c) \quad (4.13)$$

This sum follows a scaled chi-squared distribution:

$$\frac{1}{c}||E||^2 \sim \chi^2(n) \quad (4.14)$$

The PDF of the standard chi-square distribution with n DOFs:

$$f_{\chi^2(n)}(x) = \frac{x^{n/2-1}\exp(-x/2)}{2^{n/2}\Gamma(n/2)} \quad (4.15)$$

where $\Gamma(\cdot)$ is the Gamma function. For the scaled chi-squared distribution $||v||^2 = c\chi^2(x)$, where $c = c$, the PDF is scaled accordingly:

$$f_{||E||^2}(x) = \frac{1}{c} f_{\chi^2(n)}(\frac{x}{c}) = \frac{x^{n-1}\exp(-x/(2c)}{2^{n/2}c^{n/2}\Gamma(n/2)} \tag{4.16}$$

To obtain the PDF of $||E||$, we use the change of variable $y = \sqrt{x}$. The relationship between the PDFs of x and y is given by:

$$f_{||E||}(y) = f_{||E||^2}(y^2)|\frac{d}{dy}(y^2)| = f_{||v||^2}(y^2)2y \tag{4.17}$$

Substituting $x = y^2$ into the PDF of $||v||^2$, we get:

$$f_{||E||}(y) = \frac{y^{n-1}\exp(-y^2/(2c))}{2^{(n/2)-1}c^{n/2}\Gamma(n/2)} \tag{4.18}$$

To obtain the CDF, we integrate the PDF from 0 to z:

$$\begin{aligned} F_{||E||}(z) = P(||E|| < z) &= \int_0^z f_{||E||}(y)dy \\ &= \int_0^z \frac{y^{n-1}\exp(-y^2/(2c))}{2^{(n/2)-1}c^{n/2}\Gamma(n/2)}dy \end{aligned} \tag{4.19}$$

We can perform a change of variables to simplify the integral. Let:

$$u = \frac{y^2}{2c} \Rightarrow y = \sqrt{2cu} \Rightarrow dy = \frac{\sqrt{2c}}{2\sqrt{u}}du \tag{4.20}$$

Substitute these into the integral:

$$F_{||E||}(z) = \int_0^{z^2/(2c)} \frac{(2cu)^{(n/2)-1}\exp(-u)}{2^{(n/2)-1}c^{(n/2)}\Gamma(n/2)} \frac{\sqrt{2c}}{2\sqrt{u}}du \tag{4.21}$$

Simplifying:

$$F_{||E||}(z) = \frac{1}{\Gamma(n/2)} \int_0^{z^2/(2c)} u^{(n/2)-1}\exp(-u)du \tag{4.22}$$

Recognize the lower incomplete gamma function for the integral:

$$\gamma(s,x) = \int_0^x t^{s-1}\exp(-t)dt \tag{4.23}$$

Thus, the CDF $F_{||E||}(z)$ is:

$$F_{||E||}(z) = \frac{\gamma(\frac{n}{2}, \frac{z^2}{2c})}{\Gamma(\frac{n}{2})} \tag{4.24}$$

where $n = 3m$ and $c = 2\xi^2$ can be plugged in for the exact expression.

With the CDF for the norm of the noise vector derived, we can know the probability of the noise vector's norm being in a certain range. We can use this information to assess the "quality of the Jacobian" by comparing the change in the magnetic field vector $||\delta B||$, which is related to σ_{min}, with the noise vector's norm $||E||$.

Use 4.9 as an example, the change of the magnetic field caused by δx is at least $0.1\sigma_{min}$. The sensor's noise level ξ corresponds to a certain CDF for the noise norm. From the CDF, we can obtain the probability of $||E||$ being in a certain range. If with probability 0.998, $||E|| < 0.1\sigma_{min}$, the change of magnetic field, we can conclude that $||\delta B||$ caused by the differential motion δx is larger than 99.8 percentile of $||E||$.

Criteria 1

In this section, we understand how the motion in the direction of v_{min} can be corrupted by noise and hypothesize that if σ_{min} is larger, the algorithm will be more robust to noise. Ideally, σ_{min} can be as large as possible for the entire workspace, which can be represented by the first criteria:

Criteria 1 - $mean_{x \in \mathcal{X}} \sigma_{min}(x)$

We would like to maximize this criterion to make the algorithm more robust to noise.

4.1.3 Condition number κ

The condition number of a matrix J is defined as:

$$\kappa(J) = \frac{\sigma_{max}}{\sigma_{min}} \tag{4.25}$$

where σ_{max} is the largest singular value of the matrix J and σ_{min} is the minimum non-zero singular value of J. As a result:

$$\kappa(J) > 1, \ \frac{1}{\kappa(J)} \in (0, 1) \tag{4.26}$$

There are various values for analyzing the reciprocal: 1. It measures a matrix's relative distance to a matrix with a lower rank. 2. It gives an idea of the accuracy of solving the problem $x = J^{\dagger} b$, where b is given and x is to be solved.

Distance to lower-rank matrix and observability

Assuming the matrix J is rank n, Eckart–Young–Mirsky theorem [19] shows:

$$\min_{X, \ rank(X)=k} ||J - X||_2 = \sigma_{k+1} \tag{4.27}$$

where $||\cdot||_2$ is the matrix spectral norm induced by the vector's l_2-norm. When $rank(J) = k + 1$, σ_{k+1} is the minimum non-zero singular vector that measures the absolute distance of the matrix J from losing rank. More often, we are interested in the relative distance from losing rank of a matrix J:

$$\min_{X, \ rank(X)=k} \frac{||J - X||_2}{||J||_2} = \frac{\sigma_{k+1}}{\sigma_1} = \frac{1}{\kappa} \tag{4.28}$$

where we use the fact that $||J||_2 = \sigma_1$, the maximum singular value. A small relative distance to a lower-rank matrix means even a perturbation, e.g. noise, that is small compared to the matrix would cause the matrix to lose rank.

In our situation, $rank(J) = 5$ because the magnet has 5 DOFs that the sensors can observe. For a given sensor configuration, if the Jacobian for a particular magnet configuration, (p, R), has a rank less than 5, i.e., $rank(J(p, R)) < 5$, this indicates a loss of more observable degrees of freedom at (p, R). Ideally, the Jacobian at all the magnet configurations we linearize the model is rank 5 and far from losing rank to ensure local observability.

Condition on pseudoinverse

As introduced in 3, every step of the nonlinear least square algorithm requires taking the pseudoinverse to solve the linearized least square problem:

$$\begin{bmatrix} \Delta p \\ \Delta \omega \end{bmatrix} = \Delta x = J^{\dagger}(p_k, R_k) r \tag{4.29}$$

where r is the residual for the current guess, (p_k, R_k), and J is the Jacobian of the model linearized at the current guess.

4.29 represents the ideal case where J and r can be evaluated exactly. However, they can be $J + \delta J$ and $r + \delta r$. δJ and δr are the error when J and r are evaluated. There are various sources where the error may come from, such as noise when we are calculating r, numerical rounding issues when we are evaluating J, etc. As a result, there will be an error, δx in calculating the update step:

$$(\Delta x + \delta x) = (J + \delta J)^{\dagger}(r + \delta r) \tag{4.30}$$

Wedin, 1973 [20], provides a upper bound on δx using the condition number κ under some assumptions:

1. A and $A + \delta A$ have equal rank
2. $||\delta A||_2 < \varepsilon ||A||_2$
3. $\varepsilon \kappa < 1$
4. $\Delta x = J^{\dagger} r$ and $\Delta x + \delta x = (J + \delta J)^{\dagger}(r + \delta r)$

$$\Rightarrow \frac{||\delta x||_2}{||\Delta x||_2} \leq \frac{\kappa}{1 - \varepsilon\kappa}\left(\varepsilon + \frac{||\delta r||_2}{||J||_2||\Delta x||_2} + \frac{\varepsilon\kappa||r_0||_2}{||J||_2||\Delta x||_2}\right) + \frac{\varepsilon||(JJ^T)^{\dagger} b||_2||J||_2}{||\Delta x||_2} \tag{4.31}$$

where $r_0 = r - J\Delta x$.

From 4.31, we can see that if κ is smaller, the error in calculating Δx, δx is smaller. This leads to our third hypothesis:

Hypothesis 3 - If κ is smaller in the workspace, the algorithm will converge smoother and faster.

Ideally, $\kappa = 1$, which is the smallest possible value, such that the error is most strictly bounded.

Criteria 2

In this section, we understand how the condition number represents the relative distance for the matrix to lose rank, which, in our case, losing observable DOFs. The smaller the condition number is, the further the matrix is to rank loss. It also gives an upper bound on the error when we calculate the pseudoinverse, the smaller the condition number is, the smaller the error is. Ideally, we would like to minimize κ. To align with our first criteria, σ_{min}, which needs to be maximized, we can maximize the reciprocal condition number $\frac{1}{\kappa}$ and it's the second criterion to optimize for:

Criteria 2 - $\min_{x \in \mathcal{X}} \frac{1}{\kappa}(x)$

Maximizing the minimum $\frac{1}{\kappa}$ in the workspace allows us to maximize this criterion for the entire workspace.

4.2 Sampling the workspace

With the criteria for optimization defined, we need to sample the spherical magnet workspace described in 1.1 into distinct and representative configurations. The positions selected are the ones on the surface of the sphere, as shown in Figure 4.1.

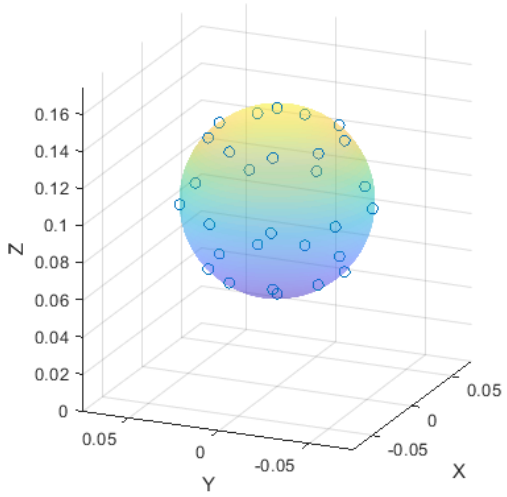

Figure 4.1: Position selected on the sphere's periphery

There are in total 29 positions are selected on the sphere evenly. At each position, orientations are also sampled. By inspecting the Jacobian in simulation, we observed the fact that the SVD of the Jacobians for a magnet at the same position but opposite orientation are the same:

$$SVD(J(p^*, R^*) = SVD(J(p^*, -R^*)) \tag{4.32}$$

As a result, the singular values that affect the criteria are the same for the two configurations, which do not need to be optimized repeatedly. Eventually, 17 orientations are sampled, facing upward, as shown in Figure 4.2.

In total, 493 configurations of the magnet are taken into account when evaluating the

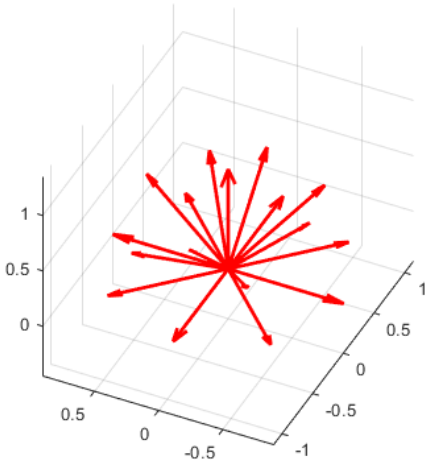

Figure 4.2: Orientation pointing toward the upper hemisphere

quality of a sensor configuration. The two criteria introduced in Section 4.1.2 and 4.1.3 are evaluated at these configurations. We aim to find a sensor configuration that maximizes the two criteria.

4.3 Baseline configuration

First, we tried to find a baseline sensor configuration, which gives us baselines for the criteria we are optimizing, i.e. $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$. A naive way for a sensor placement would be placing them on a grid. The initial guess for the "best possible sensing for a certain sensor workspace" can be achieved if we put as many sensors as possible on the workspace.

Following this mindset, we start with the minimum number that can form a grid, i.e. 4 sensors forming a 2-by-2 grid. Vary the size of the grid to find the peaks for the $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$. Ideally, the peaks are as close to each other as possible, so we can pick the size that optimizes both criteria. After picking the size, we increase the number of sensors inside the grid. The hypothesis is that two criteria will converge to certain levels as we increase the number of sensors because we are approaching the best sensing that can be achieved within the limited workspace. If this is true, we can prescribe certain thresholds for the two criteria according to our needs and use them as the baseline criteria.

