FACULTY OF ENGINEERING AND ARCHITECTURE

High Resolution Three-axis Force Sensing for Robotic Fingertips

Lowiek Van den Stockt
Student number: 01905063

Supervisor: Prof. dr. ir. Francis wyffels
Counsellor: Ir. Remko Proesmans

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Electrical Engineering - Communication and Information Technology

Academic year 2023-2024

GHENT UNIVERSITY

Acknowledgment

First and foremost, I would like to express my deepest gratitude to my supervisor, prof. dr. ir. Francis wyffels, for the oppertunity that has been given to me, both in this masters dissertation and the Hardware Design project. It is inspiring to get to work alongside passionate individuals and to possibly contribute a small part to the common goal of the AIRO IDLab research group. I am also extremely grateful to my counsellor, ir. Remko Proesmans, for the daily guidance of the dissertation. Without his aid and feedback, this would not have been possible.

It wasn't always easy to find the time and focus required for this dissertation, mainly because of my membership in UGent Racing. Still, everyone involved was understanding and compromises could be made. The value of this cannot be underestimated and is truly appreciated.

I would like to extend my sincere thanks to my fellow graduate students and friends at UGent Racing, for their camaraderie and for the stimulating discussions and support during the challenging moments of this journey. Your friendship and advice have made this experience memorable.

Finally, my heartfelt appreciation goes to my family, especially to my parents, for their unwavering support and encouragement. It cannot be mistaken how valuable this support is throughout the development of an individual.

Thank you all for your support and encouragement.

Permission of use

The author(s) gives (give) permission to make this master dissertation available for consultation and to copy parts of this master dissertation for personal use. In all cases of other use, the copyright terms have to be respected, in particular with regard to the obligation to state explicitly the source when quoting results from this master dissertation.

Lowiek Van den Stockt

May 2024

Notes related to the master’s thesis

This master’s dissertation is part of an exam. Any comments formulated by the assessment committee during the oral presentation of the master's dissertation are not included in this text.

Disclaimer on the use of generative AI

Generative AI models such as GPT-3.5, GPT-4 and GPT-4o from Open AI were used during this master's dissertation. Specfically, these models were utilized to check paragraphs for spelling mistakes and to improve general sentence structure. The output from these models were not directly copy-pasted; instead, bits and pieces were used as inspiration to improve the quality of said paragraphs.

Abstract

The world of robotics is transitioning from controlled industrial settings where everything is carefully planned to more generic environments where robots must handle diverse situations and collaborate with humans. The goals are noble: for example, to create machines that can assist people with household chores when they are unable to do so themselves. Various challenges must be addressed, including the development of better sensors and actuators, as well as advanced machine learning algorithms to solve complex tasks such as fragile object manipulation and cloth (un)folding.

One important but often overlooked sense when handling fragile or compliant objects such as cloth, fruit, or cables is the sense of touch. For example, when manipulating fruit, one needs to regulate the gripping force to avoid damaging it. For the manipulation of compliant objects like cables, the local feedback provided by the sense of touch is vital for guiding the cable.

In this thesis, a sensor is developed that predicts a three-dimensional force vector field on its surface. This sensor can be integrated into the fingertips of common grippers such as the Robotiq 2F-85 or the Schunk EGU and EGK models. Alongside the sensor construction, a complementary automatic, data-driven method is developed to calibrate the sensor's force prediction model. All resources are open-source and can be found in this GitHub repository [1].

Keywords: tactile sensing - magnetic taxels - force sensing - general purpose robotics

[1] https://github.com/LowiekVDS/masters-thesis

High Resolution Three-axis Force Sensing for Robotic Fingertips

Lowiek Van den Stockt
AIRO IDLab
Ghent University
Ghent, Belgium
lowiek.vandenstockt@ugent.be

ir. Remko Proesmans
AIRO IDLab
Ghent University
Ghent, Belgium
remko.proesmans@ugent.be

prof. dr. ir Francis wyffels
AIRO IDLab
Ghent University
Ghent, Belgium
francis.wyffels@ugent.be

***Abstract*—Tactile sensing is critical in robotics for grasping and manipulating deformable or fragile objects. Existing methods, such as using external cameras or standardized pneumatic grippers, have limitations in handling these tasks effectively. This work explores the development of a compact and scalable magnetic tactile sensor array designed to fit on common robotic grippers. The proposed sensor employs Hall sensors and a magnet array, translating magnetic field changes into a three-dimensional force field. This thesis presents the sensor's design, calibration methods, and performance evaluation, contributing to the advancement of tactile sensing in robotics.**

***Index Terms*—Tactile sensing, magnetism, taxels, force sensing, general purpose robotics**

I. Introduction

Tactile sensing is vital for robotics, particularly for manipulating deformable or fragile objects in various industries, including general-purpose robotics and agriculture. Standardized pneumatic grippers are ineffective for these tasks due to the complexity and degrees of freedom involved in deformable objects [1], [2]. Traditional methods, such as using external cameras, face limitations like occlusion and inability to measure critical parameters like slip force and grasping force [1].

Incorporating tactile sensing into robotic grippers offers a solution. General-purpose robots, such as those from Universal Robots [1], typically use Force-Torque sensors or joint force measurements. However, these are insufficient for deformable object manipulation, necessitating the development of specialized tactile sensors. Tactile feedback is essential for precise manipulation, similar to how the sense of touch is crucial for human tasks.

This research, conducted within the AIRO team at IDLab, Ghent University, aims to advance tactile sensing technologies. AIRO's broad research includes social robotics, machine learning, and hardware development, with a long-term goal of creating a general-purpose robot capable of complex tasks such as folding cloth and pouring drinks.

Various tactile sensing technologies have been explored in literature, including piezoresistive fibers [3], silicone structures with embedded liquid metal [4], and optoelectronic sensors [5]. This thesis focuses on magnetic tactile sensing technology, employing an array of Hall sensors and magnets [6]–[9], offering a compact and scalable design.

A 4x8 sensor array was developed, fitting common robotic grippers like the Robotiq 2F-85 and Schunk EGK. A two-stage force model converts magnetic measurements into a three-dimensional force field. The design ensures scalability and integration with low-level microcontrollers, reducing costs and encouraging further research. An automatic calibration procedure, similar to previous works [10]–[12], and an alternative calibration method were proposed, leading to a workshop paper [13] for the International Conference on Robotics and Automation.

This extended abstract outlines the sensor design, calibration methods, and performance evaluation. All resources are open-sourced and available on GitHub[2].

II. Sensor design

A. Mechanical construction

The finger design, depicted in Fig. 1, consists of five layers. The core component is the Printed Circuit Board (PCB), which houses a 4-by-8 array of Melexis MLX90393 Hall effect sensors. On top of the sensor array, a dome structure is positioned, similar to the one described in [14], [15]. This structure is 3D-printed using a Formlabs Form 2 with "Flexible 80A Resin" from Formlabs. After printing, the dome structure is cleaned with isopropyl alcohol to remove any unhardened resin and then cured under ultraviolet light for 10 minutes per side.

Axially magnetized cylindrical NdFeB magnets, with a diameter of 1.5 mm and a height of 1 mm, are superglued into the magnet cavity of the dome structure. The dome structure itself is also superglued to the PCB. The bottom and top casings are printed using a Prusa MK3 3D-printer in standard black PLA. These casings are bolted together using two M2 bolts. The bottom casing facilitates the attachment of the finger to a Robotiq 2F-85 gripper or other grippers, such as the Schunk EGU and EGK grippers. Importantly, the top casing secures the top silicone layer through L-shaped holes around the PLA-silicone interface.

[1]https://www.universal-robots.com/

[2]https://github.com/LowiekVDS/masters-thesis

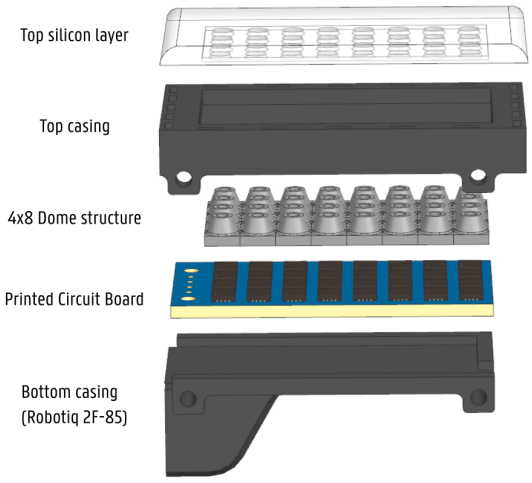


Fig. 1. The general structure of the finger.

The top silicone layer is fabricated using Silicone Addition Colorless 50 by Silicones and More [3]. The mold employed to cast this layer is depicted in Fig. 2. An unsoldered PCB is placed into the top case, along with 3D-printed dummy taxels made from PLA. The casting top is then positioned over the entire assembly. Silicone is poured into the top, and the cast is left to cure for one day. Subsequently, the casting top, PCB, and dummy taxels are removed, resulting in a top casing with the silicone layer adhered to it. It is important to note that the silicone layer features small cup-shaped holes that precisely align with the domes below. These holes must be shallow; if they are too deep, the silicone layer may be pressed onto the bottom of the dome structure, between the taxels. This would dissipate the force into the PCB without effectively actuating the taxels.

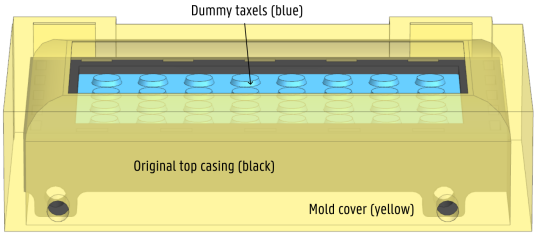


Fig. 2. The molding setup. The silicone is poured into the top opening. Note how dummy taxels are used to create the shallow cup shaped openings in the final cast. The silicone top layer is connected to the top casing because the silicone pours into the L-shaped holes of the top casing.

B. Electrical design

The PCB (Fig. 3) is an FR-4 board with four copper layers. The top layer houses the 32 MLX90393 sensors, arranged in a 4x8 grid. The pitch between the taxels is 4.5 mm. The sensors measure the magnetic field in three dimensions. The Z-axis comes out of the top of the sensor and the X and Y-axis are along the horizontal plane. These sensors communicate using the I^2C protocol, at a speed of 400kHz. Each sensor can be configured with one of four I^2C addresses: 0x0C, 0x0D, 0x0E, and 0x0F. To assign unique addresses to all sensors, they are organized into eight rows, each containing four sensors. Each row has its own I^2C bus. The eight buses are interconnected via a TCA9548APWR I^2C multiplexer, which has its own I^2C address on a ninth main bus that connects to a microcontroller. The I^2C master used is a Raspberry Pi Pico W. Although any microcontroller with an I^2C driver can be used.

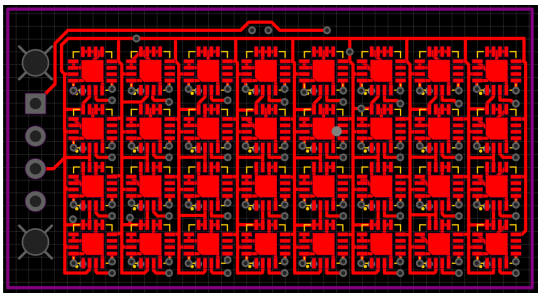


Fig. 3. The top layer of the four-layer Printed Circuit Board (PCB) containing 32 MLX90393 chips. The pitch between taxels is 4.5mm.

Alternative versions of the main 4x8 PCB were developed, including 1x4, 2x2, and 1x1 configurations. These variants can be used in conjunction with the main 4x8 version to accommodate more complex sensor shapes. The MLX90393 chips can be configured with different filter settings, as per datasheet [**?**]. The sensors are configured with an oversampling ratio (OSR) of 2, a digital filter set to 1, gain of 7 and a resoltuion of 0. The sensitivity per Least Significant Bit (LSB) is $0.150\mu T$/LSB for the X and Y directions and $0.242\mu T$/LSB for the Z axis. These settings are important, since they determine the conversion time: the time needed for the chip to produce a measurement.

C. Sensor readout schedule

The microcontroller acts as the bus master and configures the sensors to operate in single read mode. Initially, the microcontroller sends a command to start a magnetic field measurement to each sensor. This is done row by row, changing the multiplexer state every four sensors by sending a different command. After initiating the measurements, a specific conversion time must be observed, the duration of which depends on the sensor settings.

Instead of idling during this period, the microcontroller utilizes this time to send the measurements from the previous readout cycle to the computer over a serial connection of 400kbps. It begins by sending the start byte 0xAA, followed by all the measurements in order. Each sensor produces six bytes: two bytes per axis, resulting in a total of 193 bytes sent per cycle. After the conversion time, the microcontroller communicates with each sensor to read the measurements

[3]https://www.siliconesandmore.com/en/silicone-addition-clear-50-hard.html

register and stores the data in memory for transmission in the next cycle. This process then repeats.

III. Calibration procedure

An automatic calibration procedure was developed to aid in calibrating the 4-by-8 sensor array. The calibration itself was done by attaching a probe to an UR3e robot from Universal Robots. The robot has an integrated F/T-sensor used to collect the labels required for training. On the tool flange a dome-shaped probe tool was attached. The sensor array was attached to the table below the probe, see Fig. 4.

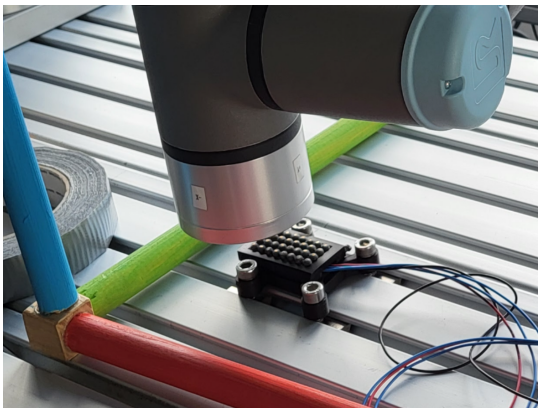


Fig. 4. The setup for the automatic calibration procedure. The sensor is mounted to the table (without silicone top layer). The tool flange has a special dome-shaped probe tool attached to it, which applies preprogrammed forces to all the taxels.

First the robot executes a homing procedure (making use of the force mode in the UR3e) aiming to find the taxel position in the 3D-space. After having found the taxel positions, it executes a force-based calibration curve, seen in Fig. 8, 9 as the blue curves. The robot moves in such a way to maintain a set force. The setpoint is changed at a rate of 200Hz.

IV. Two-stage force model

A. Analytical background

1) Perfect magnet alignment: To better understand which model and preprocessing steps would be most effective, an analytical study was conducted. The magnetic field of an axially magnetized cylindrical magnet is axially symmetrical around the Z-axis. Thus, the field can be represented by a vector $\mathbf{B}(\rho, z)$ at a point $P(\rho, z)$. Assuming the point P is sufficiently distant from the magnet's center, the magnet can be approximated as a dipole. The components of the magnetic field are then given by:

$$\begin{cases} B_z(\rho, z) = \frac{2z^2 - \rho^2}{r^5} \\ B_\rho(\rho, z) = \frac{3\rho z}{r^5} \end{cases} \tag{1}$$

The scaling factor $\frac{\mu_0 \mu}{4\pi}$ is omitted for clarity. Additionally, $r = \sqrt{\rho^2 + z^2}$ represents the distance from the magnet's center, which is strategically placed at the origin. When a taxel (magnet) is excited by an applied force, the magnet shifts

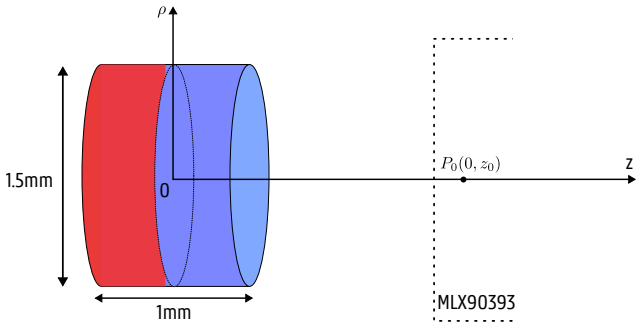


Fig. 5. The coordinate frame used for the analytical discussion. It includes the magnet and the taxel sensor, an MLX90393. The point P_0 is the idle point when no force is applied to the taxel.

relative to the three-dimensional Hall sensor below. Assuming the dome structure follows Hooke's law, the structure behaves like an ideal spring.