4.3.1 Vary the size of a 2-by-2 grid

Starting with the minimum number of sensors that can form a grid, i.e. 4 sensors for a 2-by-2 grid, vary the size of the grid and find the peaks of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$. Locate the size of the grid where the peaks of the criteria are reached. Ideally, they happen at the same size so we obtain a size that is good for both the criteria we care about. But the simulation tells us the peak of $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$ is reached at a very big size of the grid, while the peak of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is reached at a small size of the grid. The peaks are far away from each other. Further investigation is required for this phenomenon.

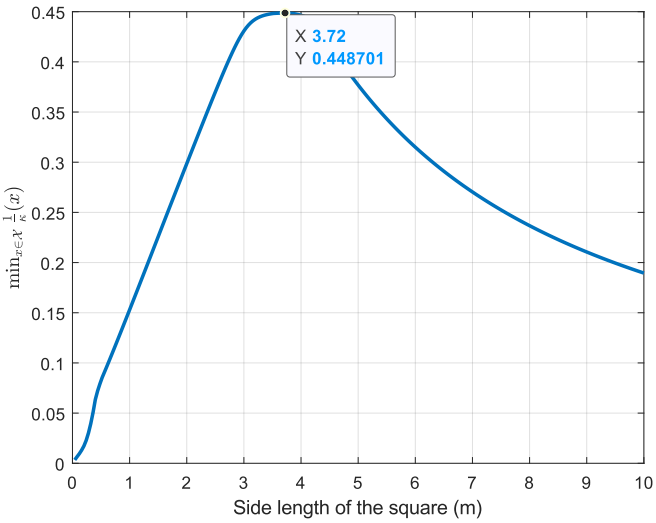

Figure 4.3: peak of $\frac{1}{\kappa}$

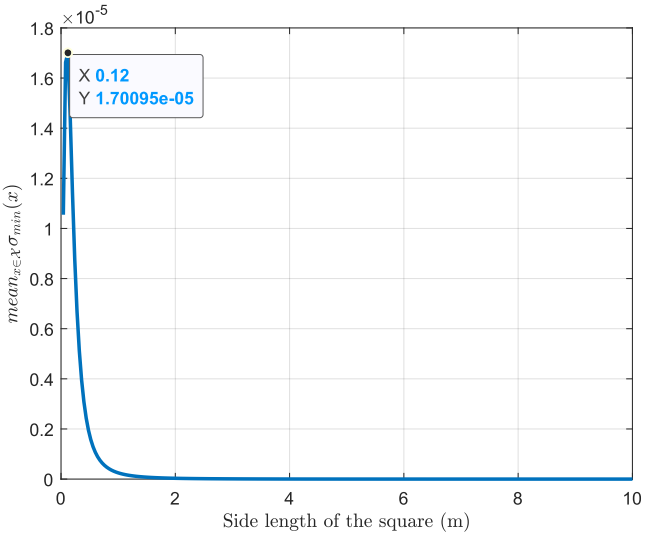

Figure 4.4: peak of σ_{min}

Observation and solution of the distinct peaks

We observe from Figure 4.3 and 4.4 that the peaks for the two criteria are far away from each other: the peak of $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ is reached at 3.72m while the peak of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is reached at 0.16m. By inspecting the Jacobian, we observed that the scale of the columns associated with the position is much bigger than the columns associated with the orientation. To solve this problem, we represent the distance in decimeters instead of meters. It will not change the peak for $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ but will bring the peak of $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ to a smaller size. Though we could not bring two peaks to the same size, we managed to bring them much closer by using a different unit for the distance: 0.52m for $\frac{1}{\kappa}$ and 0.16m for $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, as shown in Figure 4.5 and Figure 4.6. As a result, at the size for the peak of σ_{min}, the value of $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ is much higher than using m as the unit of distance. Other units have been tried but they cannot bring these two peaks closer

without largely sacrificing the scale of both criteria. dm is used as the unit for the distance from now on. For the next steps of looking for the baseline, the size is chosen to be the average value of the sizes where the two peaks are reached, i.e. $(0.52 + 0.16)/2 = 0.34m$.

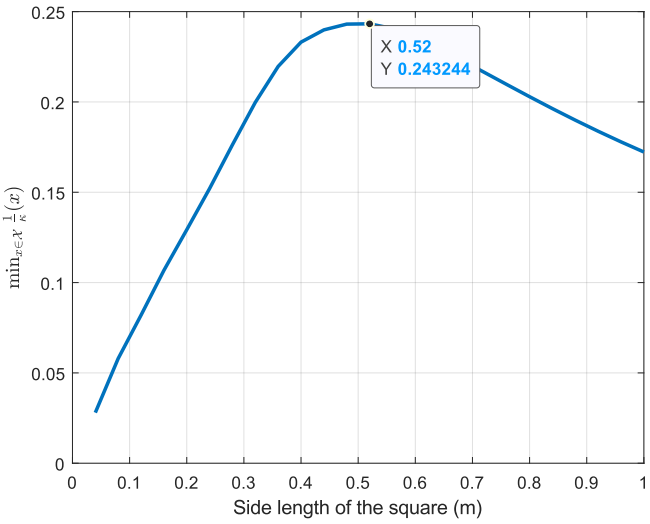

Figure 4.5: peak of $\frac{1}{\kappa}$ using dm

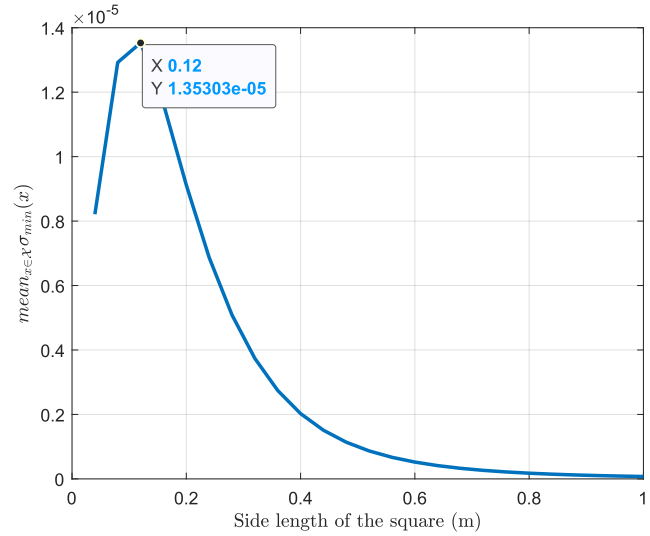

Figure 4.6: peak of σ_{min} using dm

4.3.2 Increase the number of sensors on the grid with a fixed size

Fix the size of the grid to 0.34m, increase the number of sensors to 100, i.e. a 10-by-10 grid, and check how the two criteria change. The hypothesis is that increasing the density of sensors will increase the sensing capability, the criteria will improve and converge to certain levels representing the best sensing capability. However, the simulation does not show the expected trends. $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ increases monotonically and $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ fluctuates as the density increases. Further investigation is needed for these phenomena.

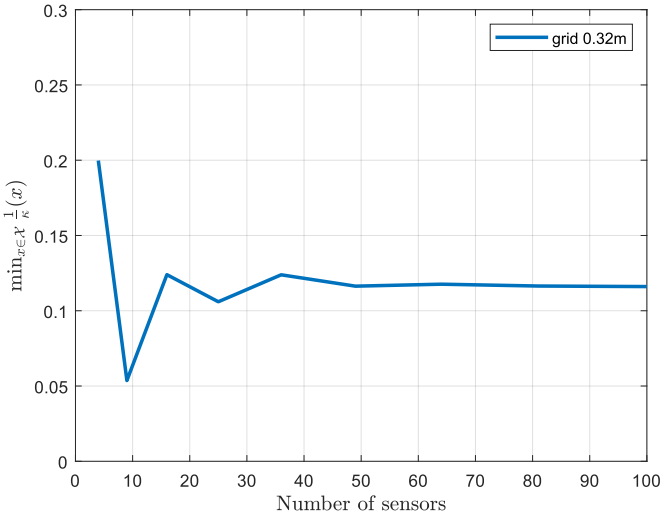

Figure 4.7: $\frac{1}{\kappa}$ as density increases

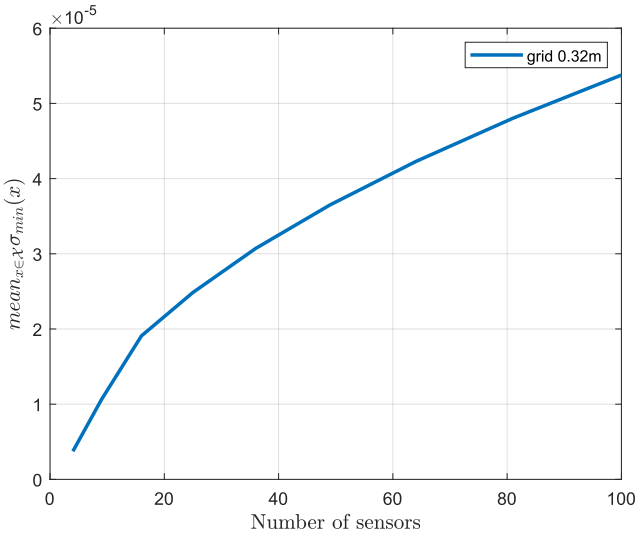

Figure 4.8: σ_{min} as density increases

Observation of the fluctuating $\frac{1}{\kappa}$

For $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$, there is a clear drop from number 4 to number 9, 16 to 25, 36 to 49, and so on, which then fluctuates and becomes stable. The hypothesis for the drops is adding sensors inside the grid will make $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ worse. Adding sensors from 4 to 9, 16 to 25, and so on, all put a sensor in the middle of the sensor workspace. We further tested this hypothesis by just adding 1 sensor inside the 4 sensor grid to make it a 5-sensor grid and observed the same phenomenon, as shown in Figure 4.9.

The reason for this phenomenon is that $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ is the ratio between the maximum singular value σ_{max} and the minimum one σ_{min}, while σ_{min} is mostly determined by the sensor that's furthest from the magnet, the maximum singular value is mostly determined by the sensor that's closest to the magnet. For $\frac{1}{\kappa}$ to be good, we need to have the sensors placed equidistant away from the magnet. In this case, σ_{max} and σ_{min} are more on the

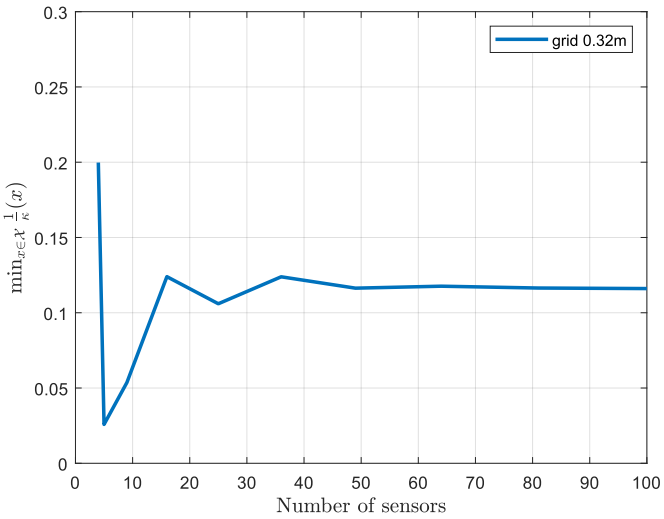

Figure 4.9: $\frac{1}{\kappa}$ as density increases with a five sensor grid

same scale. The balance breaks if one of the sensors is much closer to the magnet than the other. The maximum singular value increases because the middle sensor dominates the reading as it's much closer to the magnet. While σ_{min} is almost unchanged because it depends on the sensors that are further away. It causes a large drop in $\frac{1}{\kappa}$ when adding a sensor inside a grid. To verify this reasoning in simulation, we tried to only add sensors on the periphery of the square instead of adding sensors inside of the square to make it a grid; we also considered the case where we only added sensors on the periphery of the circle:

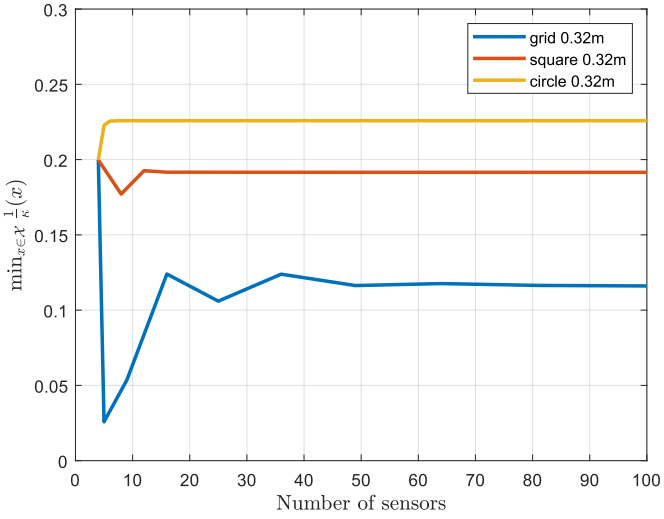

Figure 4.10: σ_{min} as density increases

Even though adding sensors on the periphery of the square, we can still observe a drop of $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ from 4 sensors to 9 sensors. The reason is the same as described above: some sensors on the square are closer to the magnet workspace compared to the others. We also observed that $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ quickly saturates as the number of sensors increases. Overall,

we can conclude that putting the sensors on the circle is beneficial for $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$.

Observation of monotonically increasing σ_{min}

For $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ shown in Figure 4.8, increasing the number of sensors monotonically increases $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, without a limit that it asymptotically converges to. This phenomenon can be proved under a certain assumption that the new sensors are added while the previous sensors' positions remain unchanged.

Let J_1 and J_2 be the Jacobians associated with two different sensors and assume they are full column rank, i.e. $N(J_1) = N(J_2) =$. Adding sensor 2 stack J_2 with J_1 and obtain a new Jacobian $J = [J_1, J_2]^T$. We want to prove that $\sigma_{min}(J) > \sigma_{min}(J_1)$:

$$\begin{aligned}\sigma_{min}(J) &= \min_{||x||_2=1} ||Jx||_2 = \min_{||x||_2=1} || \begin{bmatrix} J_1 \\ J_2 \end{bmatrix} x||_2 = \min_{||x||_2=1} || \begin{bmatrix} J_1x \\ J_2x \end{bmatrix} ||_2 \\ &\geq \min_{||x||_2=1} || \begin{bmatrix} J_1x \\ 0 \end{bmatrix} ||_2 = \min_{||x||_2=1} || \begin{bmatrix} J_1x \end{bmatrix} ||_2 = \sigma_{min}(J_1)\end{aligned} \tag{4.33}$$

The greater or equal, $\geq$, becomes strictly greater, $>$, when J_2 is full column rank such that $J_2x > 0$. This proof is also true for the case where J_1, J_2, and J have the same null space. So it is true in our case of the rotation Jacobian, where the null space is always the body's z-axis rotation. There is one difference in the proof that the minimization operation has to search $||x||_2 = 1$ in the subspace that is orthogonal to the matrices' null space.

Intuitively, adding sensors would increase the dimension of the output vector, which will increase its norm. This increase can be reflected by the increase of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. As a result, to extract valuable information from $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, we need to take the sensor noise into account because adding sensors not only increases the dimension of the output vector but also increases the dimension of the noise vector, as described in 4.1.2.

We also compare the trend of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ when the number of sensors increases in different ways, as shown in 4.11. We can observe that adding sensors on the circle increases the $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ the slowest. Because the sensors added on the circle are further than the ones added in a grid or on the square. However, the discrepancy is not significant when the sensor number is low.

4.3.3 Summary

Adding sensors monotonically increases $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. But if the sensor is placed at a distance to the magnet that is very different from others, $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ will be largely jeopardized. Thus, for a fixed number of sensors, putting the sensors on a circle is more reasonable than putting the sensors on a grid. However, putting sensors on the circle will slightly decrease $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ because the distance to the magnet is further than putting sensors inside. We will use the circular configuration as the baseline and use a global optimization tool, i.e. the genetic algorithm toolbox in MATLAB, to optimize the sensor configuration for a possibly better $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$. Then the configuration given by the global optimizer will be compared to the circular baseline configuration.

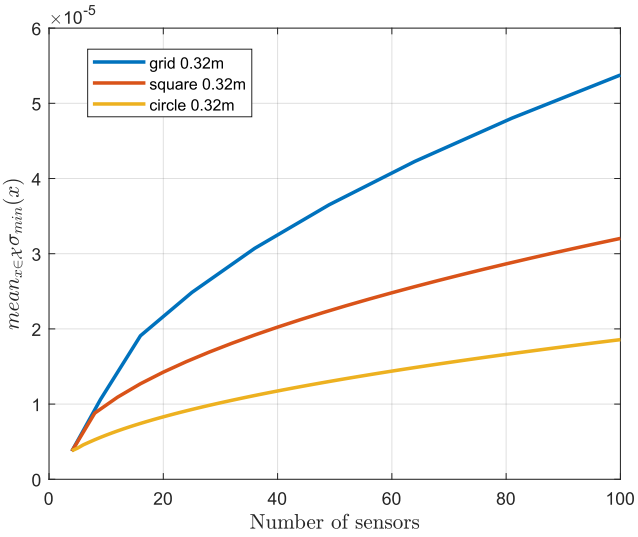

Figure 4.11: σ_{min} as density increases

4.4 Optimization with genetic algorithm

In Section 4.3, we decided to use the circular configuration as the baseline sensor configuration. In this chapter, we will use the global optimization toolbox provided by MATLAB to obtain the optimized configuration and compare that to the baseline circular configuration. Specifically, the genetic algorithm (GA) is used to optimize both of the criteria introduced in 4.2, $median_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$.

4.4.1 Multi-objective genetic algorithm and the Pareto front

A genetic algorithm (GA) solves both constrained and unconstrained optimization problems based on a natural selection process that mimics biological evolution. The algorithm repeatedly modifies a population of individual solutions. At each step, the genetic algorithm selects individuals from the current population following some selection criteria and uses them as parents to produce the children for the next generation. Over successive generations, the population "evolves" toward an optimal solution. It's more effective in finding the global minima of an optimization problem because of the randomness in generating a new population in every iteration.

It can be used for not only single-objective but also multi-objective optimization problems, which suits our problem considering two objectives, $\min_{x\in\mathcal{X}} \sigma_{min}(x)$ and $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$, as introduced in Sections 4.1.2 and 4.1.3.

For a multi-objective optimization problem, the solver tries to obtain the Pareto front for the objectives. The Pareto front is the set of all Pareto efficient solutions, which can be defined as follows. Consider a function $f : X \to \mathbb{R}^m$, where X is a compact set of feasible decisions in the space $\mathbb{R}^n$ and Y is the feasible criteria vectors in $\mathbb{R}^m$, such that $Y = \{y \in \mathbb{R}^m : y = f(x), x \in X\}$. Assuming that the preferred directions of criteria values are known, a criteria vector point $y'' \in \mathbb{R}^m$ strictly dominates (is preferred to) another point $y' \in \mathbb{R}^m$ is written as $y'' \succ y'$. In other words, $y'' \succ y'$ means each criterion in the vector y'' is not worse than the one in the vector y' and there is at least one criteria of y'' that is strictly better than the one of y'. The Pareto front is thus written as a set of

dominating points:

$$P(Y) = \{y'' \in Y; \{y' \in Y : y' \succ y'', y' \neq y''\} \in \emptyset\} \tag{4.34}$$

Figure 4.12 gives an example of the Pareto front, where the red points A to H are on the Pareto front because there are no other candidates that are better than the red ones for both objectives.

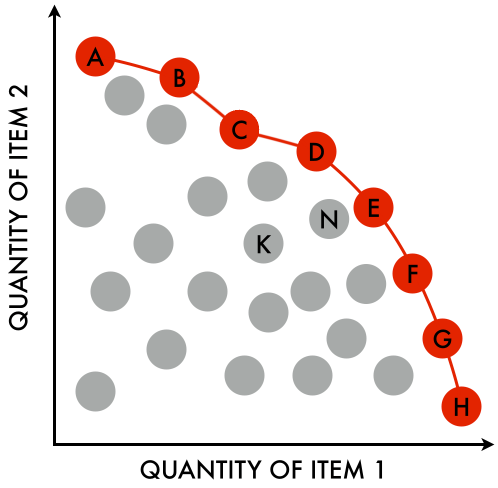

Figure 4.12: Example of the Pareto front

4.4.2 Optimization results

Using the `gamultiobj` provided by the Global Optimization toolbox in MATLAB, we can use the genetic algorithm to obtain the Pareto front of the two criteria $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$. The GA is run for 1000 iterations with a population size of 1000. The sensors can be placed within a 1m by 1m square for a more comprehensive search of the optimal solution and comparison. We perform optimization and comparison for different practically reasonable numbers of sensors, including 3, 4, 5, 9, and 16.

Figure 4.13 is the Pareto front obtained for the 4 sensors optimization. From the figure, we can observe that the two criteria are competing. On the upper left edge, we obtained the best of $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ but the worst of $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$ while on the lower right edge, it is the opposite.

We compare the results from GA with the sensor configurations where we put the sensors on the circle in Figure 4.14. The radius of the circle varies from 0.01m to 1m. The orange curve shows the trend for both criteria when the radius of the circle increases. We can observe that increasing the diameter of the circle will first increase both criteria (left part of the orange curve). $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ will first start to decrease while $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$ keeps increasing (middle part of the orange curve) until a point where both criteria start to decrease together (bottom part of the orange curve). It can be seen in Figure that the Pareto front obtained by GA is very similar to the result from putting the sensors on the circle, with some cases that the circular configuration is superior to the ones obtained from GA. We further checked the sensor configuration on different points of the Pareto front and confirmed that the solver is trying to put the sensors equidistant away from the magnet's workspace, which can be observed in Figure 4.15.

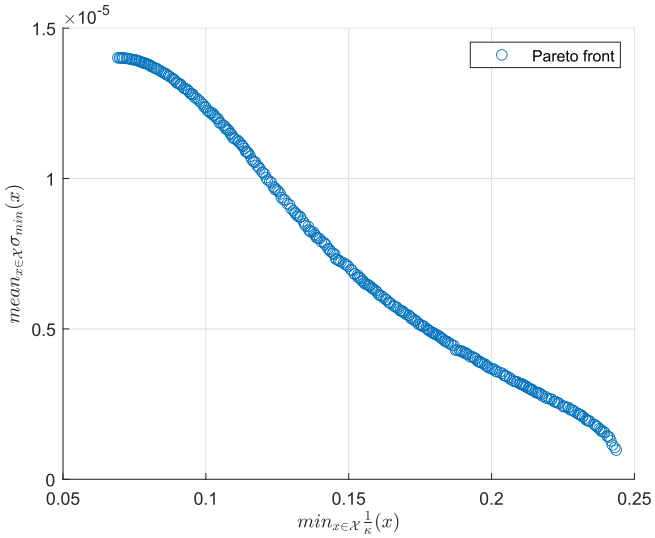

Figure 4.13: Pareto front of 4 sensors optimization

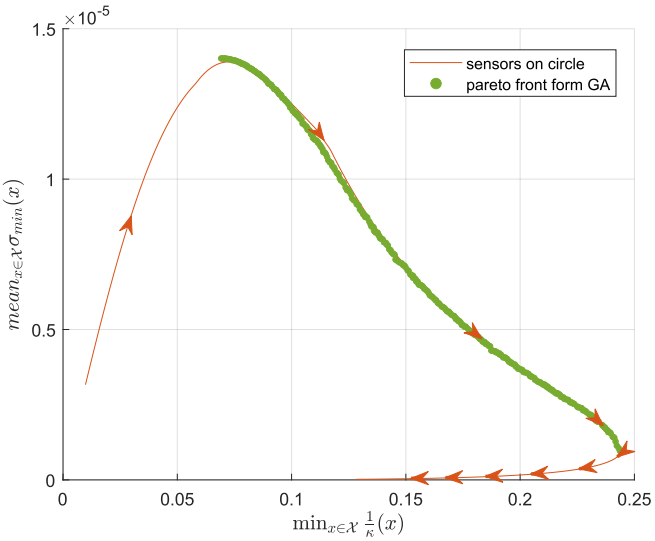

Figure 4.14: Pareto front comparison for 4 sensors

The same optimization and configuration checks are also done for the case of 3, 5, 9, and 16 sensors. We can also observe that the "Pareto front" obtained by the circular configuration coincides with the one obtained from the genetic algorithm. We conclude that the circular configuration is the optimal solutions to optimize the two criteria we are interested in, $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$.