If the magnet is perfectly aligned within the dome structure, the idle point P_0 without any applied force lies on the magnet's symmetrical axis, i.e., $P_0 = (0, z_0)$. The application of a force can be decomposed into a normal force F_z and a planar force F_ρ. The normal force shifts the magnet solely in the Z-direction, while the planar force shifts the magnet in the ρ direction. Applying a force F_z shifts the magnet to $P_1 = (0, z_0 + \Delta z) = (0, z_1)$. The change in the magnetic field is expressed as:

$$\begin{cases} \Delta B_z(P_1) = B_z(0, z_0 + \Delta z) - B_z(0, z_0) = \frac{2}{(z_0 + \Delta z)^3} - \frac{2}{z_0^3} \\ \Delta B_\rho(P_1) = B_\rho(0, z_0 + \Delta z) - B_\rho(0, z_0) = 0 \end{cases} \tag{2}$$

This shows that the change in the magnetic field is inversely proportional to the cube of the distance, and by Hooke's law, to the applied normal force. Similarly, applying a force F_ρ shifts the magnet to $P_2 = (\Delta\rho, z_0)$. Assuming the movement in the ρ direction is small compared to the distance z_0, the change in the magnetic field at P_2 can be approximated as:

$$\begin{cases} \Delta B_z(P_2) \approx 0 \\ \Delta B_\rho(P_2) \approx \frac{3\Delta\rho z_0}{r^5} \end{cases} \tag{3}$$

This indicates that the change in the magnetic field is linear with respect to the applied planar force.

2) Misaligned Magnets: Manufacturing defects can cause the magnet to be misaligned within the dome structure. In such cases, the idle point P_0 without applied force will not lie on the symmetrical axis of the magnet, i.e., $P_0 = (\rho_0, z_0)$. It suffices to realize that an arbitrary linear transformation can be applied to align these magnets again. Note that because this linearity, the fitted models will include this transformation on the condition that the models take in all measurement axes

and makes predictions for all force axes. The calculations and general structure of the magnetic field changes will thus remain similar, though the expressions will be more complex.

B. Stage I: per-force taxel model

The goal of the first stage is to predict the three-dimensional force $\vec{F_i}$ applied to taxel i, based on the three-axis magnetic field measurements of that taxel; x_i, y_i and z_i. An overview of the first stage can be seen in Fig. 6.

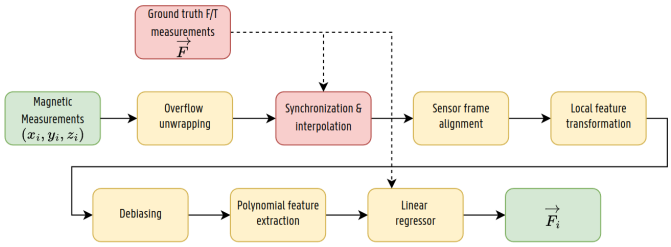


Fig. 6. An overview of the first stage for taxel i. Most of the steps are preprocessing. The red blocks, synchronization & interpolation and the F/T-sensor measurements, are only used during training.

1) Overflow unwrapping: The MLX90393 sensors report their measurements as a two-byte unsigned integer per measurement axis. Sometimes depending on the position of the magnets above the taxels, the measurements are fairly close to 0 or 65535, the maximal two-byte unsigned integer. So when applying a force that moves a magnet, an overflow from 65535 to 0 or vice versa might occur. This overflow causes discontinuities which negatively influence the fitting of the linear model.

To fix this, per datastream, so per axis per taxel, an integer counter is kept. When an absolute jump of 65535 is detected, the counter is increased or decreased by one, depending on the direction of the jump. Finally, the counter is multiplied with 65535 and added to the original signal.

2) Synchronization & interpolation: The sensor array measurements are not time synchronized with the F/T-sensor measurements. Indeed, because they come from two independent processes, there is a constant but unknown latency difference. To estimate this latency, analysis was done into the rate-dependant hysteresis caused by this lag. Suppose a time signal $X(t) = \sin(2\pi ft)$ is delayed by t_0, producing a new signal $Y(t) = \sin(2\pi ft - 2\pi ft_0)$. Plotting $Y(t)$ against $X(t)$ will show the typical hysteresis loops, due to the introduced phase lag. The phase lag $-2\pi ft_0$ is dependent on the frequency of the excitation, which is why this is sometimes called rate-dependent hystersis.

Similarly, this effect will be seen when plotting the magnetic measurements (called $X(t)$) against the applied forces ($Y(t)$), since $Y(t)$ is produced by sending $X(t)$ through a black box system with time delay (the latency difference) t_0. This latency can be found by sending $Y(t)$ through a new system that makes the signal lead ahead by t_1. When t_1 is chosen to be equal to t_0, the hysteresis should have disappeared. In practice, t_0 was determined by iterating over different values for t_1 and choosing the time that makes the hysteresis disappear the most. To aid in this, the 3Hz measurements of the normal force calibration stage is taken as the higher frequency will amplify the rate-dependent hysteresis effect.

In Fig. 7, the applied force $Y(t)$ is plotted against the measurements $1/z_0$ $(X(t))$ of the first taxel. The delayed version of the applied force called $Z(t)$ is plotted against $X(t)$. The hysteresis curve became the smallest with a delay of 80ms. So the latency difference was deemed to be 80ms, where the magnetic measurements lag behind.

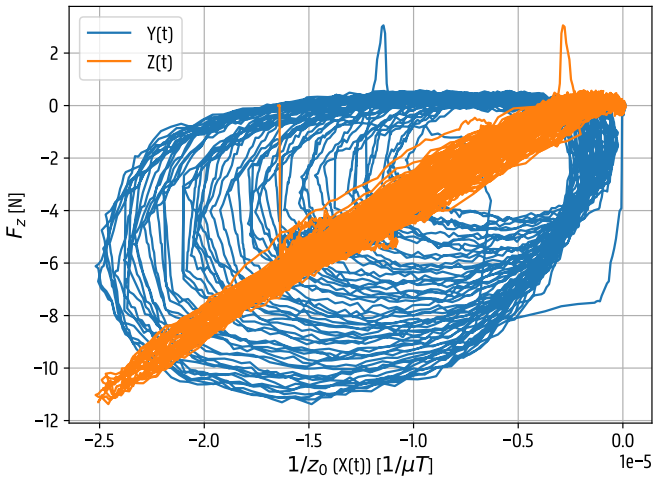


Fig. 7. The lag hysteresis of the uncompensated signal (blue) and the signaled delayed by 80ms. Notice how at this delay the hysteresis decreases. This was used to estimate the latency difference at 80ms.

3) Interpolation: The two data streams are sampled at different rates, resulting in varying lengths, which complicates model fitting and necessitatesretainingthetime dimension. To addressthis, a unified set oftimestamps is created by combining all timestamps from both data streams. Subsequently, linear interpolation is applied to ensure every timestamp corresponds to a data point in both streams.

4) Sensor frame alignment: The robotic arm executes a homing procedure to automatically find the positions of the taxels in the sensor array. An extra transformation from TCP to the sensor frame can also be set to accommodate for non-default orientations during calibration. Throughout the calibration experiments, setup errors occur causing the frame in which the force is applied (ie. the base frame of the robot) to not match the local sensor frame. Luckily, the calibration data also includes the X and Y movement of the robotic arm with respect to the base frame. This meant it is possible to calculate and correct this alignment error. This preprocessing step thus aims to rotate the X and Y components of the measurements by angle θ, for all taxels i:

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} * \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{4}$$

5) Local feature transformation: The idle height Z_0 between the magnet and the sensor is 3.27 mm. Data indicates

that under applied forces of about 10 N, the magnet does not move more than 1 mm up and down and not more than 0.5 mm along the planar direction ρ. Therefore, it is assumed that the planar magnetic measurements x_0 and y_0 are linear with respect to the applied planar forces F_x and F_y. Furthermore, the normal component F_z is inversely proportional to the cube of z_i. Because the first stage is a linear regressor, a feature transformation is applied to invert the z-component z_i:

$$\begin{cases} x_i^{'} & = x_i \\ y_i^{'} & = y_i \\ z_i^{'} & = \frac{1}{z_i} \end{cases} \tag{5}$$

No special attention is needed to address the potential numerical instabilities, such as division by zero, because the magnetic field will always be non-zero under normal operation.

6) Debiasing: After the feature transformation, the features are individually debiased. Debiasing is implemented on a feature-by-feature basis by subtracting the mean of the initial 100 samples from all samples of that feature. This step ensures that the models are trained on the variations in the magnetic field rather than on the absolute values. Practically, this debiasing step can be incorporated into the sensor's boot procedure.

7) Feature extraction: As shown previously, the relationship between the Z-measurements and the applied normal force is (inversely) cubic, while the relationship between the planar force and the planar measurements is linear. Additionally, extra nonlinearities are likely introduced by errors in assumptions such as Hooke's law and the models used for the magnetic fields themselves. The linear regressor aims to linearize these around the idle point. Before being able to fit a linear regressor, new polynomial features are extracted. The relationship between these new feature sets and the force labels then becomes linear, allowing the use of a standard linear regressor.

8) Linear regression: Per taxel, the calibration data (after preprocessing) is randomly split into a 80%/20% train/test set. A regular linear regressor was used.

C. Stage II: cross-talk compensation

A more accurate description of the force model devised in Stage I can be given by using the superposition principle:

$$\overrightarrow{F_i} = \overrightarrow{L_i} + \sum_{j \epsilon N_i} \overrightarrow{K}_{j,i} * \overrightarrow{G_j} \tag{6}$$

Where N_i is the set of direct neighbors of taxel i (including diagonals and the taxel iteself), $\overrightarrow{K}_{j,i}$ is a model for the cross-talk component of the force in taxel i caused by taxel j, $\overrightarrow{L_i}$ is a bias term and $\overrightarrow{G_j}$ is the actual, cross-talk compensated force of taxel j. In the above equation Stage I technically gives us the force with cross-talk, $\overrightarrow{F_i}$, while Stage II aims to produce the corrected $\overrightarrow{G_i}$. The complexity of this problem heavily depends on the chosen structure of the models $\overrightarrow{K}_{j,i}$.

1) Linear crosstalk asssumption: When the assumption is made that cross-talk caused by neighbouring taxels is linear, the model $\overrightarrow{K}_{j,i}$ can be written as a 3x3 matrix. By combining all the taxels into a single set of equations, the problem can be written as a single block matrix:

$$\begin{bmatrix} \overrightarrow{F_0} \\ \overrightarrow{F_1} \\ \vdots \\ \overrightarrow{F_{31}} \end{bmatrix} - \begin{bmatrix} \overrightarrow{L_0} \\ \overrightarrow{L_1} \\ \vdots \\ \overrightarrow{L_{31}} \end{bmatrix} = \begin{bmatrix} \overrightarrow{1_3} & \overrightarrow{K}_{0,1} & \dots & \overrightarrow{K}_{0,31} \\ \overrightarrow{K}_{1,0} & \overrightarrow{1_3} & \dots & \overrightarrow{K}_{1,31} \\ \vdots & & \ddots & \vdots \\ \overrightarrow{K}_{31,0} & \overrightarrow{K}_{31,1} & \dots & \overrightarrow{1_3} \end{bmatrix} * \begin{bmatrix} \overrightarrow{G_0} \\ \overrightarrow{G_1} \\ \vdots \\ \overrightarrow{G_{31}} \end{bmatrix} \tag{7}$$

Which can be notationally simplified as:

$$(\mathcal{F} - \mathcal{L}) = \mathcal{K} * \mathcal{G} \tag{8}$$

A compensated force model $\mathcal{G}$ can then be found by inverting the matrix $\mathcal{K}$:

$$\mathcal{G} = \mathcal{K}^{-1} \cdot (\mathcal{F} - \mathcal{L}) \tag{9}$$

During the calibration process the measurements of all other taxels were also recorded. Therefore it is possible to determine the matrices $\overrightarrow{K}_{j,i}$ for all i by fitting a linear regression model (including intercept) to predict the force $\overrightarrow{F_i}$ solely with the measurements from the neighbouring forces $j \in N_i$. The bias term $\overrightarrow{L_i}$ is calculated as the mean of all intercepts of the created models for $j \in N_i$.

V. Results

A. Stage I

In the first stage, the degree of polynomial features used was not specified. Different degrees of $k = 1$, $k = 3$, and $k = 5$ were tested. The R^2 scores for the models are 0.90495, 0.93886, and -24.01625, respectively. The Normalized Mean Squared Error (NMSE) of the models with different degrees was calculated on the test sets and averaged across all 32 taxels. Normalization was done by dividing the Mean Squared Error by the difference between the maximum and minimum sample. The linear model ($k = 1$) scored an NMSE of 0.03263, the third-order model 0.01693, and the fifth-order model 0.23346. The third-order model performed the best, corresponding to the analytical model. The fifth-order model heavily overfits, resulting in poor test scores. Notice how the linear model is not much worse than the third-order model. Given that the third-order model has 60 parameters compared to the 12 of the linear model, the linear model is an interesting choice for computationally constrained applications. An example of the first stage of taxel 0 can be seen in Fig. 8.

B. Stage II

The recorded calibration data contains information of the neighbouring taxels. When predicting the forces on these neighbours using stage I, this should be zero, since during calibration only a single taxel was excited. To measure the performance of the second stage, the Root Mean Squared Error

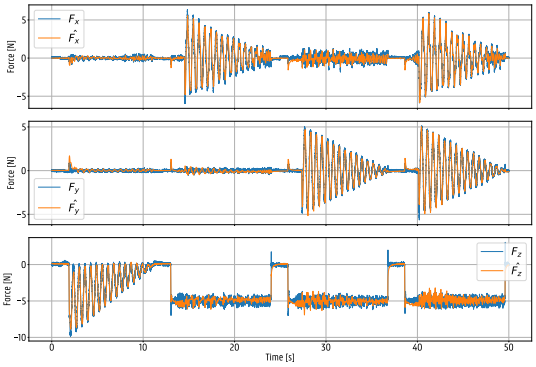


Fig. 8. The applied forces and estimated forces after stage I from taxel 0. The model used uses third-order polynomial feature extraction.

(RMSE), or in this case also the power, of the neighbouring taxel predictions was calculated and averaged across neighbours, before and after the second stage. Before the second stage, the average power was equal to $41.268mW$. After the second stage, this average dropped to $14.757mW$. A visual comparison was made of the predictions of the force on taxel 0 before (blue) and after (orange) stage II when only exciting taxel 1 are shown in Fig. 9.

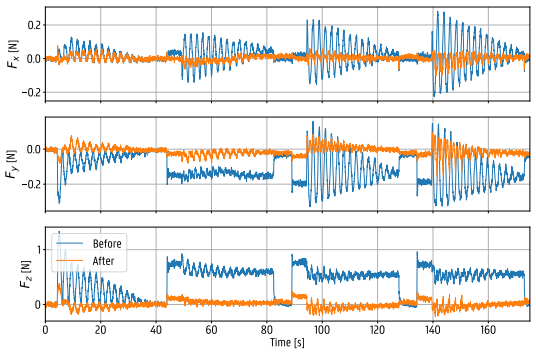


Fig. 9. A comparison between the force predictions before and after stage II of taxel 0, when only taxel 1 was excited. The second stage greatly reduces cross-talk.

C. Readout performance

The two-stage model was implemented on a Raspberry Pi Pico W using 32-bit float arethmatic to provide a readily integrateable prototype. To aid in performance, the predictions are converted to two bytes when sending over Serial. The microcontroller achieves a datarate of 50.0 Hz when only enabling Stage I. Adding Stage II, decreases the datarate to 26.0 Hz.

VI. Conclusion

This thesis addresses the need for advanced tactile sensing in robotics. A novel magnetic field sensor was developed for predicting 3D force vectors on robotic grippers like the Robotiq 2F-85 and Schunk EGU/EGK models. Key contributions include the sensor's construction and a data-driven calibration method for a two-stage force prediction model, implemented on a Raspberry Pi Pico W. Resources are open-sourced on GitHub [4]. Future work should improve mechanical constructions, explore alternative materials, top layer structures and should also try to solve complex manipulation tasks using this sensor.

References

[1] D. K. A. Billard, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 364, no. 6446, pp. 1–8, 2019.

[2] R. Yagawa, R. Ishikawa, M. Hamaya, K. Tanaka, A. Hashimoto, and H. Saito, "Learning food picking without food: Fracture anticipation by breaking reusable fragile objects," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 917–923.

[3] L. Zlokapa, Y. Luo, J. Xu, M. Foshey, K. Wu, P. Agrawal, and W. Matusik, "An integrated design pipeline for tactile sensing robotic manipulators," 2022.

[4] E. Judd, B. Aksoy, K. M. Digumarti, H. Shea, and D. Floreano, "Slip anticipation for grasping deformable objects using a soft force sensor," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 003–10 008.

[5] R. Proesmans and F. wyffels, "Augmenting off-the-shelf grippers with tactile sensing," 2023.

[6] T. P. Tomo, M. Regoli, A. Schmitz, L. Natale, H. Kristanto, S. Somlor, L. Jamone, G. Metta, and S. Sugano, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2584–2591, 2018.

[7] H. Kristanto, P. Sathe, A. Schmitz, C. Hsu, T. P. Tomo, S. Somlor, and S. Sugano, "Development of a 3-axis human fingertip tactile sensor based on distributed hall effect sensors," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–7.

[8] P. Sathe, A. Schmitz, T. P. Tomo, S. Somlor, S. Funabashi, and S. Shigeki, "Fingertac - an interchangeable and wearable tactile sensor for the fingertips of human and robot hands," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 813–10 820.

[9] K. Dai, X. Wang, A. Rojas, E. Harber, Y. Tian, N. Paiva, J. Gnehm, E. Schindewolf, H. Choset, V. Webster-Wood, and L. Li, "Design of a biomimetic tactile sensor for material classification," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10 774–10 780.

[10] R. Bhirangi, T. Hellebrekers, C. Majidi, and A. Gupta, "Reskin: versatile, replaceable, lasting tactile skins," in *2021 Conference on Robotic Learning (CoRL)*, 2021.

[11] T. Le Signor, N. Dupré, and G. F. Close, "A gradiometric magnetic force sensor immune to stray magnetic fields for robotic hands and grippers," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3070–3076, 2022.

[12] S. Xie, Y. Zhang, M. Jin, C. Li, and Q. Meng, "High sensitivity and wide range soft magnetic tactile sensor based on electromagnetic induction," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 2757–2766, 2021.

[13] L. V. den Stockt, R. Proesmans, and F. wyffels, "Automatic calibration for an open-source magnetic tactile sensor," 2024.

[14] A. C. Holgado, N. Piga, T. P. Tomo, G. Vezzani, A. Schmitz, L. Natale, and S. Sugano, "Magnetic 3-axis soft and sensitive fingertip sensors integration for the icub humanoid robot," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–8.

[15] P. Sathe, A. Schmitz, T. P. Tomo, S. Somlor, S. Funabashi, and S. Shigeki, "Fingertac - an interchangeable and wearable tactile sensor for the fingertips of human and robot hands," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 813–10 820.

[4] https://github.com/LowiekVDS/masters-thesis

Table of contents

List of Figures

List of Tables

List of Acronyms

ASIC Application-specific Integrated Circuit.

F/T-sensor Force-Torque sensor.

FDM Fused Deposition Modeling.

FPS Frames Per Second.

GUI Graphical User Interface.

I2C Inter-integrated circuit.

IR Infrared.

LED Light Emitting Diode.

LSB Least Significant Bit.

LSTM Long Short-Term Memory.

MISO Master In Slave Out.

MOSI Master Out Slave In.

NMSE Normalized Mean Squared Error.

OSR Oversampling ratio.

PCB Printed Circuit Boards.

PLA Polyactic Acid.

R/W Read/Write.

RMSE Root Mean Squared Error.

RNN Recurrent Neural Network.

SCK Serial clock.

SLA Stereolithography.

SPI Serial peripheral index.

SS Slave Select.

TCP Tool Center Point.

UDP User Datagram Protocol.

1

Introduction

Tactile sensing is a crucial area of research in robotics, particularly for addressing complex grasping problems involving deformable or fragile objects. These challenges are prevalent in various industries, including general-purpose robotics, where robots must manipulate intricate objects such as mugs and cloth and in agriculture, where robots handle delicate items like fruits, which require precise control to avoid damage.

While in industry the standardized pneumatic grippers that hold objects by sucking is well-studied, these kind of grippers are not that useful when it comes to manipulating them [2]. The complexity of grasping deformable objects is explained by the many degrees of freedom introduced by deformable objects [3].

While for example in cloth folding the robot setups typically use external camera to detect and manage these tasks, their use cases are limited to non-transparent objects. Furthermore, the object will be partially occluded when the robot is manipulating it [2].

When a robot is actively manipulating an object, it often obstructs the camera's view, making it difficult to gather necessary information. Additionally, cameras struggle to accurately measure critical parameters like slip force and grasping force, both of which are essential for handling fragile objects without causing damage.

An alternative to this is the inclusion of tactile sensing in the gripper of the robot. Most general purpose robots, like the models from Universal Robotics, provide Force-Torque sensors (F/T-sensor) or the forces felt by the different joints. While useful for rigid object manipulation or industrial assembly tasks where a binary local object detection suffices, this is not enough for the manipulation of deformable object. So, researchers and integrators have to develop and add their own tactile sensors.

To further illustrate the importance of tactile sensing, consider a thought experiment where one loses their sense of touch. Everyday tasks would become exceedingly challenging without tactile feedback. This analogy underscores the necessity of tactile sensing in robotics, as it provides the detailed, localized feedback required for effective manipulation.

In summary, advancing tactile sensing research is vital for the development of robots capable of delicate and

precise manipulation, which is essential for applications in both general-purpose robotics and specific industries like agriculture.

1.1 AIRO

This thesis was done in the context of AIRO. AIRO is a research team part of the research group IDLab, connected to Ghent University. Their research is rather broad: social robotics, machine learning and robotics hardware are a few examples. One of the long-term goals is to combine all research into a general purpose robot, that autonomously can handle a broad range of complex grasping tasks, such as folding cloth and pouring drinks. One of the active research paths is into different kind of technologies for tactile sensing.

1.2 Related work

There are quite a lot of different technologies currently being researched. For example, in [4], a knitted tactile sensor was constructed. The sensor consists of two layers, one with horizontally and the other with vertically integrated piezoresistive fibers. A sensing matrix is formed by sandwiching the fibers between conductive electrodes. The application of pressure results in electric signals. Another method involves manufacturing a silicone structure with embedded liquid metal [5]. By measuring the capacitance of embedded sensing electrodes alongside this metal, pressure can be measured since the relative distance between these electrodes and the metal changes. Optoelectronic technologies can also be used for tactile sensing, as demonstrated in [6], where infrared (IR) Light Emitting Diodes (LEDs) and IR photoreceivers were used to develop a sensor showing potential for cloth edge tracing and cable manipulation. During my Hardware Design Project [7], also in AIRO, another technology was used. In the project, we developed a prototype for a tactile sensor using the self-interferometry effect of lasers.

One of the more popular technologies, GelSight, is already being commercially distributed. A GelSight sensor consists of an elastomer, a membrane containing black dots in a grid pattern, lighting, and a camera. The idea is that when the elastomer moves in response to a force, the dots also move, which is then tracked by the camera. Although showing impressive results, the sensors are typically bulky and not very accessible in terms of cost.

1.3 Thesis outline

This thesis focuses on tactile sensing technology based on magnetic field or Hall sensors. The main principle is to have a structure holding magnets [8, 9, 10, 11] above an array of Hall sensors, optionally adding a protective layer on top. Others have used a sinusoidally magnetized flexible film, also called magnetic skins, instead [12].

1 Introduction

A compact and scalable sensor design is proposed, making use of magnetic tactile sensing technology. Concretely, a 4x8 sensor array is developed, neatly fitting on the fingertip of the commonly used Robotiq 2F-85 gripper, or the Schunk EGK and EGU grippers. Alongside the sensor construction itself, a two-stage force model is proposed to convert these magnetic measurements into a three-dimensional force field. Special care is taken to ensure the model is scalable and integrable on a low-level microcontroller, further lowering the cost of this technology to incentivize further research. To improve the accessibility of this sensor, an automatic, data-driven calibration procedure similar to [13, 14, 15] is proposed using the commonly used UR3e robot from Universal Robots. To increase the applications of this design, an alternative calibration method was also designed to make the sensor immediately usable, leading to a workshop paper [16] being written for the International Conference on Robotics and Automation.

Chapter two will dive into the design of the sensor. Chapter three will provide insights on the calibration method(s) and the two-stage force model, including an analytical section to back some design choices. The last and fourth chapter will discuss the performance of the sensor.

All resources for this sensor design are open-sourced and can be found on a GitHub repository

[1].

[1] https://github.com/LowiekVDS/masters-thesis

2

Finger Design

In this chapter, the design of the fingertip will be described. The general design is based on a 2x2 prototype designed by ir. Remko Proesmans. The main goals of this master's dissertation were to make the finger design scaleable, modular, and as compact as possible.

2.1 Mechanical Construction

The main finger design (Figure 2.1) consists of array of 32 individual taxels, laid out in a 4x8 grid on a Printed Circuit Board (PCB). The PCB is encased to hold everything together and to be able to integrate nicely with existsing robotic grippers. The finger is topped of by a protective silicone layer. A single taxel consists of multiple layers: the magnetic field sensor, a flexible dome structure and a magnet. The PCB, including the magnetic field sensor is thoroughly discussed in Section **??**. The other components will be discussed here. Since the thesis began with the 2x2 prototype, the discussion of some mechanical components, such as the dome structure and top layer, will cover both the 2x2 prototype and the upscaled version proposed in this dissertation.

When the finger grips an object, force will be applied to the protective silicone layer. This layer will then transfer the force to the dome structures, which causes the magnets to move. The movement of the magnets results in changing magnetic fiels, which is picked up by the magnetic field sensors below the domes. These measurements are processed and send through a two-stage model, which predicts the forces appllied to the individual forces. These forces together form a vector field, which can be used for further processing.

Magnet selection

Inside the dome structure a magnet is embedded. This magnet produces magnetic fields, of which changes are measured using the sensors below the domes. The magnet used was the DZ-001.5-001-G52 from Neomagtec GmbH. The cylindrical magnet is made of NdFeB, is axially magnetized and has a diameter of 1.5mm and a thickness of 1mm.

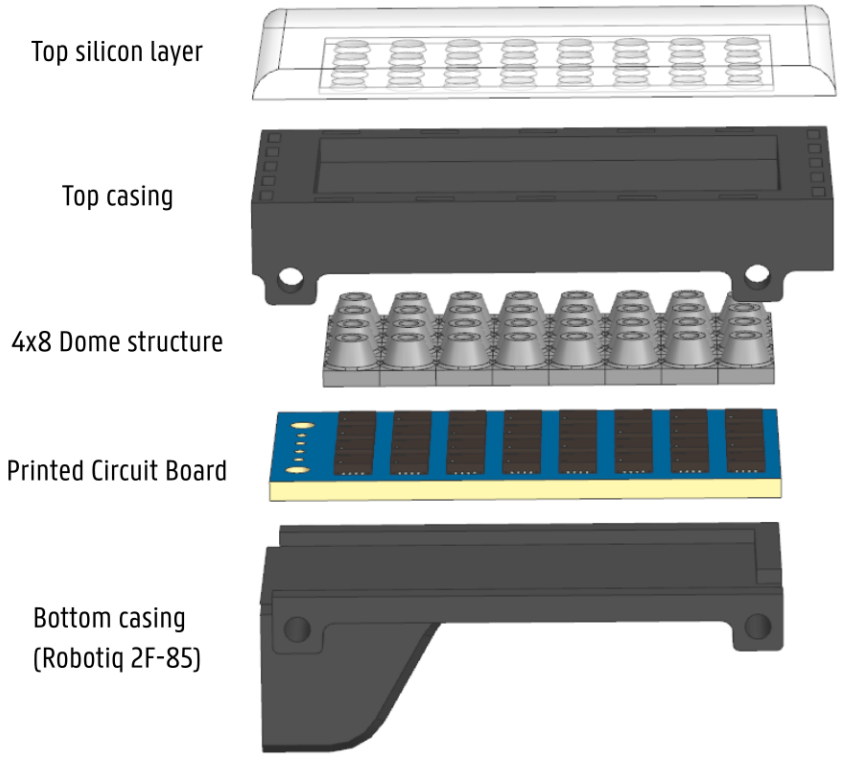


Figure 2.1: The general design of the fingertip, without silicone top layer.

2.1.1 Dome structure

One of the most important structures of the finger are the domes. This slightly flexible structure provides support for the magnet positioned above the Hall sensor. The dome needs to be flexible enough to allow forces applied to it to robustly move the magnet above the sensor. However, it should not be too flexible, as this could result in the magnet "bottoming out". When bottomed out, the dome structure with the magnet physically sits on the top of the sensor, obstructing further movement. Aside from the damage that might occur to the dome structure or the sensor itself, the proposed model will have difficulties with dealing with this, as the assumption is made that the dome structure acts like a mechanical spring.

Additionally, as will be shown in Section 3.1, the magnetic fields produced by the magnet are non-linear. If the domes are designed such that the magnet movements are minimized, a linearization of the magnetic field around the idle point can more accurately predict the magnet movement and, consequently, the applied force.

Furthermore, the height of the magnet is crucial. Placing the magnet higher above the sensor reduces non-linearities but compromises structural integrity due to the increased height of the dome: when a planar force is applied, it introduces higher moments on the thin walls of the dome, causing it to shear. Reducing the height between the magnet and the sensor mitigates this issue but, as shown in Section 3.1, increases non-linearity in the sensor measurements in the planar direction. Additionally, a smaller height might cause the magnet to physically

bottom out.

The main design was inspired on related work done in [17, 18] and designed in Siemens NX. The design can be seen in Figure 2.2. It features a rectangular section containing the sensor. On top of that, a cone shape is placed containing the cylindrical magnet. The dome structure is hollow to allow for mechanical movement. The thickness of the walls are important. Thin walls makes the domes fragile, while thick walls make the domes stiffer, resisting mechanical movement and thus reducing sensitivity.

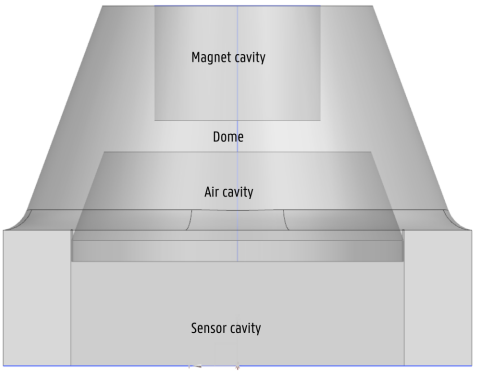


Figure 2.2: A single dome shape used for the 4x8 array.

Dome height tests

Three different dome versions were created with different (total) heights: 3.5mm, 4mm and 4.5mm. They were all manufactured and tested with the manual calibration procedure as outlined in Section **??**. The designs can be seen in Figure **??**.

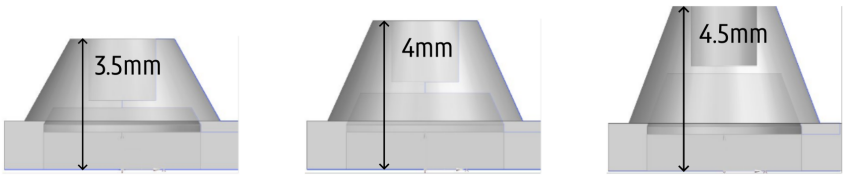


Figure 2.3: The three different dome heights tested.

The 4.5mm version was deemed structurally inadequate, as tests often resulted in sheared or damaged domes. The choice between the 3.5mm and 4mm versions depends on the application. When applying forces of around 10N, the smaller 3.5mm version physically bottomed out, which was not the case for the 4mm version. Thus, the 3.5mm version is suitable only for applications involving smaller forces. Since the maximum normal force in the context of this thesis was set to 10N, the 4mm option was chosen.

Scaling the design

Because the fingertip consists of an array of taxels, the dome structure has to be patterned in a similar fashion. The 2x2 prototypes have a inter-taxel pitch of 6.5mm. The 4x8 version proposed in this dissertation brought this pitch down to 4.5mm. The decreased pitch prompted a slight design change, since the original design was created for the 2x2 prototype. The height of 3.5mm was maintained. Dimensional drawings of the final single taxel used can be found in appendix **??**. The single taxel was finally patterned to create a 4x8 array, as can be seen in Figure 2.4.

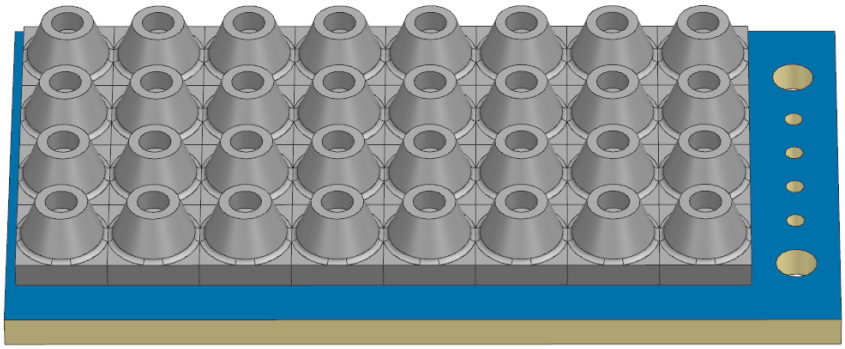


Figure 2.4: The 4x8patterned version of the single taxel dome, placed on the PCB.

Manufacturing

All domes are 3D-printed with a Formlabs Form 2 Stereolithography (SLA) printer. SLA technology utilizes a laser to cure liquid resin into hardened plastic in a layer-by-layer process. This method allows for improved precision and detail compared to traditional Fused Deposition Modeling (FDM) printers, making it ideal for producing complex and small geometries. During printing, the laser traces each layer's cross section, solidifying the resin onto a submerged build platform. As each layer is completed, the platform rises slightly to allow fresh resin to flow beneath the cured section. A layer height of 0.05mm was used.

To ensure the domes were flexible, Formlabs Flexible 80A Resin was used. This elastomer is designed to simulate the flexible properties of TPU or rubber, with a post-cured hardness of 80 on the Shore A scale. The domes were printed with the bottom side facing the print bed, necessitating a support structure that made contact with the bottom side. To preserve the integrity of the regions where the sensors are placed, a touchpoint size of only 0.5mm was used, with some manual adjustments to these touchpoints. All adjustments and slicing were performed using the official Formlabs slicer: Preform.