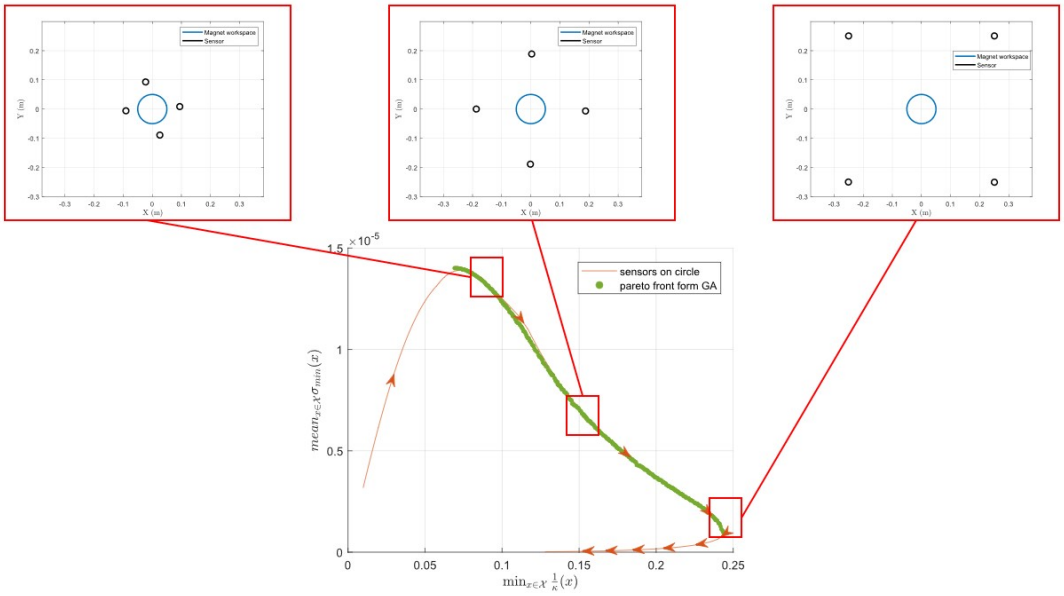

Figure 4.15: Pareto front for 4 sensors and examples of configurations

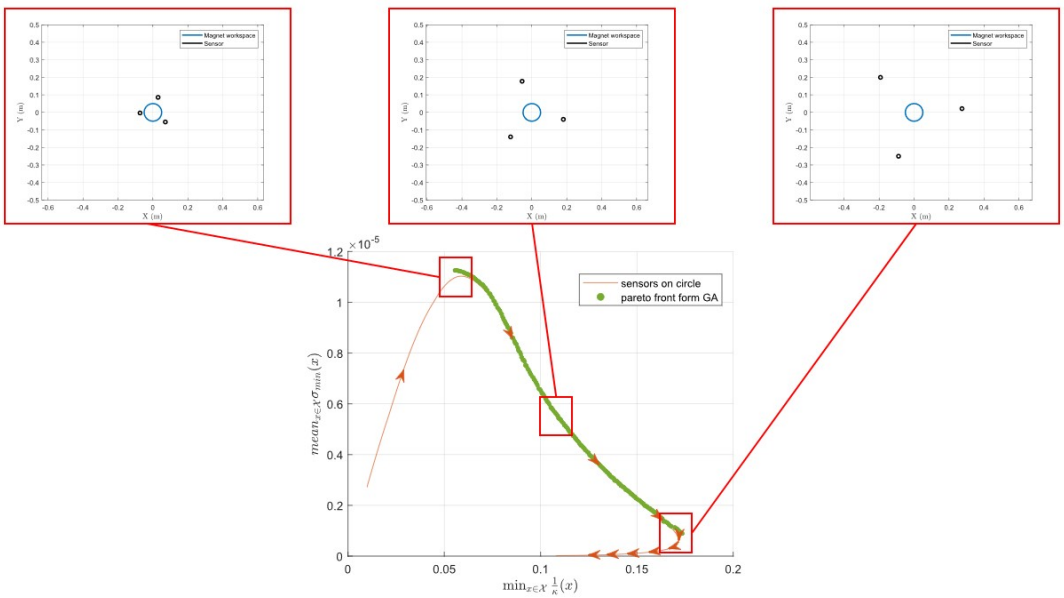

Figure 4.16: Pareto front for 3 sensors and examples of configurations

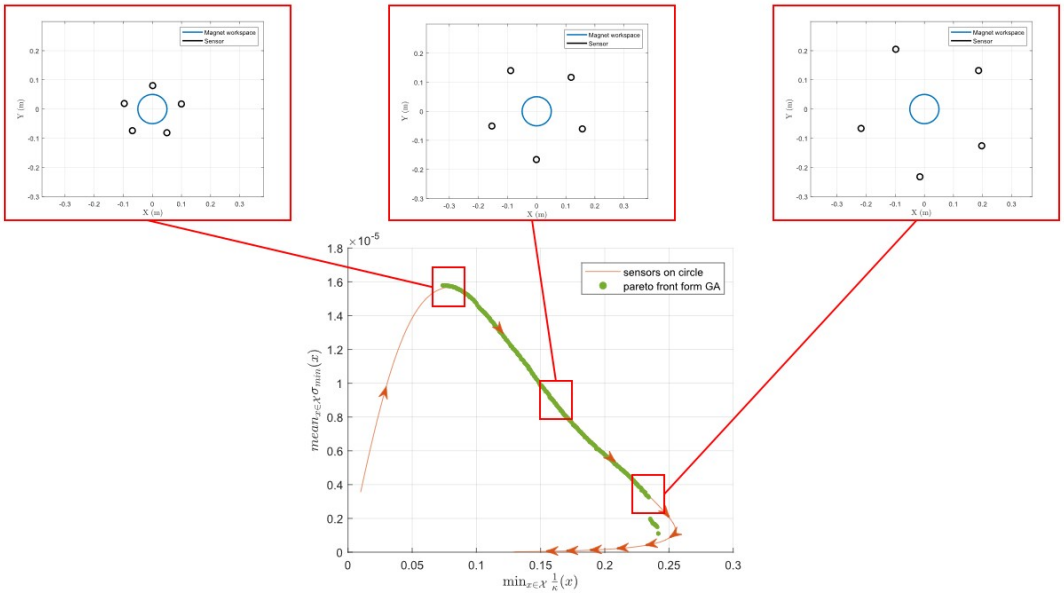

Figure 4.17: Pareto front for 5 sensors and examples of configurations

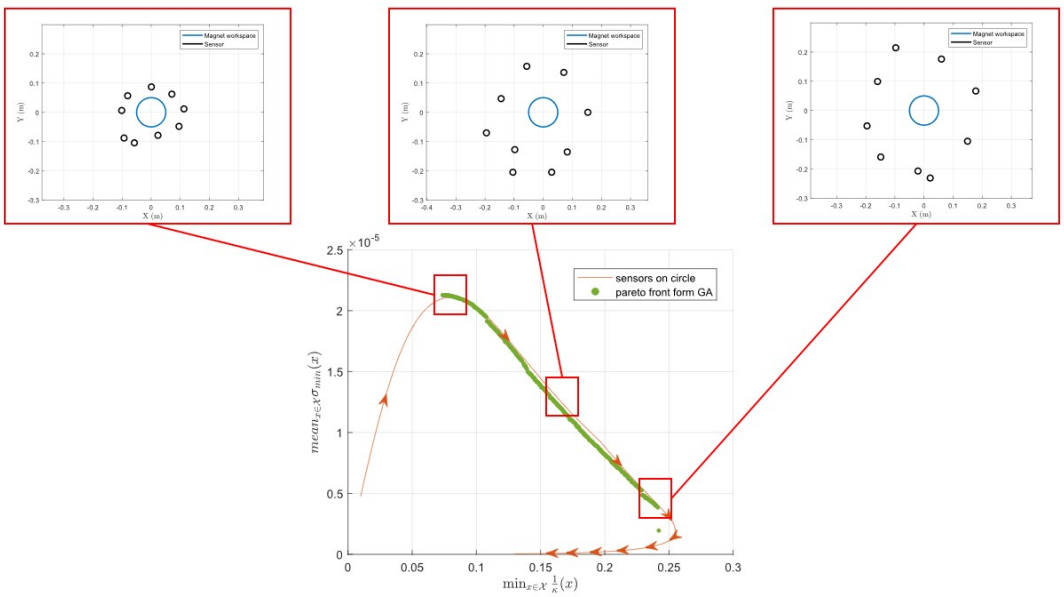

Figure 4.18: Pareto front for 9 sensors and examples of configurations

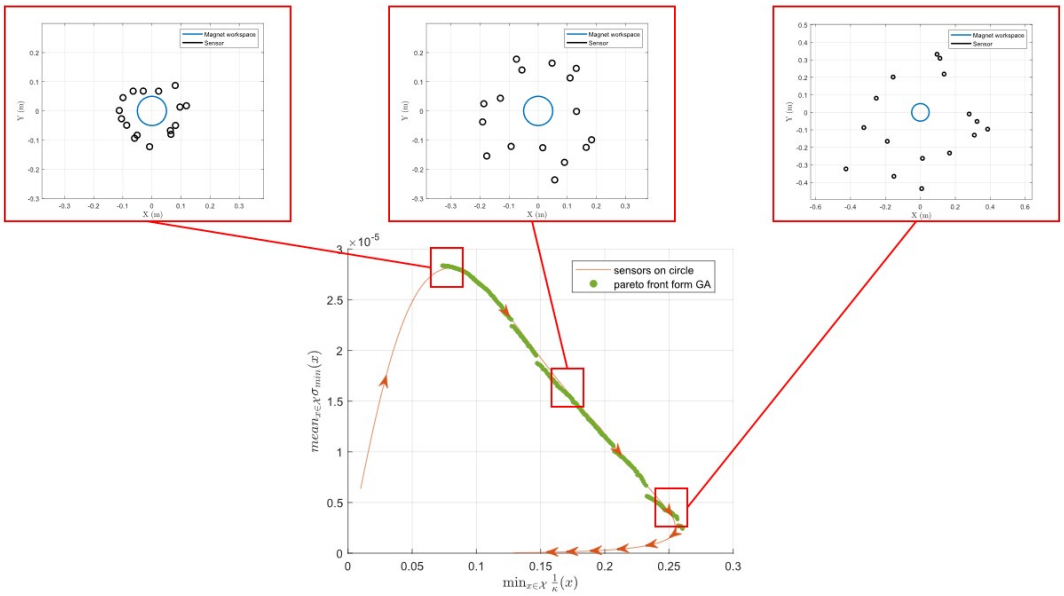

Figure 4.19: Pareto front for 9 sensors and examples of configurations

We now want to compare the results for different numbers of sensors. By putting sensors on a circular configuration and varying the diameter, we can obtain the trajectory of how the two criteria change according to the change of the diameter, which is similar to the orange curves in the figures above. On the curve, we can obtain the best performance for the two criteria, $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$. In Figures 4.20 and 4.21 we compare the best achivable $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ and for different numbers of sensors and their corresponding diameter. In Figures 4.22 and 4.23 we compare the best achievable $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$ and for different numbers of sensors and their corresponding diameter. We consider the number of sensors from 3 to 20.

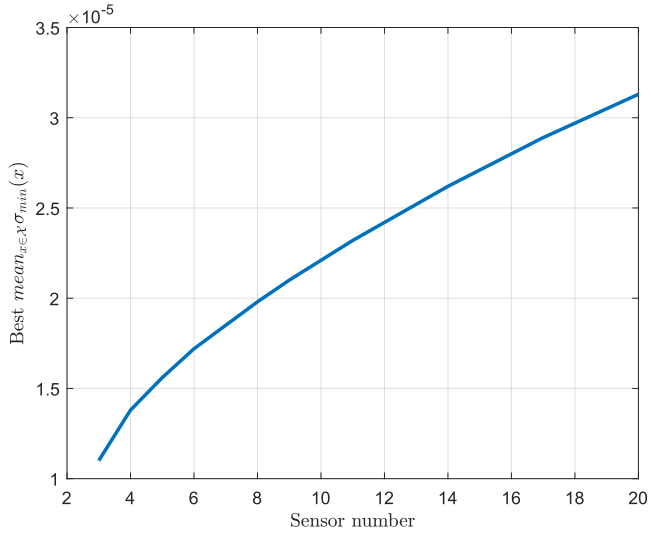

Figure 4.20: Best achievable $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ for different numbers of sensors

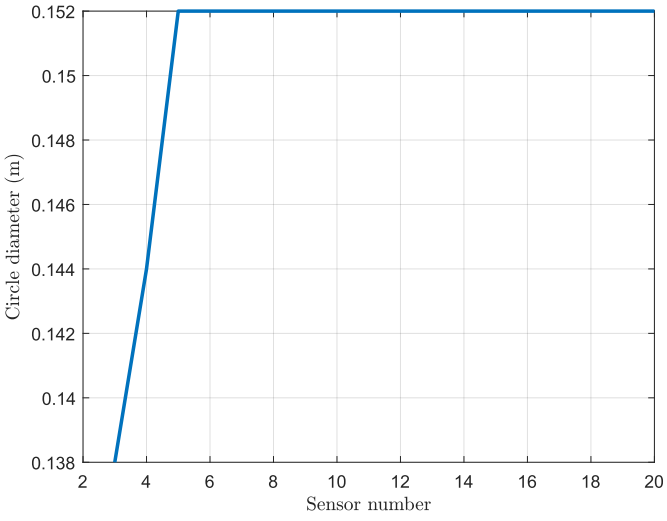

Figure 4.21: Circle diameters that achieve the best $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ for different numbers of sensors

From Figures 4.20 and 4.21, we observe that the best achievable $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ increases monotonically as the number of sensor increases. It aligns with the observation in Section 4.3. If the hypotheses introduced in Section 4.1.2 holds, as the number of sensors increases,

the average error for estimating the magnet's configuration in the workspace will decrease. The circle diameter that achieves the best $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ increases slightly from 0.152m to 0.186m and remains unchanged from number 6.

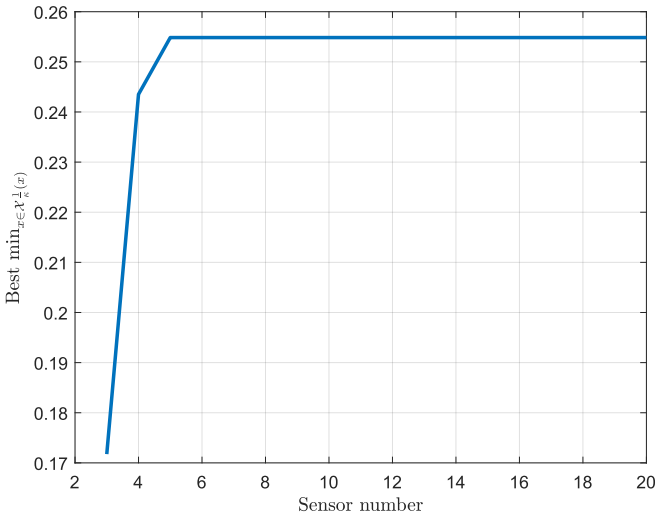

Figure 4.22: Best achievable $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$ for different numbers of sensors

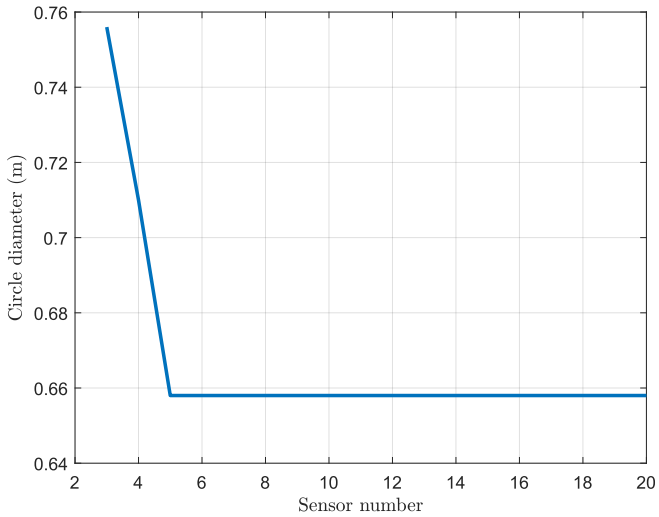

Figure 4.23: Circle diameters that achieve the best $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$ for different numbers of sensors

From Figures 4.22 and 4.23, we observed that the best achievable $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$ increases for the initial part and remains stable afterward, which aligns with the observation in Section 4.3. If the hypothesis introduced in Section 4.1.3 holds, the convergence speed for the algorithm will increase for the initial part and then remain stable. The circle diameter that achieves the best $\min_{x\in\mathcal{X})}\frac{1}{\kappa}(x)$ decreases slightly from 0.756m to 0.658m and remained unchanged from number 5.

We further compare the $||\delta B||$ generated by the small motion to the noise level described in Section 4.1.2. To make the comparison, we fix the sensor configuration to the one that

achieves the best $mean_{x \in \mathcal{X}} \sigma_{min}(x)$ and consider the extreme case, i.e. the configuration that has $min_{x \in \mathcal{X}} \sigma_{min}(x)$. As introduced in Section 4.1.2, the motion in the direction of v_{min} that corresponds to σ_{min} will be scaled down the most by the factor of σ_{min}. In other words, moving in the direction of v_{min} by the amount of 1 will generate a $||\delta_B||$ equal to σ_{min}. We will compare this value to the 99.8 percentile of the norm of noise, $||noise||$. The 99.8 percentile of $||noise||$ is a function of the sensor number, which is the n in Equation 4.24. The results are shown in Figures 4.24 and 4.25.

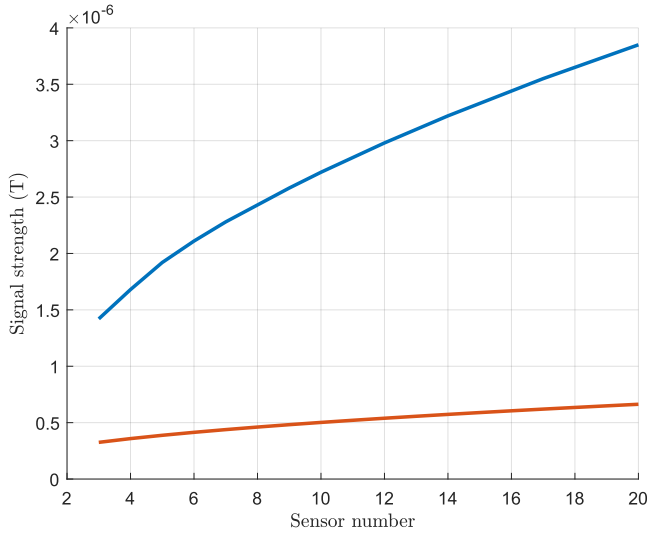

Figure 4.24: Signal and noise level for a movement in the direction of v_{min} by the amount of 1 for different numbers of sensors

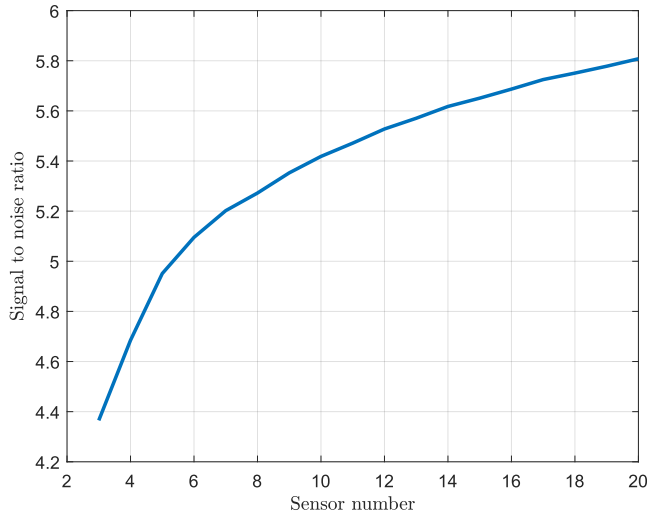

Figure 4.25: Signal-to-noise ratio for a movement in the direction of v_{min} by the amount of 1 for different numbers of sensors

From the figures, we can observe with an increasing number of sensors, the change of magnetic field, δB, generated by the motion that generates the least change in magnetic field, increases monotonically. It aligns with the monotonically increasing $mean_{x \in \mathcal{X}} \sigma_{min}(x)$. The level of noise norm, $||noise||$, also increases monotonically because its dimension is

increasing. However, the signal-to-noise ratio is increasing too, indicating that increasing the number of sensors is beneficial for distinguishing small motions. However, the figure also shows that the growth of the signal-to-noise ratio is slowing down.

Further analysis suggests that the magnet configuration that $min_{x \in \mathcal{X}} \sigma_{min}(x)$ is obtained is always $p = \begin{bmatrix} 0 & 0 & 0.2 \end{bmatrix}^T$ and $R = I$. It is because when sensor configuration is fixed, the further the magnet is from the sensors, the smaller signal can be detected by the sensors. The motion in the direction of v_5 at this configuration produces the smallest signal change among all the considered magnet configurations.

4.5 Summary

In this chapter, we defined the optimization criteria with the hypothesis that they will affect the accuracy of the algorithm, the error contribution from different DOFs, and the smoothness and speed of the algorithm. We observed that putting the sensors on the circle is beneficial compared to putting them on the grid, which is usually done in previous research. Using the genetic algorithm toolbox from MATLAB, we further confirmed the circular sensor configuration is the best for both criteria. In the next chapter, experiments will be conducted to compare different sensor configurations with different levels of the criteria to examine the validity of the hypotheses.

Chapter 5

Experiments

We conducted experiments to examine how the two criteria, σ_{min} and $\frac{1}{\kappa}$, affect the convergence quality of the algorithm. The experiment is first done in simulation, where noise-free and non-noise-free cases are considered. Then the magnet is put into the spherical workspace as described in 1.1, with the magnetic field captured by the sensor array of different configurations. Based on the recorded magnetic field, the algorithm is run to solve for the magnet's position and orientation and the results from different sensor configurations are compared. We examined the case of 4 sensors and the results are generalizable to other numbers of sensors.

5.1 Simulation experiments

We first carried out experiments in simulation. The test configuration is similar to the ones described in Section 3.4.1, where 210 positions are sampled and at each position, a cone of 25 different orientations is selected, resulting in 5250 test configurations in one test set. Within one test set, the cones at different positions face the same direction. To cover different magnet orientations, 16 cone directions are considered, resulting in 16 test sets.

In the simulation, the magnetic field generated by the magnet at position p and orientation R is modeled by the magnetic dipole model 2.4:

$$B(p,R) = \frac{B_r V}{4\pi|r|^3}(3\hat{r}\hat{r}^T - I)Re_z, r = p_i - p \tag{5.1}$$

The magnetic field is captured by 4 sensors located at $p_i,\ i \in \{1,2,3,4\}$, forming a magnetic field reading $B^* \in \mathbb{R}^{12}$. Algorithm 2 is used to solve for p and R from B^*. In the noise-free convergence test, we assumed the simulated sensor's reading, B^* is not corrupted by noise. In the case where there is noise, B^* is corrupted by different levels of Gaussian noise.

5.1.1 Convergence test with noise

We first test the hypotheses regarding the minimum singular value, σ_{min}. One is that for a configuration whose Jacobian in the linearized model has a higher σ_{min}, it's more robust to the effect of noise on the estimation of the configuration, i.e. when using the

algorithm to solve the inverse problem and in the case of convergence, the solution will end up closer to the ground truth than those with smaller σ_{min}. The other one is that the error in estimation mostly come from the direction that is the hardest to distinguish from noise, v_{min}.

To verify these hypothesis, we examined the first 4 sensor configurations with different $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, i.e. $r \in \{0.05, 0.1, 0.2, 0.3\}$m. The noise is added as i.i.d. Gaussian noise onto each axis of the measured magnetic field B^*. Different noise levels, represented by the variance of the Gaussian distribution σ^2, were tested: $0.05\mu T$, $0.1\mu T$, and $0.5\mu T$. The noise level of $0.05\mu T$ corresponds to the noise level recorded from the sensor used for the physical experiment, which will be described in 5.2. The configuration of $r = 1$m is not considered here because the sensors are too far away from the magnet, and the signal picked up by the sensor is much smaller than the noise level considered.

We sample the spherical workspace for test configurations. 210 positions were sampled. At each configuration, 32 orientations are chosen to cover all the orientations, as shown in Figure 5.1. As a result, there is one test set with $210 \times 32 = 6720$ test configurations.