After printing, the support structure was removed using flush cutters. The parts were then cleaned using isopropyl alcohol and finally cured under UV light for about 10 minutes per side.

2.1.2 Top layer

To finish the sensor, a protective top layer is applied. The protective layer is made by casting silicone into a shape.

2x2 molding experiments

Just like the dome structure, the molding process was first tested on the 2x2 prototype. To facilitate molding, multiple versions of the molds shown in Figure 2.5 were created, with thicknesses (distance between the top of the dome and the overall top) of 1mm, 2mm, and 3mm. Additionally, two materials with different hardness levels on the Shore A scale were used: one with a hardness of 5A and another with a hardness of 50A. The silicone comes from Silicones and More[1].

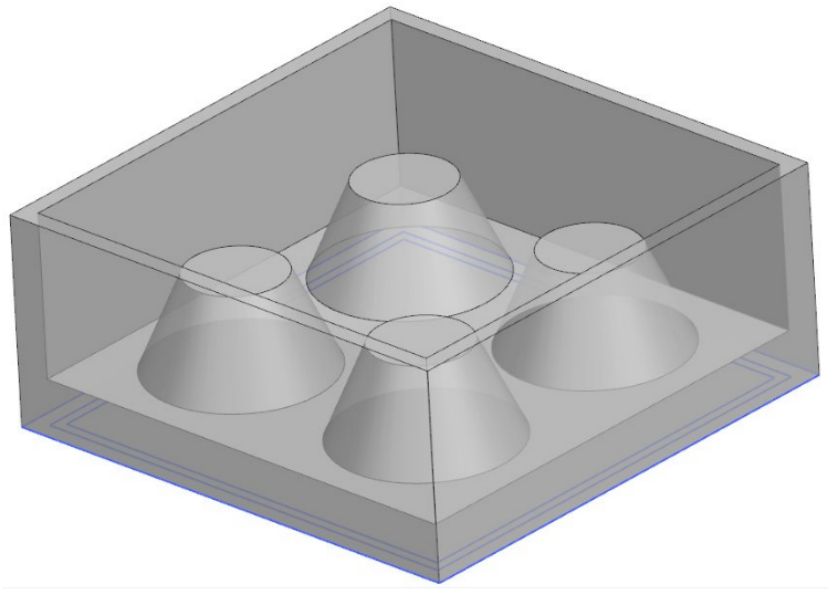


Figure 2.5: An example of the molds used to produce the 2x2 top layers

Regarding thickness, the 1mm version was deemed too thin, as these often tore during testing. Both the 2mm and 3mm versions worked, but to maintain the tactile feedback of the taxels, a thickness of 2mm was chosen. As for the material type, the softer material was found to be too soft under the prescribed application load of 10N, as it would deform too much and not transfer the force effectively to the taxels.

During testing, it was also found that the original design from Figure 2.5 was not effective in terms of force transfer. Instead of uniformly moving the taxels, the cast rests on the bottom of the dome structures, between the taxels. This hindered the transfer of the applied force to the taxels, as some (if not most) of the force dissipated through the dome structure to the PCB instead of to the sensors.

[1] https://www.siliconesandmore.com/en/silicone-addition-clear-50-hard.html

To counter this, the mold was modified. Instead of encompassing the complete taxels, only the top part supports the mold. These small cup shapes are enough to keep the structure intact during force application, while ensuring the applied force actually moves the magnet instead of dissipating through the PCB. A similar top layer was also used for the 4x8 version, which can be seen in Figure 2.6.

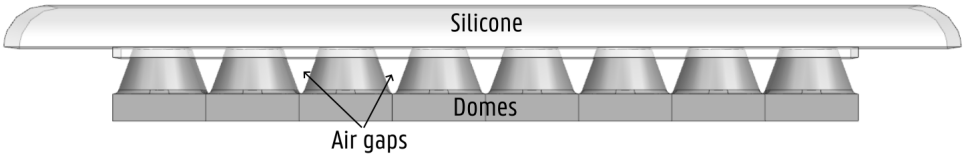


Figure 2.6: The second version of the top layer cast, now containing air gaps between the domes to make sure the applied force is transferred correctly to the sensors beneath the domes.

2.1.3 Fingertip casing

To utilize this finger design, a casing was required. An overview of the casing can be seen in Figure 2.1. The base is a modified fingertip from the Robotiq 2F-85 gripper, which is commonly used in conjunction with the Universal Robots for calibrating the sensor (see Section 3.2). Conveniently, the 4x8 array closely matches the dimensions of a Robotiq 2F-85 fingertip. The main function of the base is to hold everything together. Positioned on top of the dome structure is the top casing, which is responsible for securing the silicone-molded top layer. The fingertip casing was 3D-printed with PLA using a Prusa MK3 Fused Deposition Modeling (FDM) printer.

Fingertip base

The fingertip base is shaped such that the PCB neatly fits. A few cutouts can be seen in Figure 2.7 for some protruding electrical components. The wires going to the PCB come out of the back.

Top casing

The primary function of the top casing is to secure the silicone-molded top layer. As shown in Figure 2.8, the top casing features L-shaped hooks on all sides. During the silicone molding process, the silicone flows into these hooks, which helps to keep the silicone securely in place. The domes protrude from the top side of the top casing. Aside from securing the silicone top layer, it also contains cylinders to keep the PCB in place. However, the bottom of the finger is also required to prevent the PCB from falling out.

Molding process

To facilitate the molding process, two extra parts were created: a top mold cover and dummy taxels (Figure **??**). First the finger has to be constructed including the base and a dummy (non-soldered) PCB. On the PCB the dummy

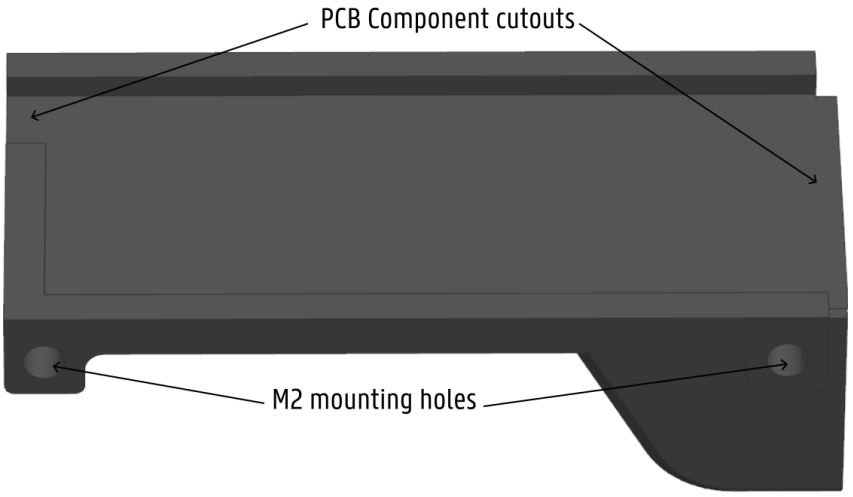


Figure 2.7: The base of the 4x8 finger prototype

taxels are placed instead of the actual domes. The dummy taxels are a 3D-printed version of the taxels but cutoff such that the silicone rests on the tops of the taxels instead of completely encompassing them, just like in the 2x2 prototype. After placing and securing the top case, the mold cover is placed over everything. The mold cover is responsible for constrocing the walls and in this design the roundings of the top layer. In the opening of the mold cover the silicone can be poured as usual. It is important to place the finger level, so that the same thickness is achieved everywhere. The top layer has a thickness of 2mm and the harder silicone with a hardness of 50 on the Shore A scale was used.

When the silicone has cured, the fingertip can be dissasembled again. The result can be seen in Figure 2.10. Notice how the sides of the silicone look quite rough. This is because the used 3D-printed has a minimal layer height. On circular features such as on the sides of the mold, this layer height is quite noticeable.

2.2 PCB design

2.2.1 Taxel sensor: Melexis MLX90393

The Melexis MLX90393 sensors are used on the PCB to detect the local magnetic field. The sensor is a highly versatile magnetic field sensor designed for a variety of applications requiring precise magnetic field measurements. This sensor utilizes proprietary Triaxis® technology to deliver accurate and reliable 3D magnetometer data, making it suitable for industrial, automotive, and consumer electronics applications. The MLX90393 sensor is characterized by its high sensitivity and configurable measurement ranges. The sensor is available in small QFN-16 packages, making it ideal for integration. It also has integrated temperature compensation, which is enabled.

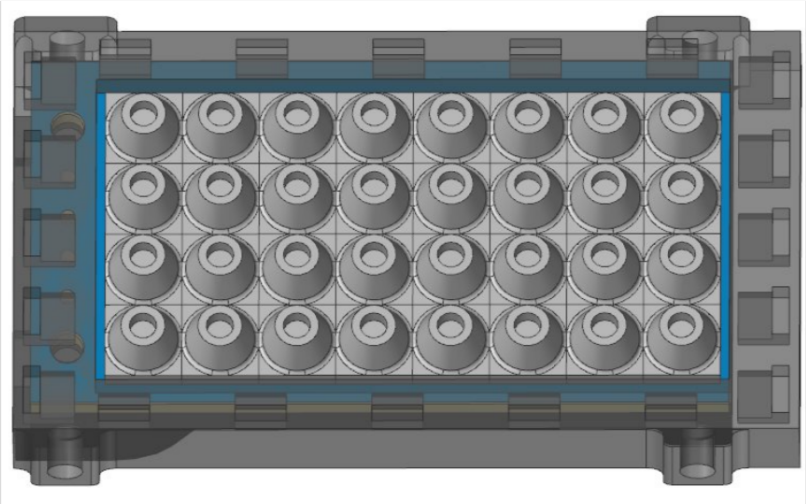


Figure 2.8: The top casing of the fingertip. Notice how L-shaped hooks were added to secure the silicone top layer in place.

The MLX90393 operates based on the Hall effect, where a voltage difference is produced across an electrical conductor through which a current is flowing when it is exposed to a magnetic field perpendicular to the current. The sensor integrates multiple Hall plates to measure the magnetic field components along three perpendicular axes. The digital output is derived from the Hall voltage after amplification and analog-to-digital conversion. The sensor also provides a temperature reading and internal temperature compensation. The chip can output these values over I2C or SPI in both burst mode or single measurement mode.

In burst mode, the internal ASIC (application-specific integrated circuit) of the sensor will operate at a programmable data rate. In single measurement mode, the bus master must ask for data using the corresponding communication protocol (I2C or SPI), which will wake up the ASIC to make a single conversion.

The time between waking up the ASIC to actually being able to read a single conversion is called the conversion time. This conversion time depends on a number of factors, such as the OSR (oversampling rate) and the digital filter setting. During the conversion time, the ASIC is busy measuring the magnetic field and applying the internal filters. Only after the conversion time the bus master may request the data from the internal registers.

The choice of the oversampling rate (OSR) and the digital filter settings are not trivial. Indeed, lowering the OSR or digital filtering settings might reduce the conversion time but increases the noise, according to the datasheet [1]. After some testing, the OSR was set to 2 and digital filtering was set to 1, resulting in a conversion time of 3 ms.

Aside from OSR and digital filtering, there are also the gain and resolution settings. The gain and resolution settings were set to 7 and 0, respectively, configuring the sensor to be the most sensitive it can be. The sensitivity per Least Significant Bit (LSB) is $0.150\mu T/$LSB for the X and Y directions and $0.242\mu T/$LSB for the Z axis. The

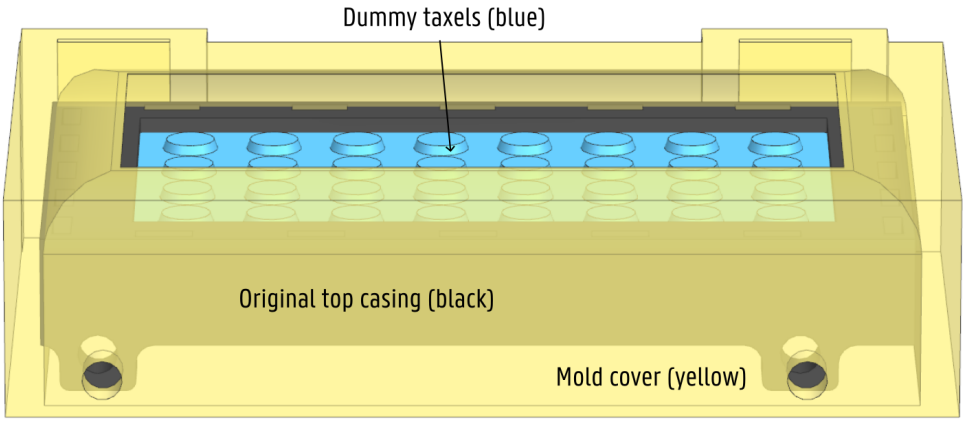


Figure 2.9: The molding setup. The silicone is poured into the top opening. Note how dummy taxels are used to create the shallow cup shaped openings in the final cast.

local sensor frame and sensitive region of the QFN-16 can be seen in Figure 2.11

2.2.2 Schematic

The PCB was designed in EasyEDA. EasyEDA is a free online tool to create PCB's.

Sensor array lines

The bulk of the PCB consists of Melexis sensors used to measure the local magnetic fields. These sensors can be configured to up to four I2C addresses by connecting the two address pins, A0 and A1, to either GND or VCC. The upscaled sensor array is 4x8, meaning the sensors are grouped into eight lines of four sensors each. The sub-schematic of a single line is shown in Figure 2.12. Each line has its own I2C bus, terminated by pull-up resistors as required by the I2C specification. The sensors were configured to each have a different I2C address, ranging from 0x0C to 0x0F (left to right).

Multiplexer

The eight I2C buses from all lines are wired into the I2C multiplexer as shown in Figure **??**. The multiplexer has a ninth main I2C bus (SDA, SCL), which is terminated with pull-up resistors and then connected to the header. The multiplexer can accommodate up to 8 addresses on the main I2C bus by using the addressing pins A0, A1, and A2. Two of these pins are hardwired to VCC, while the A2 pin can be pulled up or down depending on the configuration pad used.

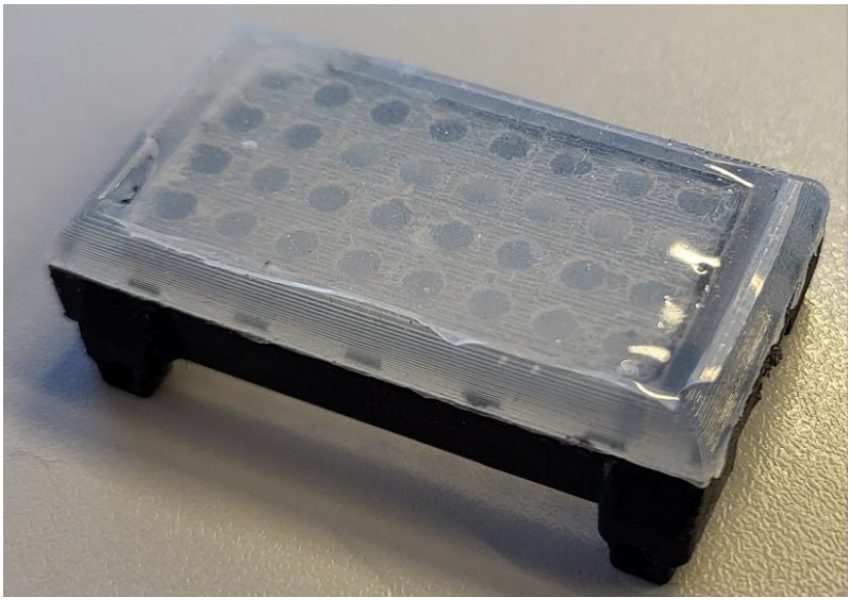


Figure 2.10: The result of the molding process. Notice the line pattern on the sides, originating from the 3D-printed molding cover.

2.2.3 Layout

Efforts were made to make this PCB as compact as possible, achieving a final pitch between the taxels of 4.5mm, with a single trace between two taxels. While attempts were made to stick with a 2-layer PCB, it was deemed too difficult with the constraint of minimal pitch, so a 4-layer PCB was used instead.

The top layer is populated with the 32 sensors, laid out in a 4x8 grid. The I2C lines connecting the lines of taxels are routed through vias to one of the middle layers. The bottom layer holds the multiplexer, resistors, and decoupling capacitors, thus responsible for the termination of the eight I2C lines and their routing to the multiplexer. The second middle layer serves as a ground plane.

Special care was taken to ensure the layout is highly symmetrical and modular, allowing for alternative versions to be easily made without changing the logic or general design. In Figure 2.14 below, the different layers (excluding the ground plane) are shown. Finally, the PCB was designed with two mounting holes and some pads used for soldering wires.

Alternative versions

Using a single line of the schematic, some alternative versions were created, depicted in Figure 2.15. Specifically, a 4x1 and a 2x2 taxel version, both with a 4.5mm pitch. Because these versions use only a single line of four sensors, it was possible to use a 2-layered PCB. Alongside these two single-line versions, a breakout board was created for the multiplexer.