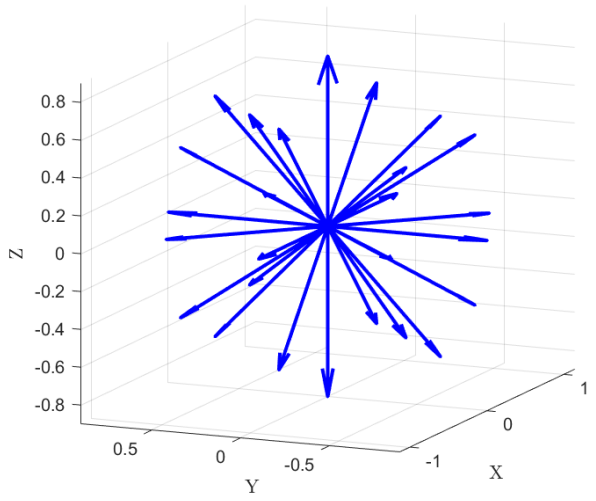

Figure 5.1: 32 orientations at each position for the simulated experiment

For the test set, $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is re-calculated to account for the fact that $\mathcal{X}$ has changed. It can be observed that $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ for each sensor configuration has the same relationship with each other, indicating that the sampled configuration used for optimization is representative. The simulation is run for 100 times to account for the randomness in the simulated noise, and the estimation errors for the position (e_p) and orientation (e_R) are recorded for each trial, which is then averaged and listed in Table 5.1, 5.2, and 5.3. Since the algorithm converges locally, to ensure convergence and only examine how the algorithm performs in estimation error, the initial guess is chosen to be the ground truth, which is close enough to the solution. We examined how the solution deviates from the ground truth.

From the tables, we can observe that in the configuration with a higher $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, the overall error is smaller than those with a lower $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. The configuration of $r = 0.1$m has the largest $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. And its error is the least. It verified our hypothesis: the magnitude of σ_{min} indicates the robustness against noise; the larger σ_{min}

noise: $0.05\mu T$	r=0.05m	r=0.1m	r=0.2m	r=0.3m
$mean_{x\in\mathcal{X}}\sigma_{min}(x)$	$1.03e-5$	$1.11e-5$	$4.07e-6$	$1.65e-6$
e_p (mm)	0.63	0.57	1.40	4.17
e_R (deg)	0.37	0.31	0.57	1.28

Table 5.1: Average error for different radii of circular configurations, $0.05\mu T$ noise

noise: $0.1\mu T$	r=0.05m	r=0.1m	r=0.2m	r=0.3m
$mean_{x\in\mathcal{X}}\sigma_{min}(x)$	$1.03e-5$	$1.11e-5$	$4.07e-6$	$1.65e-6$
e_p (mm)	1.27	1.14	2.79	8.63
e_R (deg)	0.74	0.62	1.14	2.55

Table 5.2: Average error for different radii of circular configurations, $0.1\mu T$ noise

noise: $0.5\mu T$	r=0.05m	r=0.1m	r=0.2m	r=0.3m
$mean_{x\in\mathcal{X}}\sigma_{min}(x)$	$1.03e-5$	$1.11e-5$	$4.07e-6$	$1.65e-6$
e_p (mm)	6.37	5.71	14.20	46.11
e_R (deg)	3.68	3.12	5.80	13.80

Table 5.3: Average error for different radii of circular configurations, $0.5\mu T$ noise

is, the more robust the measure is against noise, resulting in a better convergence quality when there is noise. In other words, the larger σ_{min} is, the same amount of motion in the direction of v_{min} will generate a bigger δB, which allows us to distinguish the small motions better.

To further verify the hypothesis that the motion in the direction of v_{min} is the hardest direction to resolve when there is noise, we project the estimation error $e \in \mathcal{R}^6$ onto the 5 singular vectors, respectively. It shows which direction of motion contributes the most to the error. The result is summarized in Table 5.4, 5.5, 5.6, and 5.7. The projection operation is represented by the inner product because $v_i,\ i \in \{1,2,3,4,min\}$ are all unit vectors. Only the noise level of $0.05\mu T$ is presented here, which is representative of verifying the hypotheses.

noise: $0.05\mu T$, $r = 0.05$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	75%	51%	17%	15%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	18%	14%	30%	50%	60%

Table 5.4: Projected error onto the singular vectors for the 5cm configuration

noise: $0.05\mu T$, $r = 0.10$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	88%	66%	34%	30%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	22%	19%	35%	48%	53%

Table 5.5: Projected error onto the singular vectors for the 10cm configuration

From the data, we can observe that most of the errors come from the direction of v_{min}, corresponding to the minimum singular value σ_{min}. It verifies the hypothesis that the

noise: $0.05\mu T$, $r = 0.20$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	87%	72%	55%	47%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	30%	35%	34%	43%	45%

Table 5.6: Projected error onto the singular vectors for the 20cm configuration

noise: $0.05\mu T$, $r = 0.30$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	89%	65%	52%	46%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	36%	38%	35%	38%	42%

Table 5.7: Projected error onto the singular vectors for the 30cm configuration

magnetic field change in the direction of the 5th singular value that corresponds to σ_{min} is the hardest direction to detect. When there is noise, this is the direction that is hardest to distinguish from noise. As a result, it contributes the most to the error of configuration estimation.

Another observation is that the error contributed from each singular vector is more balanced as the radius of the circle increases. It is due to the increased reciprocal condition number, $\frac{1}{\kappa}$. As we know, $\frac{1}{\kappa}$ is the ratio between σ_{min} and σ_{max}. As the radius increases, $\frac{1}{\kappa}$ increases, meaning the difference between σ_{min} and σ_{max} decreases. As a result, the difference in the difficulty in distinguishing each DOF is smaller. So, we can observe that the difference in the error contributed from each DOF becomes smaller.

Both of the hypotheses on σ_{min} holds, which gives us a criterion for choosing the sensor configuration: To ensure that the sensor configuration is more robust to noise and has a lower error in estimating the magnet configuration, one can select the ones resulting in a higher level of σ_{min} in the workspace, which can be represented by $mean_{x\in\mathcal{X}}\sigma_{min}(x)$.

5.1.2 Noise-free convergence test

Another hypothesis regarding the reciprocal condition number, $\min_{x\in\mathcal{X}}\frac{1}{\kappa}(x)$, is that: For the sensor number with a larger $\frac{1}{\kappa}(x)$ in the workspace, the algorithm will converge more smoothly and faster to the solution.

We selected 5 radii for the circular configurations according to the Pareto front, $r \in \{0.05, 0.1, 0.2, 0.3, 1\}$m, as shown in Figure 5.2.

The test configurations are similar to the ones described in Section 3.4.1, where 210 positions are selected and at each position, a cone of 25 orientations is chosen. The initial guess for the position is the center of the sphere and the initial guess for the orientation is the center vector of the cone. 16 cone directions are sampled to cover the different orientations, for all the cone directions, the initial guess for the orientation is always the center vector of the cone. In total, there are 16 test sets, and each test set with 1 initial guess for the orientation. Each test set has 5250 test configurations.

We first test if the size of the convergence cone is related to the two criteria we are optimizing, i.e. if either of the criteria we are optimizing is an indicator of how globally the algorithm can converge. In Section 3.4.1, the sensors are placed on a circle of 25cm

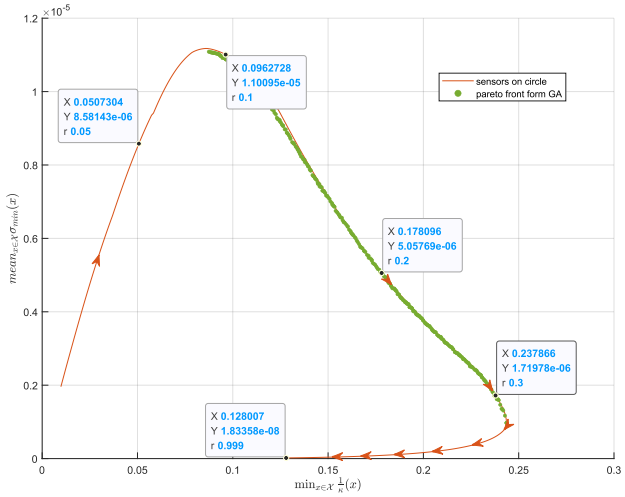

Figure 5.2: 5 radii selected for experiments and their corresponding criteria

radius, and within the cone of 60°, the algorithm converges 100%. For these 5 radii, the size 100%-convergence cone is recorded in Table 5.8

	r=0.05m	r=0.1m	r=0.2m	r=0.3m	r=1m
Cone size	5°	12.5°	55°	75°	120°

Table 5.8: 100% Convergence cone size for different radii of circular configurations

In terms of convergence speed, one hypothesis is that the higher the reciprocal condition number is in the workspace, the faster the convergence. We tested with the cone size of 12.5° for which 100%-convergence can be achieved for $r \in \{0.1, 0.2, 0.3, 1\}$m, the average convergence time for each test set of 5250 test configurations are recorded in Table 5.9.

	r=0.1m	r=0.2m	r=0.3m	r=1m
Convergence time (s)	1.72	1.49	1.37	1.22

Table 5.9: 100% Convergence time for different radii of circular configurations

We can observe that in the noise-free case, the convergence range increases monotonically as the radius of the circle increases, even though both the criteria already become very low in the case of $r = 1$m. For the same size of the convergence cone, the convergence speed also increases monotonically as the radius of the circle increases. It indicates that the reciprocal condition number in the workspace, $\frac{1}{\kappa}(x),\ x \in \mathcal{X}$, is not an indicator of the convergence speed. A possible reason is that after the guess is updated in the algorithm, the updated guess is out of the spherical workspace that has been optimized for. The reciprocal condition number can be very low in those configurations, indicating that the Jacobian is ill-conditioned. As a result, taking the pseudo-inverse of that Jacobian will result in a very big step in the later update, causing the algorithm to diverge. However, putting the sensor far away from the magnet's workspace makes the reciprocal condition number stable for a larger workspace. Even though one update step might result in a guess that is out of the optimized spherical workspace, the Jacobian is well-conditioned enough for the algorithm to converge back to the solution. Further investigation is required to understand this phenomenon.

5.1.3 Noise-free small motion detection test

All the analysis of the signal-to-noise ratio in Section 4.1.2, and Figures 4.24 and 4.25 are based on the linearized model. We further compared results from the linearized model and the results obtained from the nonlinear magnetic dipole model to confirm the validity of the linearized model's results.

In Figure 4.1.2, the orange curve represents $||\delta B||$ when the motion is taken in the direction of v_{min} for the amount of 1. We use 4 sensors as the example, putting them on the circle with radius 0.1m that maximizes $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ among the 4 cases considered above. We considered the motion taken in the direction of v_5 not only when the magnet is at the configuration with $min_{x\in\mathcal{X}}\sigma_{min}(x)$, which is always $p = \begin{bmatrix}0 & 0 & 0.2\end{bmatrix}^T$ and $R = I$ as described in Section 4.4.2 but also the v_{min} at other positions, such as $p = \begin{bmatrix}0 & 0 & 0.1\end{bmatrix}^T$, $p = \begin{bmatrix}0 & 0 & 0.15\end{bmatrix}^T$, $p = \begin{bmatrix}0.05 & 0 & 0.15\end{bmatrix}^T$, $p = \begin{bmatrix}0 & 0.05 & 0.15\end{bmatrix}^T$, $p = \begin{bmatrix}-0.05 & 0 & 0.15\end{bmatrix}^T$, and $p = \begin{bmatrix}0 & -0.05 & 0.15\end{bmatrix}^T$. In total, 7 positions are considered, and the orientations are chosen to be $R = I$.

In those magnet configurations, v_{min} that corresponds to σ_{min} are computed, representing the motion direction that generates the least $||\delta B||$. Different amounts of motion are taken in the direction of v_{min} and the magnetic field change suggested by the linear model and the nonlinear dipole model are computed and compared with each other.

The results are shown in Table. To better interpret the table, take $p = \begin{bmatrix}0 & 0 & 0.1\end{bmatrix}$ for example, its σ_{min} is $1.46e-5$ and $v_{min} = \begin{bmatrix}-0.45 & 0 & 0 & 0 & -0.89 & 0\end{bmatrix}^T$, $||v_{min}|| = 1$. It means that the linear model suggests that when the magnet is in the configuration of $p = \begin{bmatrix}0 & 0 & 0.1\end{bmatrix}$ and $R = I$, the motion that moves in the x-axis for -0.45dm and rotates around the y-axis for -0.89rad would generates the least magnetic field change $||\delta B|| = \sigma_{min} = 1.46e-5$. If the motion taken is $0.1v_{min}$, i.e. moves in the x-axis for -0.045dm and rotates around the y-axis for -0.089rad, the linear model suggests $||\delta_B|| = 1.46e-6$. This information is listed in the $||\delta B||_{linear}$ columns. The $||\delta B||_{dipole}$ columns show the magnetic field change suggested by the nonlinear magnetic dipole model, which is considered the "real" field change.