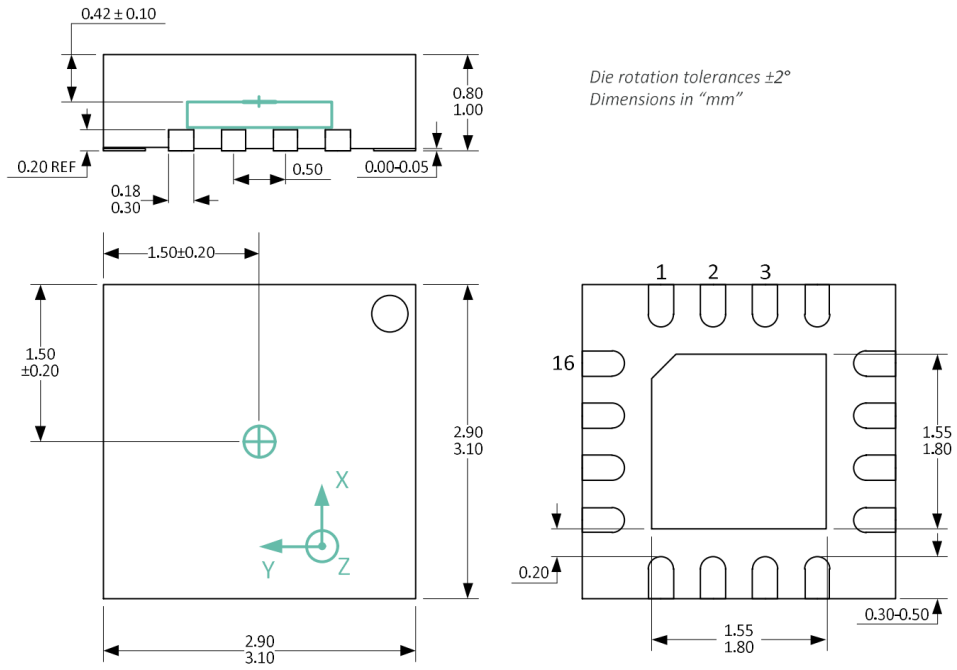


Figure 2.11: The sensitive region of the QFN-16 package of the MLX90393, referenced from the datasheet [1].

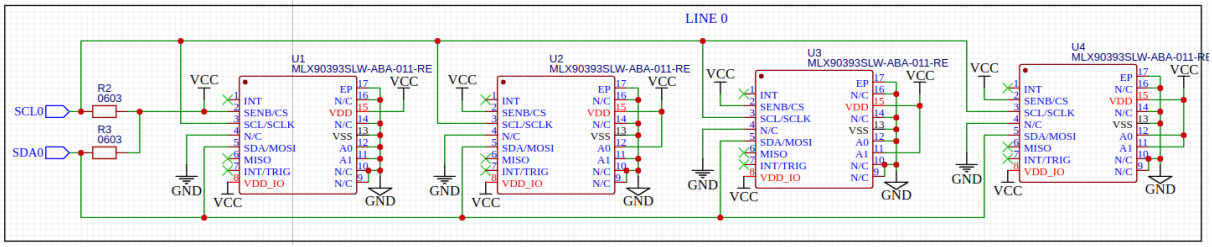


Figure 2.12: A single sensor line on the 4x8 PCB. Every line has their own I2C bus. The sensors are also all configured with a different address from 0x0C through 0x0F.

These alternative versions can be used in conjunction with a more advanced mechanical design to create sophisticated three-dimensional sensor arrays. A potential issue that might arise is the wiring from the smaller 4x1 and 2x2 versions to the multiplexer breakout board. One method to address this would be to design a flexible PCB that could wrap around a more complex mechanical design. The downside is that such a design would be highly application-specific. A more modular approach could involve using a flex-rigid PCB, where flexible parts (for the wiring) are combined with rigid regions containing the sensors. Alternatively, flat cables could be used instead of standard copper wires. Because this thesis focuses more on the behavior of this sensor technology,

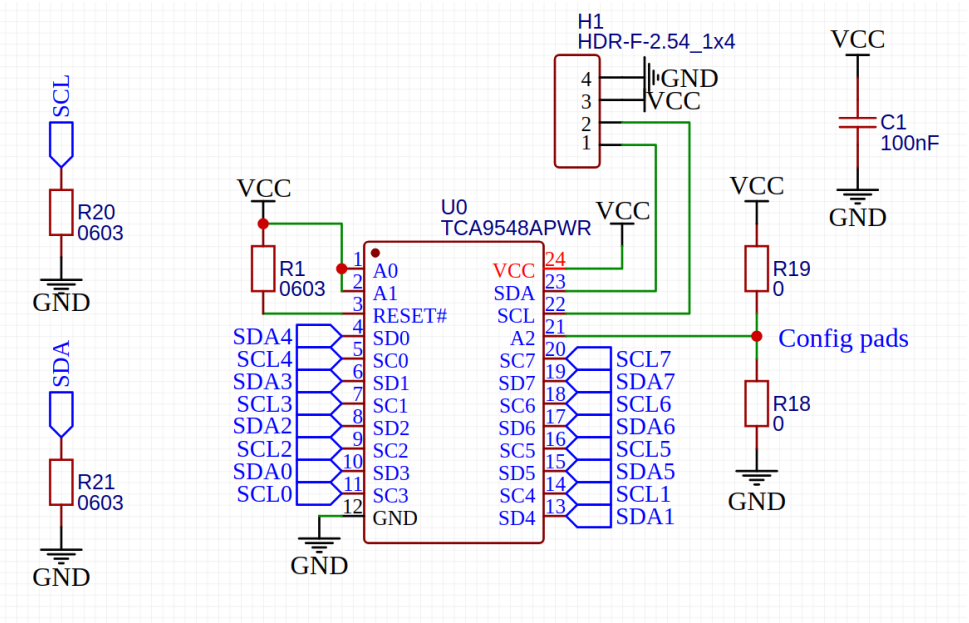


Figure 2.13: The I2C multiplexer taking all the I2C busses from the sensor lines and multiplexing them into the main I2C line SDA,SCL. The schematic has two config pads, which allows one to select one of two possible I2C addresses for the multiplexer.

this application-specific issue was left out of the scope of this dissertation.

2.3 Readout

This section talks about the communication between the different aspects of the system on a higher level.

2.3.1 Taxel communication

As described in the previous section, the MLX90393 supports both I2C and SPI. Here an overview will be given of both technologies and why in the end I2C was chosen.

SPI

SPI, or Serial Peripheral Interface, is a synchronous serial communication protocol used to transfer data between a microcontroller and peripheral devices. It operates in full-duplex mode, meaning data can be sent and received simultaneously. SPI uses four main signals:

- MOSI (Master Out Slave In): Line for sending data from the master to the slave.
- MISO (Master In Slave Out): Line for sending data from the slave to the master.

- SCK (Serial Clock): Clock signal generated by the master to synchronize data transfer.
- SS (Slave Select): Line used to select a specific slave device.

SPI is known for its high data rates and simplicity, making it ideal for high-speed communication with sensors, memory devices, and other peripherals. Despite the advantage of higher data rates, it has a significant drawback when scaling to larger sensor arrays: the requirement for a separate slave select (SS) pin for each slave device. This increases the complexity of the microcontroller's pin management and can limit the number of devices that can be connected.

I2C

In digital electronics, I2C (Inter-Integrated Circuit) is a communication protocol used for connecting integrated circuits, sensors, and peripheral devices, typically located on the same motherboard. It utilizes a two-wire serial interface consisting of a clock line (SCL) and a bidirectional data line (SDA). The bus supports multiple masters and uses open-drain outputs, meaning devices can only pull the bus lines to ground. When no device is pulling the line low, pull-up resistors pull the bus to a high level. This design prevents short circuits if two devices attempt to communicate simultaneously.

Communication on the I2C bus generally involves writing to and reading from memory registers. A communication sequence starts with a START condition, initiated by pulling SDA low while SCL is high. The master then sends a 7-bit slave address followed by a read/write (R/W) bit. A low R/W bit indicates a write operation, while a high bit indicates a read operation. The slave device acknowledges the address by pulling the bus low during the ACK bit cycle.

One advantage of I2C is its addressed 2-wire protocol, allowing multiple devices to share the same bus. Each slave responds to a specific address. In this thesis, the MLX90393 sensors used support up to four different hardware-configurable I2C addresses, determined by connecting certain pins to ground or VCC. The addresses used are 0x0C, 0x0D, 0x0E, and 0x0F.

The bus speed was set to Fast I2C, or 400kHz.

I2C multiplexing

Given that the MLX90393 sensor can only be configured with one of four I2C addresses, an I2C multiplexer is necessary when an array of more than four sensors is required on a single I2C bus. An I2C multiplexer, such as the TCA9548A from Texas Instruments, allows multiple I2C buses to be controlled by a single master bus. The TCA9548A has one master I2C bus and eight independent slave I2C buses.

The multiplexer itself has an I2C address, configurable through its A0, A1, and A2 pins, allowing for up to eight unique addresses. To select which slave bus is connected to the master, a command byte is sent to the multiplexer.

Each bit of this byte corresponds to a slave channel, and setting a bit to 1 enables the corresponding channel. For instance, to connect to the fourth child channel, one would write the byte “00001000” to the multiplexer's address.

Using the TCA9548A IC simplifies the design and scalability of the sensor array, enabling the integration of more than four sensors on a single master I2C bus while maintaining unique addresses for each sensor.

2.3.2 Microcontroller

The sensor array can be read by any microcontroller with an I2C driver. Three microcontrollers were utilized during this dissertation: the Arduino Nano 33 IoT, the Arduino Nano Every and the Raspberry Pi Pico W. The Arduino Nano 33 IoT features a single-core Arm® Cortex-M0 32-bit SAMD21 processor and a built-in NINA-10 wireless module from U-blox. The Arduino Nano Every, on the other hand, has a simpler architecture, featuring a single ATMega4809 processor and no built-in wireless capabilities. The third microcontroller (also one of my personal favourites) is the Raspberry Pi Pico W. It has an RP2040, which is a custom designed 32-bit dual ARM Cortex-M0+. The W version of the Raspberry Pi Pico also has an Infineon CYW43439 wireless chip, supporting Wi-Fi and Bluetooth 5.2. The Arduino boards were used for initial experimentation and data gathering for the calibration of the sensor. The two-stage force model was implemented on the Raspberry Pi Pico W because it has better performance. In the results section a small performance analysis will be made. Cost-wise, they are all fairly affordable, with the Raspberry Pi spanning the crown with at the time of writing being priced at €7.95 or €3.95 without wireless capabilities [2].

Even though these microcontrollers have vastly different architectures, they were all programmed using the Arduino framework in PlatformIO [3]. Thanks to the open-source community, this framework allows you to write cross-platform C/C++ code. Other frameworks exist such as FreeRTOS, but the Arduino framework suffices for this project.

However, one still has to be careful. For example, the base library in the Arduino framework used to read out I2C busses is the Wire library[4]. It provides a generic cross-platform interface to the programmer, while using a platform-specific implementation. While experimenting with the Arduino Nano 33 IoT, which uses mBed OS as the underlying framework, I found that the implementation of the Wire library was different. When using the Wire library to scan an I2C bus for slaves, the master sends a valid I2C address, followed by the R/W bit and waits for the bus to be asserted (pulled low) by the slave. If this happens, the master knows there is a slave responding to said address. Most Wire implementations set the R/W bit in this case to “write”. This was not the case for the Arduino Nano 33 IoT, which runs on mBed OS. The implementation of that Wire library sends a “read” command instead. And as it turns out, the MLX90393 will only acknowledge its address when the address is followed by

[2] https://www.kiwi-electronics.com/en/search?search=pico%20w

[3] https://docs.platformio.org/en/stable/frameworks/arduino.html

[4] https://www.arduino.cc/reference/en/language/functions/communication/wire/

the R/W bit being set to “write”. It is not clear if mBed OS has wrongly implemented the Wire library, or if the MLX90393 doesn’t completely follow the I2C specification. In the end, this might be an issue for other platforms as well.

To make sure this issue doesn’t persist, a customized version of the Adafruit MLX90393 and the Adafruit I2CDevice libraries were used as middleware between Wire and the scheduling procedure proposed in this section. The PlatformIO project including those customized versions can be found in the GitHub repository[5].

To minimize wiring, the wireless options were initially considered ideal due to its capability to send measurements wirelessly. However, as discussed in the results section, the feasibility of using a wireless architecture is debatable due to the extra overhead introduced by wireless communication protocols.

2.3.3 Readout and scheduling

Overview

To optimize the readout performance of the sensor array, a readout schedule is proposed in Figure 2.16. As noted before, the sensors are configured in single read mode and the bus master (the microcontroller) has to explicitely send a command over I2C to start a measurements. Then the sensor is busy for the so-called conversion time, after which the bus master can read the measurement registers. The schedule makes use of this conversion time by communicating to the computer during this period.

The schedule looks like this:

- The bus master sends a “start measurement” to all sensors in sequence. Every four taxels it also sends an I2C command to the I2C multiplexer to target a different row of the sensor array.
- While waiting on the conversion times of the sensors, the bus master sends the measurements of the previous cycle back to the computer over serial.
- Finally, the bus master reads out all the measurement registers, in the same order

Note how a more optimal approach would be to start a new measurement immediately after a measurement register has been read. Later it will be shown that the bottleneck in the design is the serial communication. Therefore, this optimization would not improve the overall rate for larger arrays. Also, starting the measurements sequentially as quickly as possible after each other makes sure the actual measurement points are as close as possible together in time.

The implementation can be seen in the provided GitHub repository [6].

[5] https://github.com/LowiekVDS/masters-thesis
[6] https://github.com/LowiekVDS/masters-thesis

Serial communication

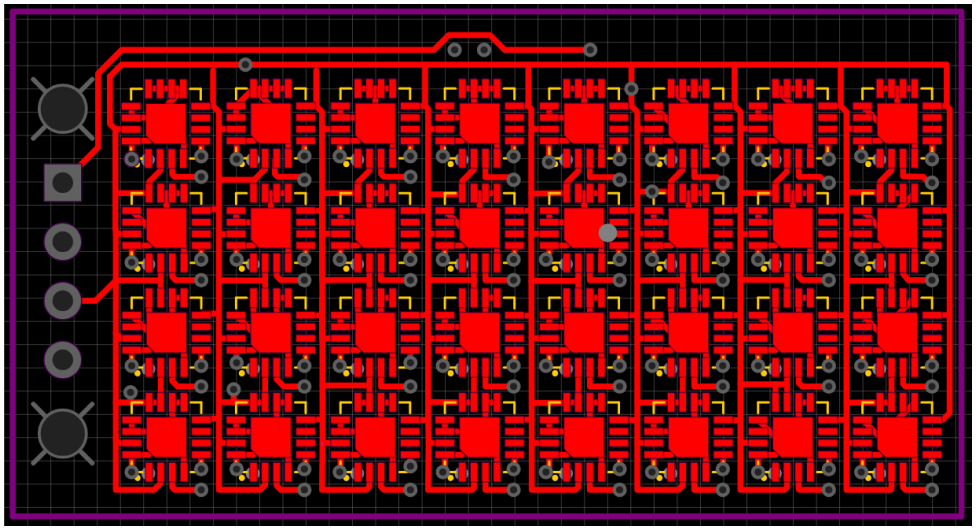
The microcontroller starts by sending a start byte, 0xAA. Following this start byte the microcontroller sends the raw magnetic measurements. The data is send per taxel, in the same order the measurements were started. Per taxel the microcontroller sends 2 bytes per axis, so 6 bytes in total, in the order X, Y, Z. So in the case of the 4x8 sensor array, one packet of measurements contains 193 bytes.

Communication scalability

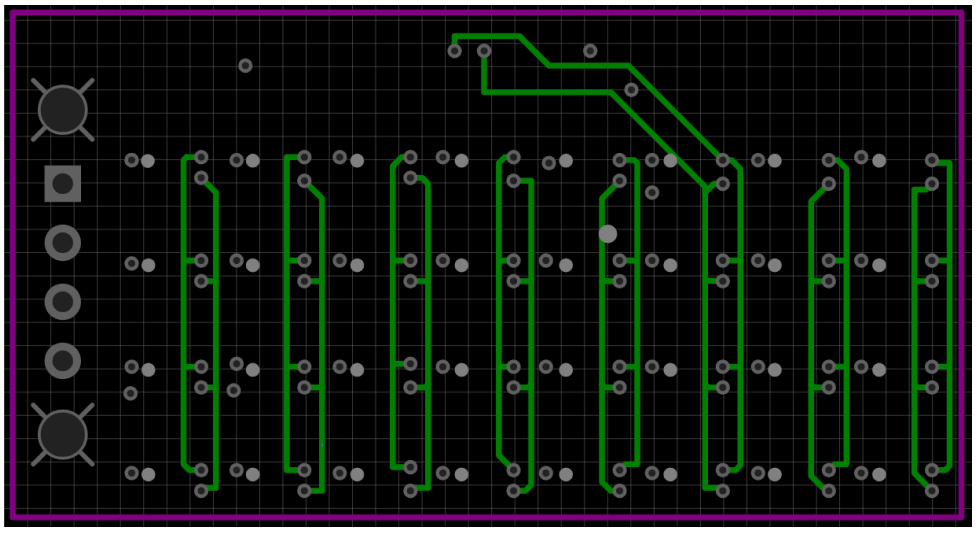
To verify if I2C would still support a high enough throughput required for the sensor array, a model was constructed to calculate the maximal achievable sensor throughput, given some communication parameters. The parameters are:

- I2C speed
- Sensor parameters to calculate the conversion time
- Serial speed
- Number of sensors
- Processing overhead

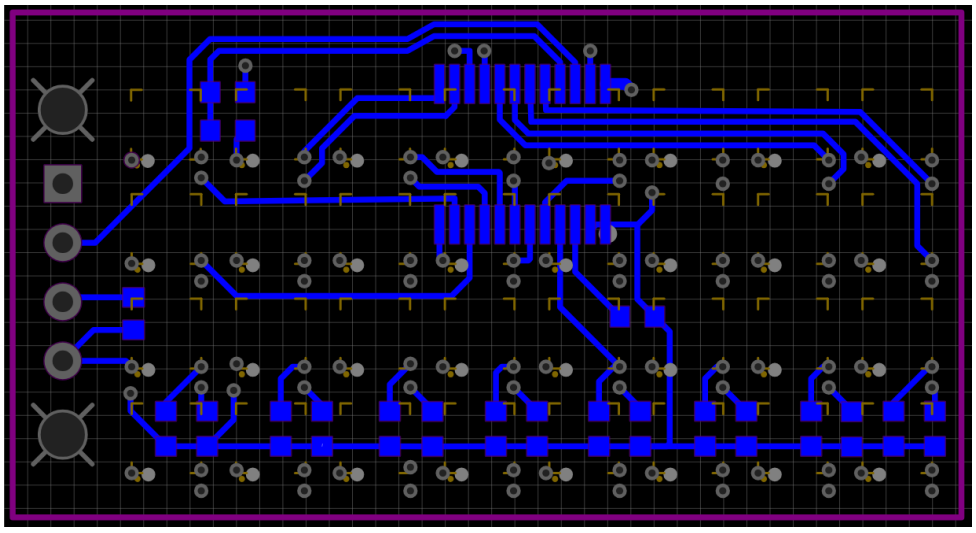
From this analysis became clear that even with processing overhead, the main bottleneck is the serial communication. The results section will go into more depth into the performance of the different microcontrollers, but this analysis shows that adding extra communication overhead by doing it wirelessly is probably not a good idea. Unless a microcontroller is used with multiple cores or a co-processor that can be used to offload the communication side of the software. A disadvantage is the much more complex software architecture. This is also one of the main reasons why the Raspberry Pi Pico W was chosen for the final integration of the sensor. The Pico W is the only microcontroller from the three discussed that can have a serial baud rate higher than 115200 bits per second. In the results chapter it will be shown that this (aside from better single-thread performance) has a large effect on the communication performance.



(a) The top layer of the PCB, containing the sensor array.



(b) The middle layer of the PCB, containing the 8 I2C busses connecting all the sensors together, four per bus.



(c) The bottom layer of the PCB, containing the multiplexer and passive components such as the resistors used for addressing and I2C termination.

Figure 2.14: Three of the four layers of the PCB. The fourth layer is a ground plane.

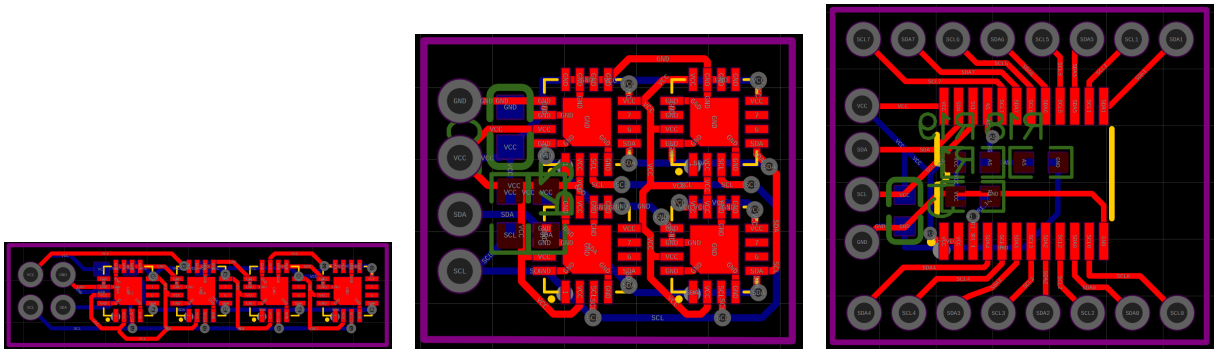


Figure 2.15: Alternative PCB's designs. From left to right: a 4x1 line, a 2x2 square and a multiplexer breakout board.

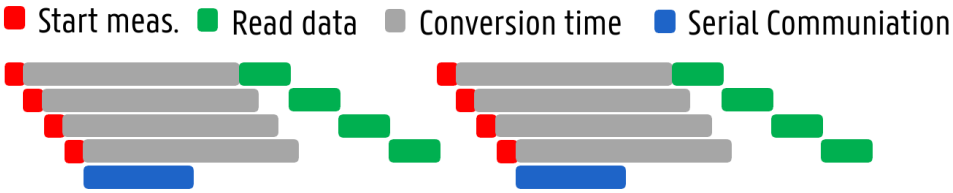


Figure 2.16: A schematical overview of the scheduling used to read out the sensor array and to send them to a computer.

3

Two-stage force model

In this chapter a two-stage force model will be proposed to convert the measurements of the sensor array to a three-dimensional force field. A general overview of this two-stage model is given in Figure 3.1. The first stage, to convert the measurements into an initial force $\overrightarrow{F_i}$, will be discussed in Section 3.4. The second stage, in which crosstalk will be tackled, is discussed in Section **??**. To substantiate the design choices made in the first stage, an analytical approach will be discussed in the next section.

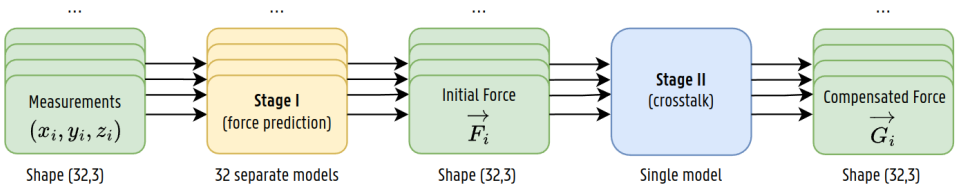


Figure 3.1: A general overview of the two-stage approach to convert the magnetic measurements to a force vector field.

3.1 Analytical background

To better understand what happens when a force is applied to a taxel dome and in what kind of measurements this would result, an analytical study was done based on the works in [19]. The magnetic field of an axially magnetized cylindrical magnet is axially symmetrical around the Z-axis. Therefore, the field can be represented by a vector $\overrightarrow{B}(\rho, z)$ at a point $P(\rho, z)$. When the assumption is made that the point P is far enough from the center of the magnet, the magnet can be seen as a dipole. Then these components are given by:

$$\begin{cases} B_z(\rho, z) = \frac{\mu_0 \mu}{4\pi} \frac{2z^2 - \rho^2}{r^5} \\ B_\rho(\rho, z) = \frac{\mu_0 \mu}{4\pi} \frac{3\rho z}{r^5} \end{cases} \tag{3.1}$$

Where μ_0 is the permeability of vacuum and μ is the magnetic moment of the cylindrical magnet. To ease notation, the magnetic measurements were normalized:

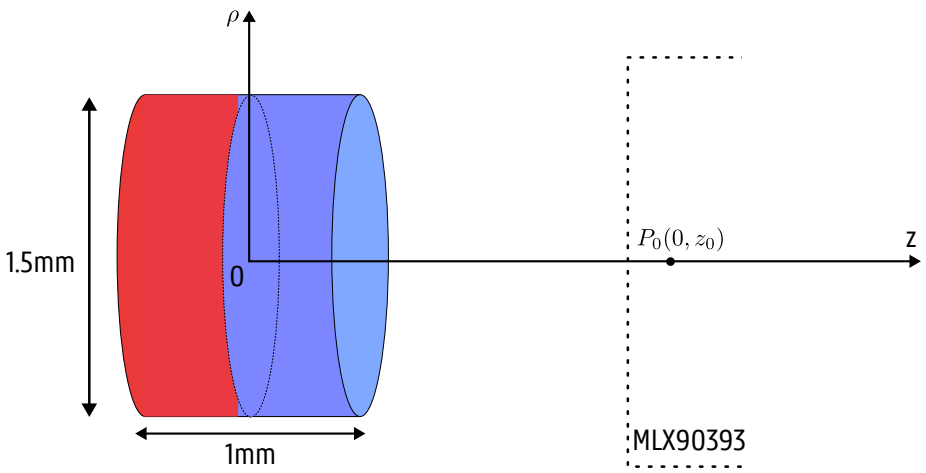


Figure 3.2: The coordinate frame used for the analytical discussion. It includes the magnet and the taxel sensor, an MLX90393. The point P_0 is the idle point when no force is applied to the taxel.

$$\begin{cases} B'_z(\rho, z) = \frac{2z^2 - \rho^2}{r^5} \\ B'_\rho(\rho, z) = \frac{3\rho z}{r^5} \end{cases} \tag{3.2}$$

Finally, $r = \sqrt{\rho^2 + z^2}$ is the distance from the center of the magnet, strategically placed at the origin.

When a taxel (magnet) is excited by applying a force to it, the magnet will shift position relative to the three-dimensional TriAxis Hall sensor below. We assume the taxel structure follows Hooke's law, meaning the material will act like an ideal spring. For example, when applying a normal force, the following holds under this assumption:

$$F_z = k_z \Delta z \tag{3.3}$$

Where F_z is the normal force applied to the taxel, the convention was chosen such that a negative force causes a negative displacement Δz, k_z is the spring constant of the taxel and Δz is the shift in the Z-direction. A similar equation holds for the planar force F_ρ:

$$F_\rho = k_\rho \Delta \rho \tag{3.4}$$

In what follows, we will look at what happens to the magnetic field when a force is applied to the taxel under specific circumstances.

Perfect alignment

In case the magnet is perfectly aligned in the dome structure, the idle point P_0 when no force is applied will lie on the symmetrical axis of the magnet, ie. $P_0 = (0, z_0)$. The application of a force will be decoupled into a normal force F_z and a planar force F_ρ. The normal force will cause the magnet to shift solely in the Z-direction, while the planar force will cause the magnet to solely shift in the ρ direction. Applying a force F_z will cause the magnet to shift to $P_1 = (0, z_0 + \Delta z) = (0, z_1)$. The change in the magnetic field can be written as:

$$\begin{cases} \Delta B_z(P_1) = B_z(0, z_0 + \Delta z) - B_z(0, z_0) = \frac{2}{(z_0+\Delta z)^3} - \frac{2}{z_0^3} \\ \Delta B_\rho(P_1) = B_\rho(0, z_0 + \Delta z) - B_\rho(0, z_0) = 0 \end{cases} \tag{3.5}$$

From which it becomes clear that the change in the magnetic field is inversely proportional to the cube of the distance, or assuming Hooke's law, the applied normal force. Similarly, applying a force F_ρ will cause the magnet to shift to $P_2 = (\Delta\rho, z_0)$. The change in the magnetic field can be written as:

$$\begin{cases} \Delta B_z(P_2) = B_z(\Delta\rho, z_0) - B_z(0, z_0) = \frac{2z_0^2 - \Delta\rho^2}{(\Delta\rho^2 + z_0^2)^{5/2}} - \frac{2}{z_0^3} \\ \Delta B_\rho(P_2) = B_\rho(\Delta\rho, z_0) - B_\rho(0, z_0) = \frac{3\Delta\rho z_0}{(\Delta\rho^2 + z_0^2)^{5/2}} \end{cases} \tag{3.6}$$

When we further assume that the movement of the magnet in the ρ direction is small compared to the distance z_0 between the center of the magnet and P_0, the change in the magnetic field at P_2 can be approximated as:

$$\begin{cases} \Delta B_z(P_2) \approx 0 \\ \Delta B_\rho(P_2) \approx \frac{3\Delta\rho}{z_0^4} \end{cases} \tag{3.7}$$

From which it becomes clear that the change in the magnetic field is linear to the applied planar force, assuming the planar motion caused by the planar force is small compared to the distance between the center of the magnet and P_0. When instead the movement in the ρ direction has a similar magnitude to the distance z_0, the change in the magnetic field at P_2 can be approximated as:

$$\begin{cases} \Delta B_z(P_2) \approx \frac{1}{2^{5/2}} * \frac{1}{\Delta\rho^3} \\ \Delta B_\rho(P_2) \approx \frac{3}{2^{5/2}} * \frac{1}{\Delta\rho^3} \end{cases} \tag{3.8}$$

Which shows that in this case the change in the magnetic field is inversely proportional to the cube of the distance, or assuming Hooke's law, the applied planar force.

Imperfect alignment

Manufacturing defects can cause the magnet to be misaligned in the dome structure. In this case, the idle point P_0 when no force is applied will no longer lie on the symmetrical axis of the magnet, ie. $P_0 = (\rho_0, z_0)$. To analyse the behaviour of the magnet when a force is applied, one can add a coordinate transformation from the magnet local frame to a global taxel frame (in which the forces are applied). Therefore, instead of using the cylindrical coordinates ρ and z, we will use the Cartesian coordinates x and y with $\rho^2 = x^2 + y^2$. The Cartesian description of the magnetic field produced by the magnet is:

$$\begin{cases} B_x(x,y,z) = \frac{3xz}{r^5} \\ B_y(x,y,z) = \frac{3yz}{r^5} \\ B_z(x,y,z) = \frac{2z^2 - x^2 - y^2}{r^5} \end{cases} \tag{3.9}$$

Where again $r = \sqrt{x^2 + y^2 + z^2}$ is the distance from the center of the magnet.

A rotation matrix and a translation vector can be defined to transform the coordinates from the magnet local frame to the global taxel frame. Note that the transformations are linear, which means that even though the magnet is misaligned, the magnetic field can after applying the transformation still be decomposed into a radial and longitudinal component like in the section before. The calculations and the general structure of the change in magnetic fields will thus remain the same, but the expressions will be more complex. Therefore, no further analysis will be done in this section.

Conclusion

While not all cases were rigorously analysed, a few important conclusions can be made. Those conclusions will aid in the choice for both the first Stage of the force prediction model and the preprocessing steps. The change in the magnetic field along the Z-direction is shown to be inversely proportional to the cube of the applied force in that direction. Furthermore, depending on the relative magnitude of the distance between the center of the magnet and the sensor z_0 and the relative planar movement $\Delta\rho$, the planar measurements (X and Y) might be also inversely proportional to the cube of the applied force in the planar direction or, when $\Delta\rho$ is rather small compared to z_0, a linear relationship will occur.

In any case, even when the magnetic field equations themselves are highly non-linear, this analysis is the reason why a polynomial regression model together with some smart transformations was chosen for the first Stage of the force prediction model. The fact that a misaligned magnet can be modelled as a (linear) change of coordinate

frames, shows that the X, Y and Z magnetic field measurements are actually going to be coupled. The knowledge that the measurements of the different axes are likely coupled with one another shows that the cross products of the polynomial model are actually valuable. In other words, model architectures where components are not coupled (such as only using the Z measurements to predict the normal force) will most likely perform worse than model architectures were all components are coupled.

3.2 Calibration methods

In order to be more robust against manufacturing defects a data-driven approach was used. A data-driven approach requires a proper calibration procedure. In this section the calibration method used will be described, as well as an improved method will be proposed.

3.2.1 The UR3e robot as calibration platform

For the calibration procedures an UR3e robot from Universal Robots was used as calibration platform. Initially, the collaborative robotic arm was chosen as a platform because it was readily available in the laboratory and it already had an integrated force sensor on the tool flange.

The force sensor (also referred to as the F/T-sensor) has a range of 30.0N, a precision of 2.0N and an accuracy of 2.5N. While on paper this might not look spectacular, in real-life it felt to perform better. That being said, much more accurate sensors do exists, which would help the calibration.

The advantage of the robotic arm is that you have precise control over what the probe does. Also, because a robotic arm is the application of this sensor, using the robotic arm during calibration opens up a whole range of possibilities, such as real-time (re)calibration before a grasping operation. This alternative method was tested and published as a paper, included as appendix A.

The disadvantage of using the robot is that compared to for example XY-carriages, a robotic arm will most likely be less stiff. This can influence the calibration results.

3.2.2 Manual Calibration

Initially, a manual procedure similar to [18, 20, 21] was developed and applied to the 2x2 version. In this procedure, the sensor is fixed to the tool flange, and a person manually excites the taxel to be calibrated while recording data from both the taxels and the force sensor. Some advantages of this method include:

- Randomness in the acquired data due to the human aspect of the calibration

- Calibration costs, as it only requires a rigidly mounted force sensor, instead of a platform that can precisely apply forces in preprogrammed directions, such as the UR3e.