From the simulation results, we can observe $||\delta B||_{linear}$ always underestimates the real magnetic field change $||\delta B||_{dipole}$. The discrepancy between the linear and dipole models decreases dramatically when the motion taken decreases from v_{min} to $0.1v_{min}$, i.e. when the motion is small. This is because the linear model is more accurate when the deviation from the linearized point is smaller. This suggests the validity of the linear model and its value for estimating $||\delta B||$ generated by the small motion.

Another observation is that the discrepancy between the linear and nonlinear models is smaller when the linearized point is in the middle of the spherical workspace, i.e. $\begin{bmatrix}x & y\end{bmatrix}^T = \begin{bmatrix}0 & 0\end{bmatrix}^T$, which is also in the middle of the sensor's circular configuration. A possible reason is the symmetry. We can also observe the symmetry between the last 4 configuration in their v_{min} and $||\delta B||$.

p	σ_{min}	v_{min}	Motion	$\|\|\delta B\|\|_{linear}$	$\|\|\delta B\|\|_{dipole}$	Motion	$\|\|\delta B\|\|_{linear}$	$\|\|\delta B\|\|_{dipole}$
$\begin{bmatrix} 0 \\ 0 \\ 0.1 \end{bmatrix}$	$1.46e-5$	$\begin{bmatrix} -0.45 \\ 0 \\ 0 \\ 0 \\ -0.89 \\ 0 \end{bmatrix}$	v_{min}	$1.46e-5$	$1.89e-5$	$0.1v_{min}$	$1.46e-6$	$1.46e-6$
$\begin{bmatrix} 0 \\ 0 \\ 0.15 \end{bmatrix}$	$4.82e-6$	$\begin{bmatrix} -0.52 \\ 0 \\ 0 \\ 0 \\ -0.86 \\ 0 \end{bmatrix}$	v_{min}	$4.82e-6$	$6.72e-6$	$0.1v_{min}$	$4.82e-7$	$4.85e-7$
$\begin{bmatrix} 0 \\ 0 \\ 0.20 \end{bmatrix}$	$1.87e-6$	$\begin{bmatrix} -0.59 \\ 0 \\ 0 \\ 0 \\ -0.80 \\ 0 \end{bmatrix}$	v_{min}	$1.87e-6$	$3.00e-6$	$0.1v_{min}$	$1.87e-7$	$1.89e-7$
$\begin{bmatrix} 0.05 \\ 0 \\ 0.15 \end{bmatrix}$	$4.33e-6$	$\begin{bmatrix} 0 \\ -0.51 \\ 0 \\ 0.86 \\ 0 \\ 0 \end{bmatrix}$	v_{min}	$4.33e-6$	$7.06e-6$	$0.1v_{min}$	$4.33e-7$	$4.38e-7$
$\begin{bmatrix} 0 \\ 0.05 \\ 0.15 \end{bmatrix}$	$4.33e-6$	$\begin{bmatrix} -0.51 \\ 0 \\ 0 \\ 0 \\ -0.86 \\ 0 \end{bmatrix}$	v_{min}	$4.33e-6$	$7.06e-6$	$0.1v_{min}$	$4.33e-7$	$4.38e-7$
$\begin{bmatrix} -0.05 \\ 0 \\ 0.15 \end{bmatrix}$	$4.33e-6$	$\begin{bmatrix} 0 \\ 0.51 \\ 0 \\ -0.86 \\ 0 \\ 0 \end{bmatrix}$	v_{min}	$4.33e-6$	$7.06e-6$	$0.1v_{min}$	$4.33e-7$	$4.38e-7$
$\begin{bmatrix} 0 \\ -0.05 \\ 0.15 \end{bmatrix}$	$4.33e-6$	$\begin{bmatrix} 0.51 \\ 0 \\ 0 \\ 0 \\ 0.86 \\ 0 \end{bmatrix}$	v_{min}	$4.33e-6$	$7.06e-6$	$0.1v_{min}$	$4.33e-7$	$4.38e-7$

Table 5.10: Experiment results for the 5cm configuration

5.2 Physical experiments

Physical experiments are also conducted to further support the hypotheses on σ_{min}

5.2.1 Experiment setup and data collection

The experimental setup is shown in Figure 5.3. 4 sensors are placed on the planar sensor board. The magnet is put in a spherical workspace whose radius is 5cm and the bottom is 10cm away from the sensor board. The setup resembles the one described in 1. The figure shows a magnet located at $p = \begin{bmatrix} -0.05 & 0 & 0.15m \end{bmatrix}^T$ with an orientation $R = I$, which is at the surface of the spherical workspace and facing upward. The sensor configuration in the figure is $r = 0.05$m. In the experiment, both $r = 0.05$m and $r = 0.1$m are considered.

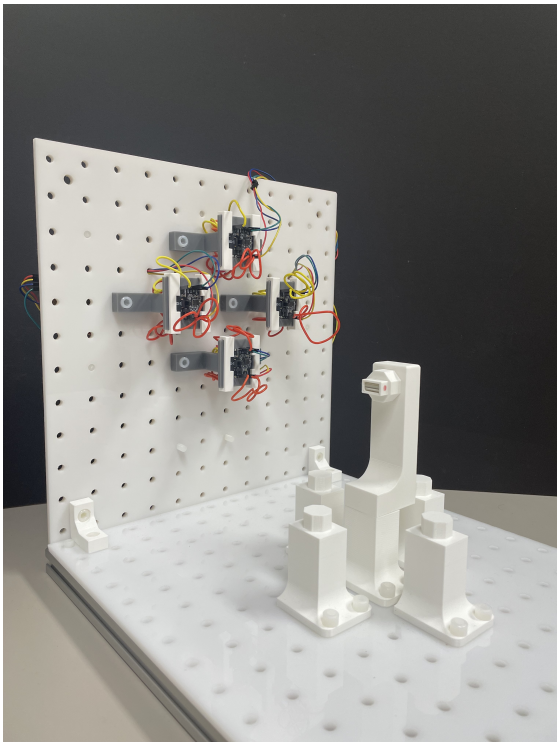

Figure 5.3: Experiment setup

The 4 sensors are the RM3100 magneto-inductive sensors from PNI. The sampling rate is set to 150Hz for one sample of 3-axis magnetic field. They are communicated with an Arduino Leonardo board via I^2C, whose reading can be obtained by MATLAB for data recording and analysis. An online running average filter is applied in MATLAB to reduce the sensor noise. Assuming the noise is Gaussian white noise, the average noise recorded is $0.05\mu T$.

In terms of the test configurations, 7 positions are sampled from the spherical workspace. At each position, 32 orientations are sampled, resulting in $7 \times 32 = 224$ test configurations. They are shown in Figure 5.4. For each magnet configuration, the generated magnetic field is recorded by the 4 sensors. 10 samples are averaged to obtain the magnetic field reading for 1 magnet configuration, reducing the effect of noise further.

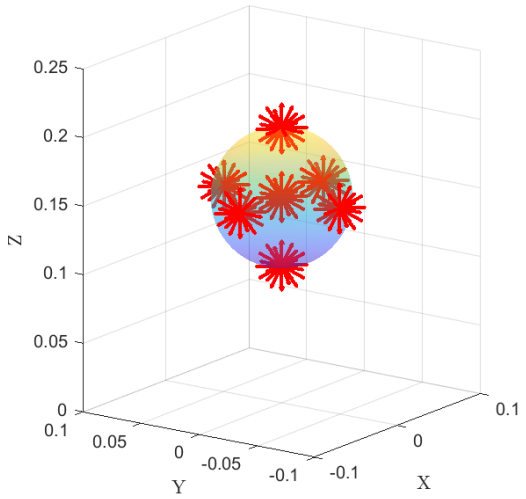

Figure 5.4: 224 experiment configurations, 32 orientations at 7 positions

5.2.2 Sensor calibration

Upon acquiring the data, sensor calibration is performed to account for the model mismatch in sensor configuration. The calibration is under the following assumptions:

- 3 axes of one sensor are at the same position.
- Sensor's orientations are assumed to be known as aligned with the world frame.
- The magnet configurations are assumed to be accurate.

The recorded magnetic field vector for the 224 configurations is $B^* \in \mathbb{R}^{2688}$, which corresponds to 224×12 measurements. The estimated magnetic field for sensors at p_s is $\hat{B}(p_s) \in \mathbb{R}^{2688}$, where $p_s \in \mathbb{R}^{3\times4}$ and each column represents the $\begin{bmatrix} x & y & z \end{bmatrix}^T$ coordinates of each sensor.

The calibration procedure is trying to account for the misalignment when building the system, where the sensor might not be perfectly located at the expected position. And it is trying to solve the following minimization problem:

$$min_{p_s}|B^* - \hat{B}(p_s)| \tag{5.2}$$

The calibrated sensor positions for the 5cm configuration are shown in 5.3 and 5.4. The unit is m.

$$p_1 = \begin{bmatrix} 0.0463 \\ -0.0012 \\ 0.0015 \end{bmatrix}, p_2 = \begin{bmatrix} -0.0039 \\ 0.0507 \\ 0.0038 \end{bmatrix}, p_3 = \begin{bmatrix} -0.0537 \\ -0.0015 \\ 0.0025 \end{bmatrix}, p_4 = \begin{bmatrix} -0.0040 \\ -0.0515 \\ 0.0011 \end{bmatrix} \tag{5.3}$$

And the calibrated sensor positions for the 1cm configuration are:

$$p_1 = \begin{bmatrix} 0.0972 \\ -0.0013 \\ 0.0002 \end{bmatrix}, p_2 = \begin{bmatrix} -0.0039 \\ 0.1015 \\ 0.0030 \end{bmatrix}, p_3 = \begin{bmatrix} -0.1054 \\ -0.0017 \\ 0.0029 \end{bmatrix}, p_4 = \begin{bmatrix} -0.0038 \\ -0.1017 \\ -0.0003 \end{bmatrix} \tag{5.4}$$

After the calibration, the mean error reduces from $0.61\mu T$ to $0.36\mu T$ for the 5cm configuration and from $0.34\mu T$ to $0.22\mu T$ for the 10cm configuration. The calibrated p_s is used for further calculation as the ground truth sensor position, e.g. in the algorithm that solves for the magnet's configuration.

5.2.3 Results

Similar to the non-noise-free simulated experiment, the recorded magnetic field is used as B^* to run Algorithm 2. To ensure convergence, the initial guess is chosen to be the ground truth configuration. The mean and max error after convergence are calculated. For both sensor configurations, the $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is also re-calculated to account for the change of magnet configuration. We also report the error and $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ for the 7 positions respectively. The results are summarized in Table 5.11 and 5.12.