However, it became apparent that despite the randomness in the procedure, it was challenging to obtain sufficient quality data in all directions, particularly in the horizontal X and Y directions. Additionally, the domes themselves are relatively fragile, so they are more prone to breaking during the calibration because of the random, mostly uncontrolled applied forces.

Calibration Setup

For the manual calibration, the tool flange of the robot was placed facing upwards. Then a mount was 3D-printed in standard Polyactic acid (PLA) to securely hold the sensor array in place. The result for the 2x2 prototype can be seen in Figure 3.3.

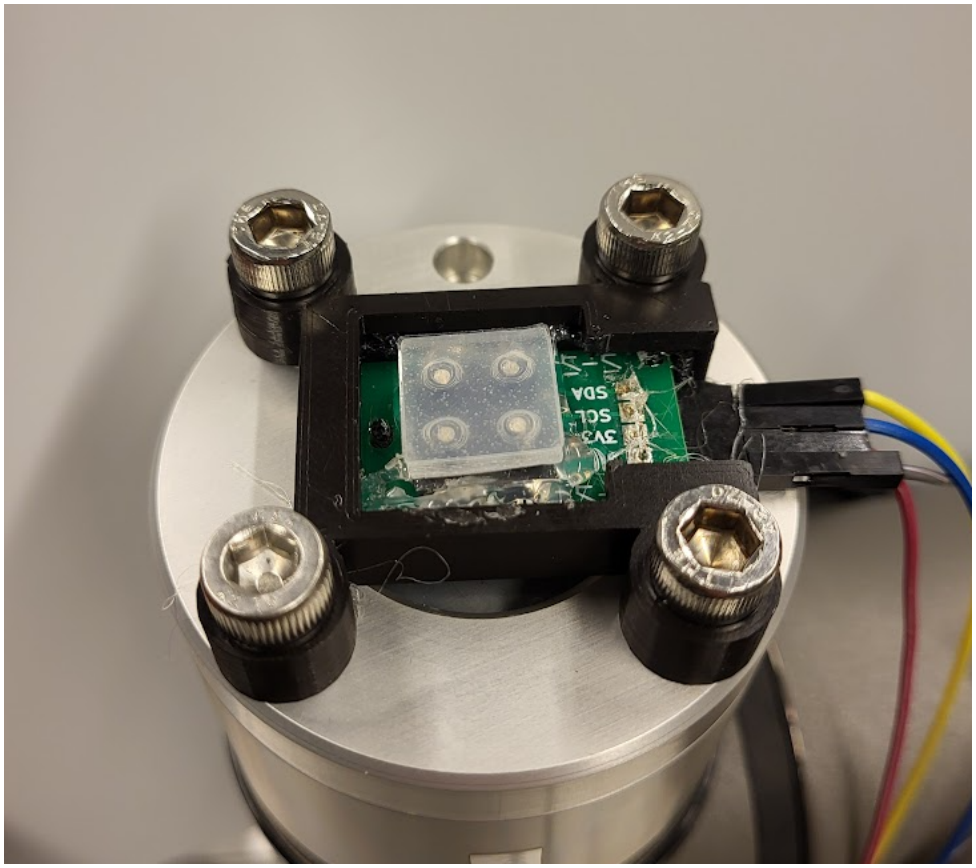


Figure 3.3: The 3D-printed fixture used to securely mount the sensor array to the tool flange of the UR3e robotic arm during a manual calibration.

To execute the calibration, a calibration probe was 3D-printed in PLA. This small handheld dome-shaped probe features a dome that neatly fits over a taxel. The person calibrating the sensor places this probe over the dome

of the taxel that is to be calibrated. During the recording, random motions should be executed. In principle, the more randomness, the better.

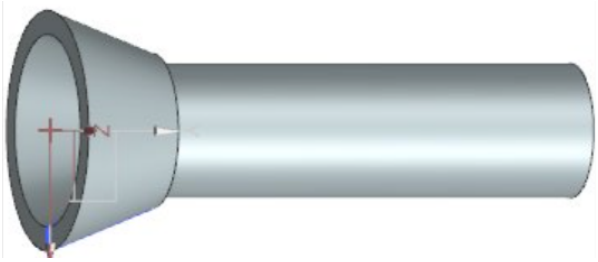


Figure 3.4: The 3D-printed probe used to apply force to the taxels during manual calibration.

3.2.3 Automatic Calibration

To better understand the behavior of the sensor and obtain predictable, higher quality data for calibration, an automatic calibration method was also developed and used on the upscaled 4x8 sensor. In this approach, the UR3e robotic arm was used as a platform to precisely apply preprogrammed calibration cycles to each individual taxel.

Calibration setup

Because now the robot applies forces to the taxels, the sensor is securely attached to the stationary table instead. A mount was 3D-printed that securely holds the sensor array fixed to the table. Also a probe tool was mounted to the tool flange of the robot. The probe tool is similar to the manual probe. The probe is dome-shaped to neatly fit over a taxel. The setup can be seen in Figure 3.5.

Homing Procedure

To automate the process as much as possible, a homing procedure was devised to find the taxel positions. This way, after the calibration procedure you have to roughly positioning the sensor array below the calibration probe, the robot will take care of the rest.

The collaborative UR3e robotic arm features a force mode in which it moves in a given direction until it detects resistance from contact. This mode is employed in various directions to locate the sensor array relative to the robot's base frame. Upon detecting contact, the robot stops, and the custom developed calibration software records current Tool Center Point (TCP).

The procedure begins with the robot moving downward to find the top of the sensor array. Subsequently, the robot quickly moves horizontally (in the positive X direction) and downward below the top of the sensor array. The robot then probes the sensor in the opposite direction (negative X), again finding the contact point using the

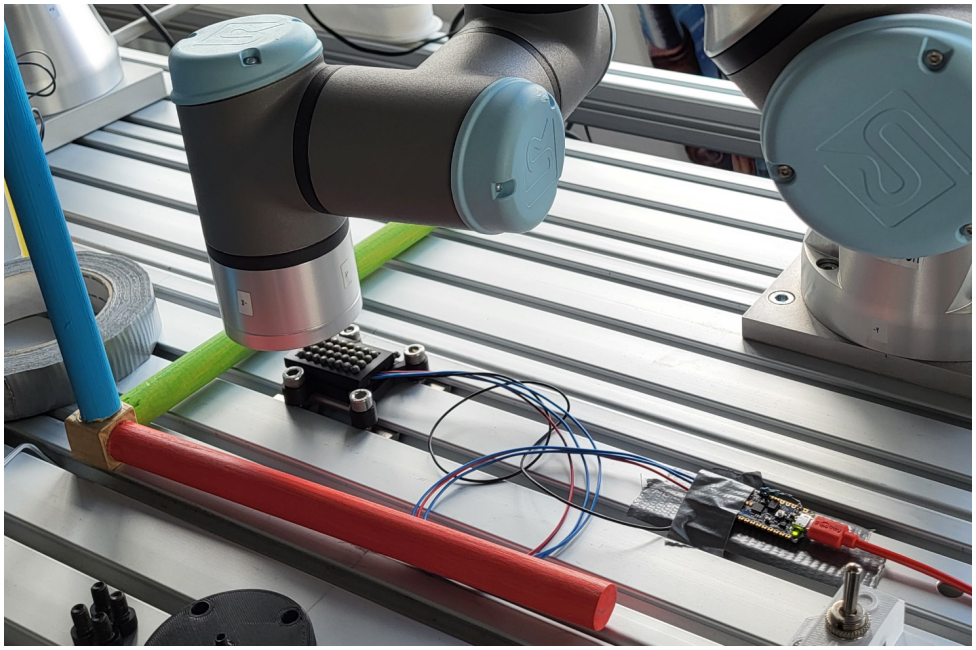


Figure 3.5: The experimental setup used to calibrate the sensor array with the automatic calibration procedure.

force mode of the robot. After the first contact point, the robot moves sideways along the Y-axis and probes again in the negative X direction. These two probing points correspond to the right black points in Figure 3.6. Then the robot moves all the way to the other side to find a contact point by probing in the positive X direction. This is the black point on the left side in Figure 3.6. A similar procedure is done for the Y-direction, but only once. This is the bottom probe point in Figure 3.6.

The probing points are then processed by an algorithm to determine the two-dimensional planar rectangle representing the rectangular outline of the sensor design. After compensating for probe thickness, knowledge of the mechanical dimensions of the sensor array is utilized to calculate the taxel positions. An example can be seen in Figure 3.6. These taxel positions are then used as reference points during calibration.

After locating the sensor array, the robot applies a predefined calibration force to each taxel. The main calibration procedure consists of four distinct stages, each with a different calibration pattern. First, the normal force was calibrated on its own, followed by the planar forces while a constant normal force was applied. Finally, a constant normal force was again applied while simultaneously applying planar forces in quadrature, causing the robot to make a planar circular motion. This was done at two frequencies: 0.5 Hz and 3 Hz. Between stages, the robot releases the taxel for a few seconds to allow the taxel to recover from hysteresis and to settle back to its idle position.

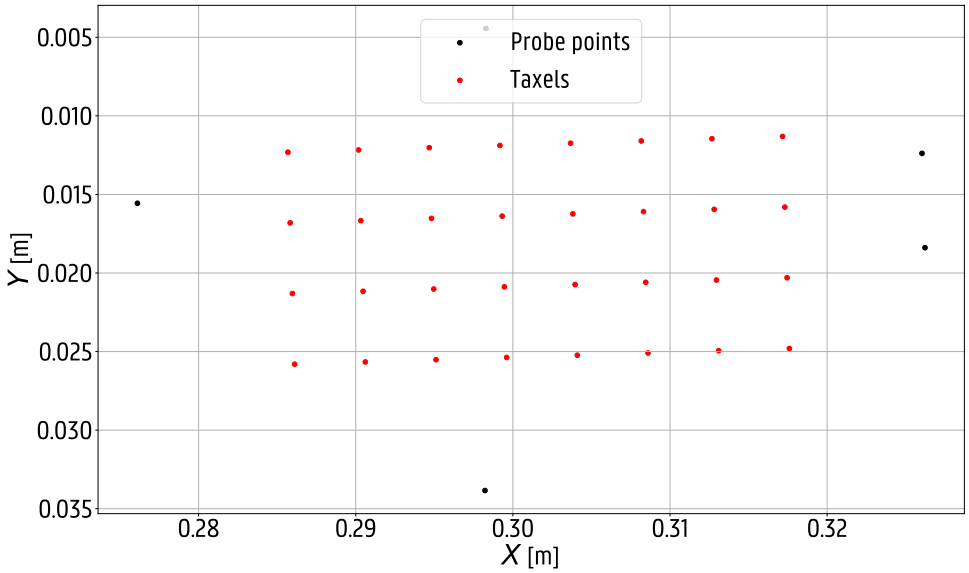


Figure 3.6: An example of the homing procedure on the 4x8 sensor array. The black points are the probing points discovered using the force mode of the robot. The red points are the calculated taxel positions based on these probe points, tool thickness and known mechanical sensor dimensions.

Descending sine wave

Inspired by work done in [22], the main calibration waveform used is a descending sine wave. A descending sine wave is a sine wave that decreases in amplitude over time, eventually reaching zero. The mathematical expression for such a wave is given by:

$$s(A, t_{\text{max}}, f, \theta, t) = A \cdot \frac{t}{t_{max}} \cdot \sin(2\pi f \cdot (t - t_{\text{max}}) - \frac{\pi}{2} + \theta) \tag{3.10}$$

Where A, t_{max}, f and θ are respectively the maximal amplitude, the maximal time of the curve, the frequency and the phase shift. An alternative shifted version is given by the following expression:

$$s_{\text{shifted}}(A, t_{\text{max}}, f, \theta, t) = A \cdot \frac{t}{2t_{\text{max}}} \cdot [\sin(2\pi f \cdot (t - t_{\text{max}}) - \frac{\pi}{2} + \theta) + 1] \tag{3.11}$$

An example of a shifted descending sine wave can be seen in Figure 3.7.

Note that t should be limited to the interval $[0, t_{\text{max}}]$ and t_{max} should be a whole number. This way, the sine wave will start at $t = 0$ and terminate at $t = t_{\text{max}}$.

An alternative to the descending sine wave is the ascending sine wave. The ascending sine wave is a sine wave that increases in amplitude over time, eventually reaching the maximal amplitude. The mathematical expressions can be found by first mirroring the time axis and then shifting the curve to the right by t_{max}:

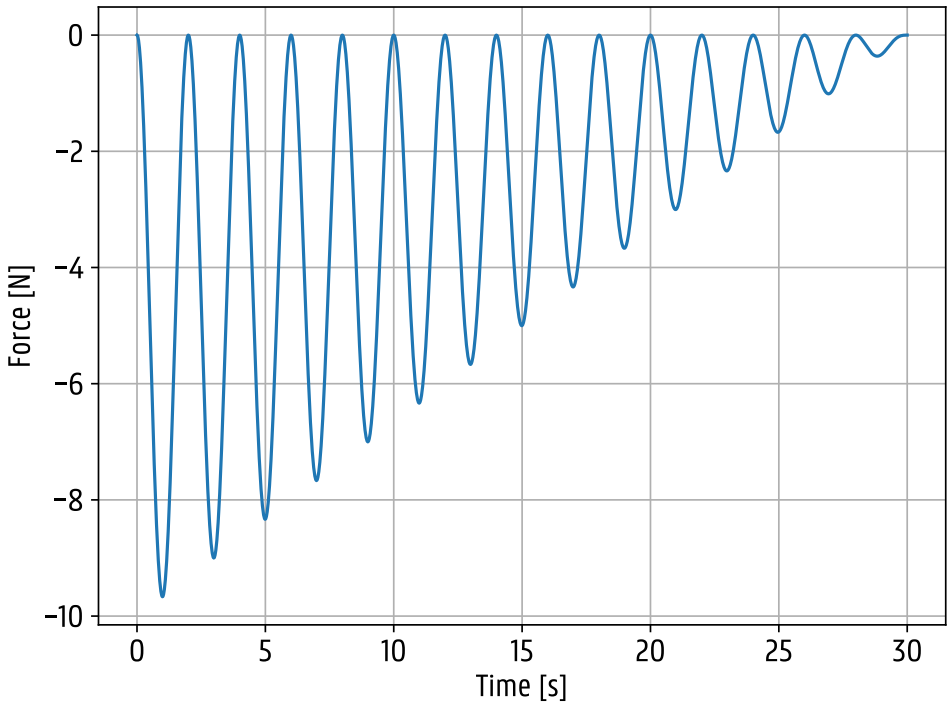


Figure 3.7: A shifted descending sine wave, used as a basis for the calibration procedure. $A = -10, t_{\text{max}} = 30$ and $f = 0.5Hz$

$$\begin{cases} s^{'}(A, t_{max}, f, \theta, t) = s(A, t_{max}, f, \theta, t_{max} - t) \\ s^{'}_{shifted}(A, t_{max}, f, \theta, t) = s_{shifted}(A, t_{max}, f, \theta, t_{max} - t) \end{cases} \tag{3.12}$$

For the calibration procedure the descending sine wave was chosen. By almost immediately applying a large force, the calibration platform (here the robotic arm) is able to quickly settle itself properly on the taxel, should (small) misalignments have occurred during the homing procedure.

Applying calibration forces to the taxel

For the application of calibration forces to the taxels, the UR3e collaborative robot was put into force mode. In this mode, the user can choose which of the six degrees of freedom should be fixed. On the free degrees of freedom, a target force can be set. The robot will then apply those target forces, resulting in a force-controlled movement. To prevent the robot from exceeding joint limits, a velocity limit can also be set.

In the context of automatic calibration, the rotational degrees of freedom are fixed, while a target force time series is set in the translational degrees of freedom. A global velocity limit of $0.1m/s$ on each translational axis was set. The target force time series is sampled (and applied) at 200Hz.

In order for the force mode to properly work the payload of the robot must be correctly calibrated. This can be done by following the instructions in the UR3e firmware.

Step I: normal force

For the calibration of the normal force, a shifted descending sine wave was used with $A = -10, t_{max} = 30$ and $f = 0.5$Hz as the Z component of the force. The X and Y force components were set to zero. The curve is visualized in Figure 3.8.