	$mean(\sigma_{min})$	$mean(e_p)$(mm)	$max(e_p)$(mm)	$mean(e_R)$(deg)	$max(e_R)$(deg)
Overall	$1.25e-5$	3.77	14.98	2.62	15.79
$p = \begin{bmatrix} 0 \\ 0 \\ 0.1 \end{bmatrix}$	$4.43e-5$	1.90	4.03	1.39	3.19
$p = \begin{bmatrix} 0 \\ 0 \\ 0.15 \end{bmatrix}$	$9.44e-6$	2.91	8.35	2.06	6.34
$p = \begin{bmatrix} 0 \\ 0 \\ 0.20 \end{bmatrix}$	$2.78e-6$	5.61	14.98	3.23	9.57
$p = \begin{bmatrix} 0.05 \\ 0 \\ 0.15 \end{bmatrix}$	$7.49e-6$	4.32	11.07	2.90	12.15
$p = \begin{bmatrix} 0 \\ 0.05 \\ 0.15 \end{bmatrix}$	$7.60e-6$	4.49	10.80	3.45	13.16
$p = \begin{bmatrix} -0.05 \\ 0 \\ 0.15 \end{bmatrix}$	$7.95e-6$	3.50	14.79	2.66	15.79
$p = \begin{bmatrix} 0 \\ -0.05 \\ 0.15 \end{bmatrix}$	$7.61e-6$	3.70	7.60	2.69	6.59

Table 5.11: Experiment results for the 5cm configuration

From the data, we can observe that the 5cm configuration whose $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is smaller has a higher error than the 10cm configuration. Another observation is that within the same sensor configuration, the error on estimating the magnet configuration, whose σ_{min} is smaller, is also larger. For example, in the 5cm configuration, the error is the largest at position $p = \begin{bmatrix} 0 & 0 & 0.2 \end{bmatrix}^T$, whose $mean(\sigma_{min})$ is the least. the error is least for

	$mean(\sigma_{min})$	$mean(e_p)$(mm)	$max(e_p)$(mm)	$mean(e_R)$(deg)	$max(e_R)$(deg)
Overall	$1.18e-5$	2.74	13.46	1.68	12.66
$p=\begin{bmatrix}0\\0\\0.1\end{bmatrix}$	$3.09e-5$	1.27	2.59	1.33	2.78
$p=\begin{bmatrix}0\\0\\0.15\end{bmatrix}$	$1.05e-5$	2.01	5.31	1.47	6.57
$p=\begin{bmatrix}0\\0\\0.20\end{bmatrix}$	$3.79e-6$	4.22	13.46	2.57	12.66
$p=\begin{bmatrix}0.05\\0\\0.15\end{bmatrix}$	$9.00e-6$	2.66	7.05	1.55	6.39
$p=\begin{bmatrix}0\\0.05\\0.15\end{bmatrix}$	$9.35e-6$	3.44	11.16	1.72	10.27
$p=\begin{bmatrix}-0.05\\0\\0.15\end{bmatrix}$	$9.35e-6$	2.55	4.64	1.30	10.27
$p=\begin{bmatrix}0\\-0.05\\0.15\end{bmatrix}$	$9.31e-6$	3.00	6.63	1.81	4.46

Table 5.12: Experiment results for the 10cm configuration

$p=\begin{bmatrix}0 & 0 & 0.1\end{bmatrix}^T$, whose $mean(\sigma_{min})$ is the largest. The error for $p=\begin{bmatrix}0.05 & 0 & 0.15\end{bmatrix}^T$, $p=\begin{bmatrix}0 & 0.05 & 0.15\end{bmatrix}^T$, $p=\begin{bmatrix}-0.05 & 0 & 0.15\end{bmatrix}^T$, $p=\begin{bmatrix}0 & -0.05 & 0.15\end{bmatrix}^T$ are comparable and they also have comparable $mean(\sigma_{min})$. This is another evidence that σ_{min} of the magnet's configuration indicates how well we can estimate the configuration.

The projected error analysis is also done for experiment data, which is summarized in Table 5.13 and 5.14. We see consistent results for the 10cm configuration. For the 5cm configuration, more error is lying on v_4. The reason for this discrepancy can be that σ_4 is close to σ_5 and there is bias in the collected data. The bias in the data can be caused by non-perfect calibration of the sensors. When we calibrate the sensors, we assume the magnet configuration are accurate and the 3 axes of one sensor are at the same position and the z-axis is perfectly pointing upward. But in reality, there is also discrepancies in the magnet's configurations, and the 3 axes of the magnetic sensor are not located at the same position. These are also the reason that the error we observe in the physical experiment (Table 5.11 and 5.14) is much higher than that in simulation (Table 5.1 and 5.2).

$r=0.05$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	75%	61%	19%	14%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	22%	25%	21%	57%	52%

Table 5.13: Projected error onto the singular vectors for the 5cm configuration

$r = 0.10$m	$\frac{\sigma_1}{\sigma_1}$	$\frac{\sigma_2}{\sigma_1}$	$\frac{\sigma_3}{\sigma_1}$	$\frac{\sigma_4}{\sigma_1}$	$\frac{\sigma_5}{\sigma_1}$
Ratio	100%	72%	59%	38%	29%
Project of e	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$	$\frac{\lvert\langle e,v_1\rangle\rvert}{\lvert e\rvert}$
Ratio	24%	27%	21%	51%	54%

Table 5.14: Projected error onto the singular vectors for the 10cm configuration

5.3 Summary

Experiments in both simulation and physical are conducted to compared the results from the optimization.

In the simulation, both noise-free experiment and non-noise-free experiments are conducted. In the noise-free experiments, we observed that the larger the circular configuration is, the better the algorithm converges, in terms of speed and the convergence range, regardless of what $\min_{x\in\mathcal{X}} \frac{1}{\kappa}(x)$ is. A possible reason is given in 5.1.2. More analysis is required to understand better the range of convergence.

In the non-noise-free experiment, we observed that the larger $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ is, the better the algorithm converges the solution. It verifies the hypothesis that the configuration with larger σ_{min} is more robust to noise. By designing the sensor array configuration, we can achieve an overall larger σ_{min} in the entire workspace, which is represented by $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. As long as the initial guess is close enough to the solution, the error after convergence is smaller for the configuration that provides a larger $mean_{x\in\mathcal{X}}\sigma_{min}(x)$.

Results from the physical experiment further validate the hypothesis. The experiment results indicate that σ_{min} is a valuable criterion when designing a magnetic sensor configuration to detect the magnet's configuration. It represents how accurately the magnet's configuration can be solved. Further analysis is required to understand the convergence range and speed of the algorithm.

Chapter 6

Conclusion

In this thesis, we aimed to optimize the configuration of a planar sensor array for better tracking the magnet. The magnet can be mounted on the tip of the catheter operating in the human heart and by tracking it, we can reconstruct the shape of the catheter to close the control loop. To achieve this goal, we

- Derived the linearized magnetic dipole model properly using the local parametrization of the orientation. It considers that the parametrization for orientation in the Euclidean space is not isomorphic to the Special Orthogonal group $SO(3)$, where the orientation lies. Compared to the global parametrization, e.g. Euler angles, whose linearized model is not physically correct, the properly derived linearized model allows us to analyze the local behavior of the magnetic field when the magnet moves around a certain configuration. The validity of the linear model can be shown in Section 5.1.3.

- Developed the Gauss-Newton on manifold algorithm that tracks the magnet configuration from the measured magnetic field using the linearized magnetic field model. Compared to the algorithm that uses overparametrization for orientation, e.g. quaternion, the speed of Gauss-Newton on manifold algorithm for tracking the magnet is 100 times faster, as shown in Section 3.4. It is beneficial for real-time control of the instrument where the magnet is mounted.

- Derived the criterion that affects the convergence accuracy of the algorithm, i.e. $mean_{x\in\mathcal{X}}\sigma_{min}(x)$. Compared to the grid configuration, putting the sensors on a circle is beneficial in maximizing this criterion, resulting in higher accuracy in estimating the magnet's configuration. This can be observed from the edge of the Pareto front that maximizes $mean_{x\in\mathcal{X}}\sigma_{min}(x)$, whose corresponding sensor configuration is circular, as shown in Section 4.4.2. To select a sensor configuration in practice, we can choose the one that has higher $mean_{x\in\mathcal{X}}\sigma_{min}(x)$ to achieve better overall estimation accuracy, based on the experimental results shown in Sections 5.1.1 and 5.2.3.

- By analyzing the linear magnetic field model, we understood where the estimation error comes from, i.e. the direction of motion that generates the least change in the magnetic field. It can be verified by the experimental results in Sections 5.1.1 and 5.2.3.

In future work, to obtain a more representative result in the physical experiment, the

calibration procedure can be improved:

- The magnet configurations should also be considered unknown.
- The axes of the magnetic sensor should be considered independent. It also requires adapting the Gauss-Newton on manifold algorithm because the original algorithm assumes the magnetic sensors' axes have the same origin.

What remains unknown is the criterion that would affect how globally and how fast the algorithm works. Our results indicate that the algorithm works more globally and quicker as the size of the circle where the sensors are placed grows. The algorithm's performance can also be improved by some heuristic methods, such as adaptive step size like the backtracking line search [21], multiple initial guesses that explore different regions of the parameter space [22], etc. More understanding and adaptation of the algorithm remain as potential future work.

Bibliography

[1] M. Shurrab, A. Danon, I. Lashevsky, A. Kiss, D. Newman, T. Szili-Torok, and E. Crystal, "Robotically assisted ablation of atrial fibrillation: A systematic review and meta-analysis," *International Journal of Cardiology*, vol. 169, no. 3, pp. 157–165, 2013.

[2] A. J. Tatooles, P. S. Pappas, P. J. Gordon, and M. S. Slaughter, "Minimally invasive mitral valve repair using the da vinci robotic system," *The Annals of thoracic surgery*, vol. 77, no. 6, pp. 1978–1984, 2004.

[3] R. A. Rippel, A. E. Rolls, C. V. Riga, M. Hamady, N. J. Cheshire, and C. D. Bicknell, "The use of robotic endovascular catheters in the facilitation of transcatheter aortic valve implantation," *European Journal of Cardio-Thoracic Surgery*, vol. 45, no. 5, pp. 836–841, 12 2013.

[4] M. Ackerman, "The visible human project," *Proceedings of the IEEE*, vol. 86, no. 3, pp. 504–511, 1998.

[5] I. Floris, J. M. Adam, P. A. Calderón, and S. Sales, "Fiber optic shape sensors: A comprehensive review," *Optics and Lasers in Engineering*, vol. 139, p. 106508, 2021.

[6] H. Guo, F. Ju, Y. Cao, F. Qi, D. Bai, Y. Wang, and B. Chen, "Continuum robot shape estimation using permanent magnets and magnetic sensors," *Sensors and Actuators A: Physical*, vol. 285, pp. 519–530, 2019.

[7] C. Zhang, Y. Lu, S. Song, and M. Q.-H. Meng, "Shape tracking and navigation for continuum surgical robot based on magnetic tracking," in *2017 IEEE International Conference on Information and Automation (ICIA)*, 2017, pp. 1143–1149.

[8] J. Wang, Y. Lu, C. Zhang, S. Song, and M. Q.-H. Meng, "Pilot study on shape sensing for continuum tubular robot with multi-magnet tracking algorithm," in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2017, pp. 1165–1170.

[9] C. R. Taylor, H. G. Abramson, and H. M. Herr, "Low-latency tracking of multiple permanent magnets," *IEEE Sensors Journal*, vol. 19, no. 23, pp. 11 458–11 468, 2019.

[10] N. Sebkhi, N. Sahadat, S. Hersek, A. Bhavsar, S. Siahpoushan, M. Ghoovanloo, and O. T. Inan, "A deep neural network-based permanent magnet localization for tongue tracking," *IEEE Sensors Journal*, vol. 19, no. 20, pp. 9324–9331, 2019.

[11] G. Pittiglio, M. Brockdorff, T. da Veiga, J. Davy, J. H. Chandler, and P. Valdastri, "Collaborative magnetic manipulation via two robotically actuated permanent magnets," *IEEE Transactions on Robotics*, vol. 39, no. 2, pp. 1407–1418, 2022.

[12] J. Stuelpnagel, "On the parametrization of the three-dimensional rotation group," *SIAM Review*, vol. 6, no. 4, pp. 422–430, 1964.

[13] C. J. Taylor and D. J. Kriegman, "Minimization on the lie group so (3) and related manifolds," 1994.

[14] G. Gallego and A. Yezzi, "A compact formula for the derivative of a 3-d rotation in exponential coordinates," *Journal of Mathematical Imaging and Vision*, vol. 51, no. 3, p. 378–384, Aug. 2014.

[15] Å. Björck, *Numerical Methods for Least Squares Problems*. Society for Industrial and Applied Mathematics, 1996.

[16] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. Society for Industrial and Applied Mathematics, 1996.

[17] MathWorks, "Least-squares (model fitting) algorithms," https://www.mathworks.com/help/optim/ug/least-squares-model-fitting-algorithms.html#mw_e9424223-f6a8-4dc1-8b91-3b693dd54123, accessed: 18-Sep-2024.

[18] K. Baker, "Singular value decomposition tutorial," *The Ohio State University*, vol. 24, p. 22, 2005.

[19] C. Eckart and G. Young, "The approximation of one matrix by another of lower rank," *Psychometrika*, vol. 1, no. 3, pp. 211–218, 1936.

[20] P.-Å. Wedin, "Perturbation theory for pseudo-inverses," *BIT Numerical Mathematics*, vol. 13, pp. 217–232, 1973.

[21] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[22] R. Martí, "Multi-start methods," *Handbook of metaheuristics*, pp. 355–368, 2003.