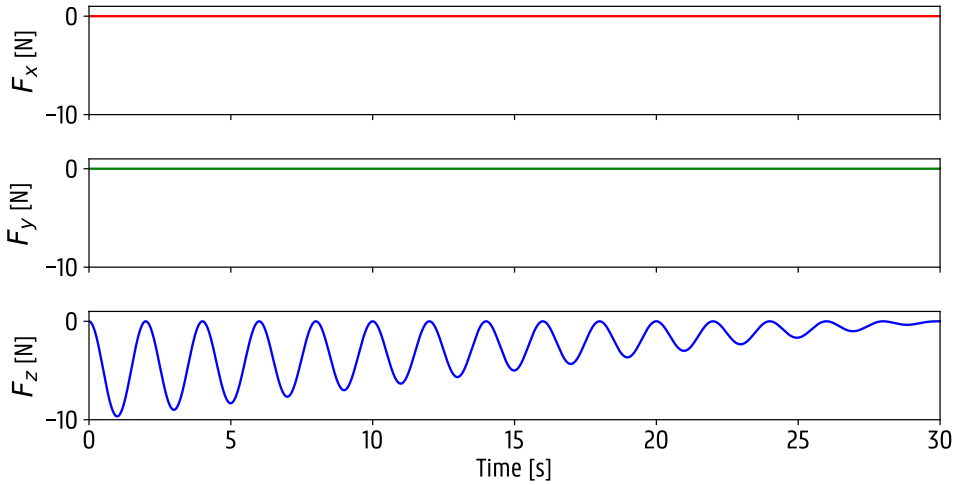


Figure 3.8: The first step of the calibration procedure

Steps II and III: planar forces

In these steps as seen in Figure 3.9 the normal force was set to a constant -5N. This to ensure the robot was firmly gripping the taxel. Otherwise the robot might slip off the taxel when applying the planar forces. Then a descending sine wave is applied to the X-component of the force, followed by the Y-component. Notice how in-between the different subprocedures the robot procedure waits to let the taxels return to their idle position. Also when first applying the normal force the robot waits to allow the taxel to settle.

Step IV: planar quadrature forces

As an additional test, a constant normal force of -5N was applied, while simultaneously applying two descending sine waves in quadrature, respectively to the X- and Y-component of the applied force as can be seen in Figure 3.10.

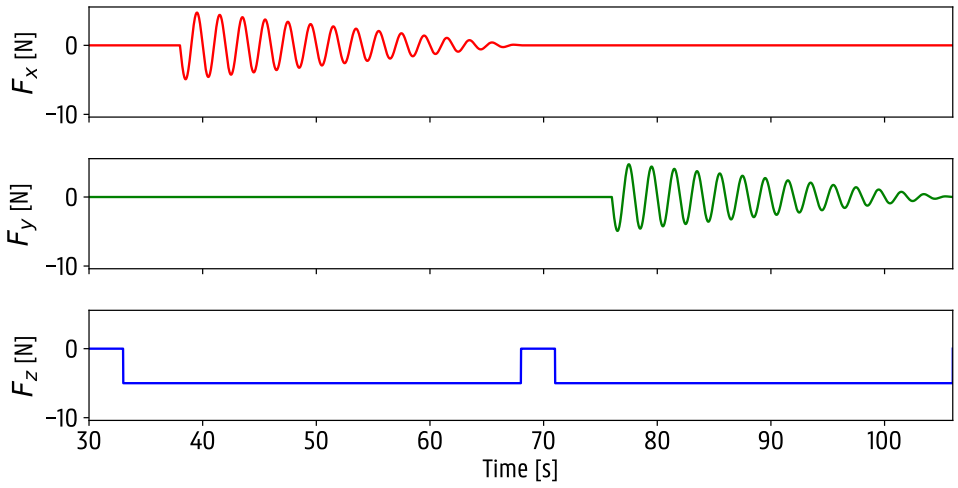


Figure 3.9: The second and third steps of the calibration procedure. A constant normal force, together with a descending sine wave sequentially applied to respectively the X and Y components of the force

Descending three-dimensional spiral

The approach described in the previous paragraphs was used to clearly separate the normal and planar forces. This way, if any calibration issues should occur, it would be easier to identify the cause. However, doing all components separately takes a rather long time. The prescribed procedure takes about 260 seconds per taxel, or 130 seconds per frequency setting per taxel. Scaling to a 4x8 array the whole calibration procedure would take about 2 hours and 19 minutes.

Also, the planar forces are only calibrated with a constant normal force of -5N. This is a rather arbitrary choice. Provided that the final model performs well without being too sensitive to the specifics of the sample points in the calibration data set, an approach where all force components are descending sine waves could be used. This way, the calibration procedure would be faster by a factor of about 3. An alternative procedure can be seen in Figure 3.11.

The alternative procedure first does a 15 second long descending sine wave on the normal force. This first calibration Stage is then followed by a second Stage where the planar forces are two descending sine waves in quadrature, creating a circular motion. The maximal force amplitude of the planar forces is set to 4N.

For the normal component, a descending sine wave was chosen with a slightly different frequency of 0.6Hz, removing periodicity in the spiral motion. To prevent the calibration tip from slipping off the taxel, the normal force was offsetted to always have an envelope that in absolute value is larger than the absolute value of the envelope of the planar forces. This was done with a scaling operation:

$$F_z'(t+18) = F_z(t) \cdot (\frac{0.4}{30} \cdot t + 0.6) \tag{3.13}$$

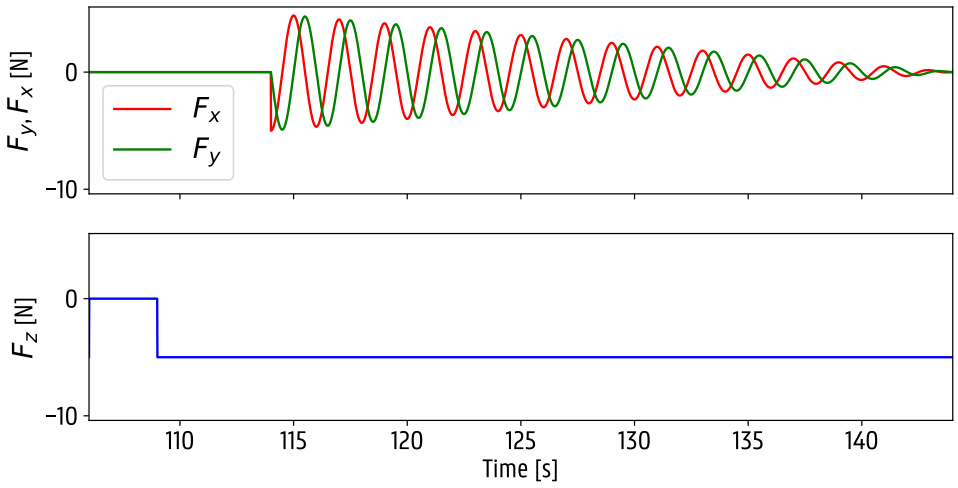


Figure 3.10: The final step of the calibration procedure. A constant normal force, together with two descending sine waves in quadrature applied to the X- and Y-component of the force. Together they form a planar circular motion.

Followed by a translation:

$$F_z''(t+18) = F_z'(t+18) + \frac{4}{30} \cdot t - 4 \tag{3.14}$$

The spiral procedure was also applied to the taxels of the 4x8 sensors. The differences between the two approaches were analysed in the results chapter.

A note about the 3Hz frequency

The calibration was done in multiple frequencies: 0.5Hz and 3Hz. But as shown in Figure 3.12 the higher frequency excitation causes odd effects to occur. On the figure the inverse of the Z component of the magnetic measurements of taxel 0, z_0 is shown. Below that, the applied force as reported by the F/T sensor and below that the Z-position as reported by the robotic arm. Notice how over time both the robot probe rises, which is also confirmed by the magnetic measurements, while the applied force supposedly stays ON as it should. It appears as this kind of excitation causes some bias to creep into the F/T sensor over time, causing the robot to gradually drift as well. My hypothesis is that likely due to the fact that the F/T sensor is not able to keep up with the high frequency excitation. Showing that a robotic arm, or at least the used F/T sensor, might not be the ideal platform to perform calibration procedures like this. As a result the 3Hz measurements are not useful for the creation of the models. Indeed, when Stage I is trained including this data, the performance metrics drastically drop.

That being said, the 3Hz measurements are used to estimate the latency difference between the F/T-sensor and the magnetic measurements. This is discussed further in Section 3.4.2.

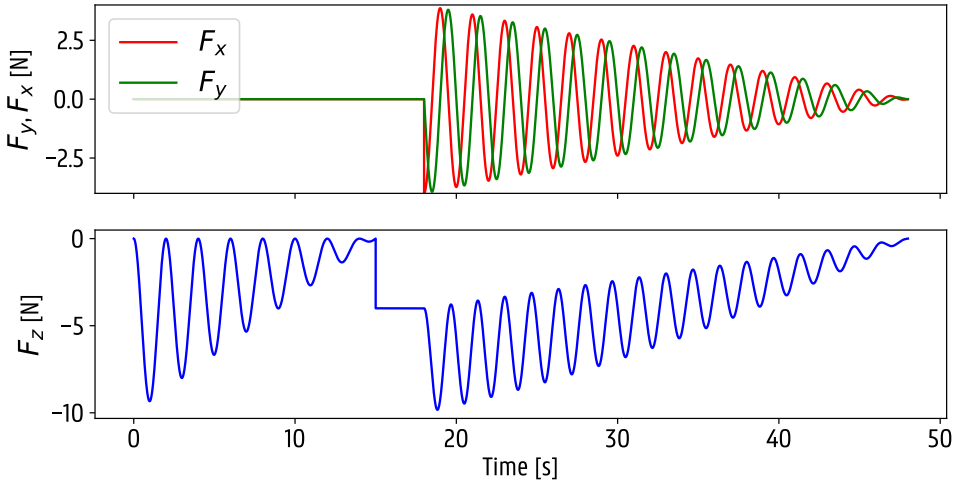


Figure 3.11: The alternative faster calibration procedure. A constant normal force, together with two descending sine waves in quadrature applied to the X- and Y-component of the force. Together they form a planar circular motion.

3.2.4 Reversed automatic calibration procedure

In a real life application the sensor is typically going to be mounted on a gripper, like the Robotiq 2F-85 or the Schunk EGK 40-MB-M-B gripper. In this case the sensor is not fixed to the tool flange. This means that the sensor will move with the gripper. To be able to calibrate the sensor this way, an external probe setup had to be created. This probe consists of an aluminium profile fixed to the table, with again a dome shaped probe at the top. This setup can be seen in figure 3.13.

The calibration procedures stay the same, but the reference frame is different. To compensate this difference, the code was adjusted. Before starting the procedure, one can set a transformation matrix consisting of the sensor frame unit vectors expressed in the base frame of the robot. The written software, available on GitHub [1], takes care of the rest.

This method of calibrating was also described in a workshop paper [16] for the annual International Conference on Robotics and Automation (ICRA) conference, this year held in Yokohama, Japan. Due to the relevance of the paper to this section of the dissertation, the paper is attached as Appendix A.

[1] https://github.com/LowiekVDS/masters-thesis

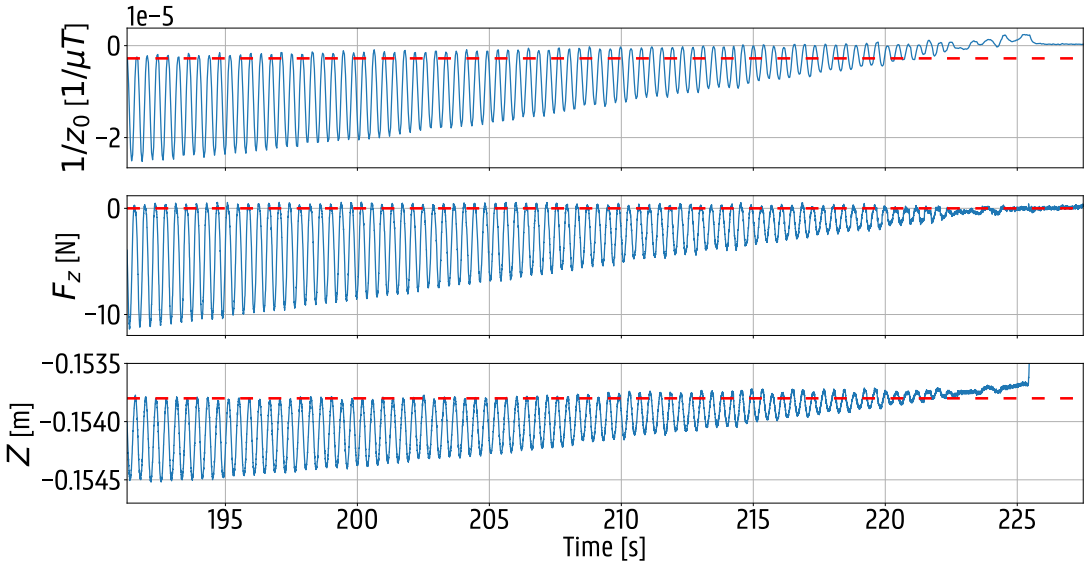


Figure 3.12: The first Stage of the calibration procedure with an excitation frequency of 3Hz. Notice how both the magnetic measurement and the robot position starts to deviate from the reference line, while the supposedly applied force stays the same.

The method works fine on the condition that the robotic arm is still stiff enough. For the Schunk gripper, this condition is fulfilled, albeit marginally. The Robotiq 2F-85 gripper turned out to be not stiff enough, introducing too much uncertainty in the calibration data.

3.3 Calibration Readout Software

Python software was developed to read out the measurements of the force/torque (F/T) sensor and the sensor array simultaneously. More specifically, a MeasuringInterface class was constructed that holds and manages two Thread objects. One Thread for the readout of the F/T sensor and a second one for the sensor array.

Both threads save their data in a separate csv-file, which can be later used to train the different models. Aside from csv, the data is also dumped on a UDP port (User Datagram Protocol). This way, realtime monitoring of the data is possible using external software. During calibration the open-source PlotJuggler [2] was used.

Reading out the F/T sensor works by polling the controller object given when connecting to the robot using the RTDE library [3]. The maximal datarate is about 200Hz.

The sensor array is read out by connecting to the microcontroller over Serial. Then, following the communication protocol described in Section 2.3.3, the incoming data is parsed. The sensor array readout function first compen-

[2] https://plotjuggler.io/

[3] https://github.com/UniversalRobots/RTDE_Python_Client_Library

sates overflow. More specifically, the measurements are unsigned integers with a length of 2 bytes. When a jump is detected of at least 2^{15}, the value is increased or decreased by 2^{16}. This makes sure the measurements are continuous to aid in model training. The raw bytes are then converted to μT, based on the sensor settings such as gain and resolution.

3.4 Stage I: per-taxel force model

The goal of the first Stage is to convert relative magnetic field measurements to a three-dimensional force vector on a per-taxel basis. This allows one to work with the more easily interpret-able force values instead of magnetic fields.

As can be seen in Figure 3.14 the response of the different taxels in the array can heavily vary. Possible causes are human errors in the magnet insertion process and inconsistencies in the printed dome structure. Notice how the response direction may also vary as a result of rotating the magnets 180 degrees, causing the north and south poles to be flipped. Therefore, a unique model per taxel was trained.

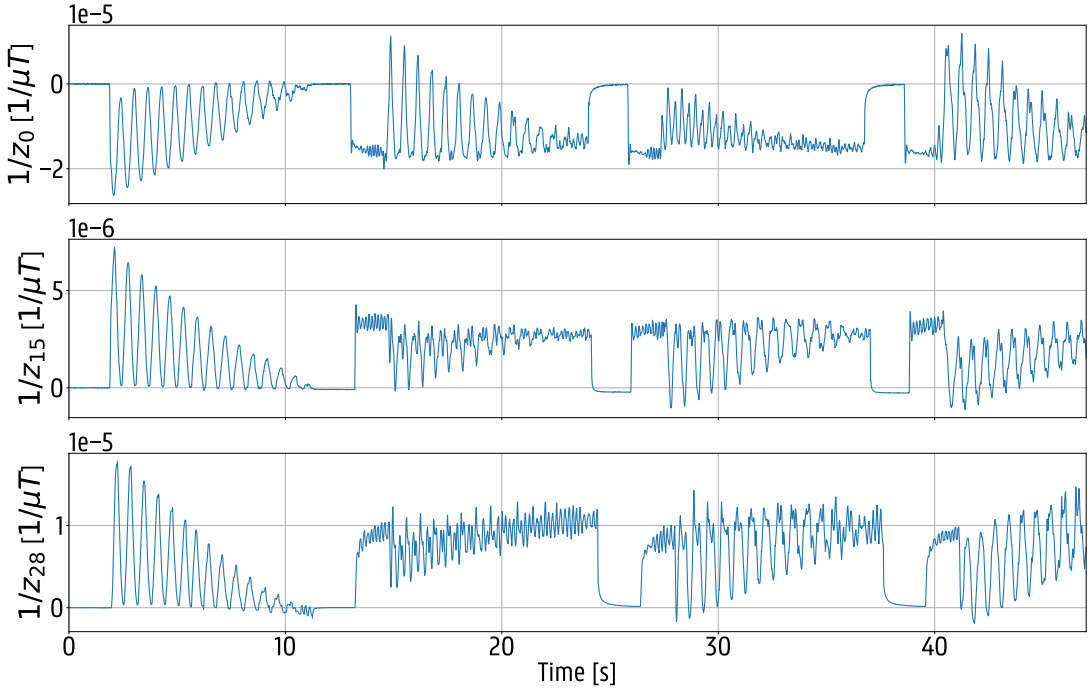


Figure 3.14: A comparison of the z component of the magnetic measurements from the calibration data of taxels 0, 15 and 28.

Mathematically the problem can be defined as finding the three-dimensional force model $\overrightarrow{F_i}(x_i, y_i, z_i)$ per taxel i, where x_i, y_i and z_i are the measurements of the magnetic field produced by the moving magnet. An overview

of the first Stage is given in Figure 3.15.

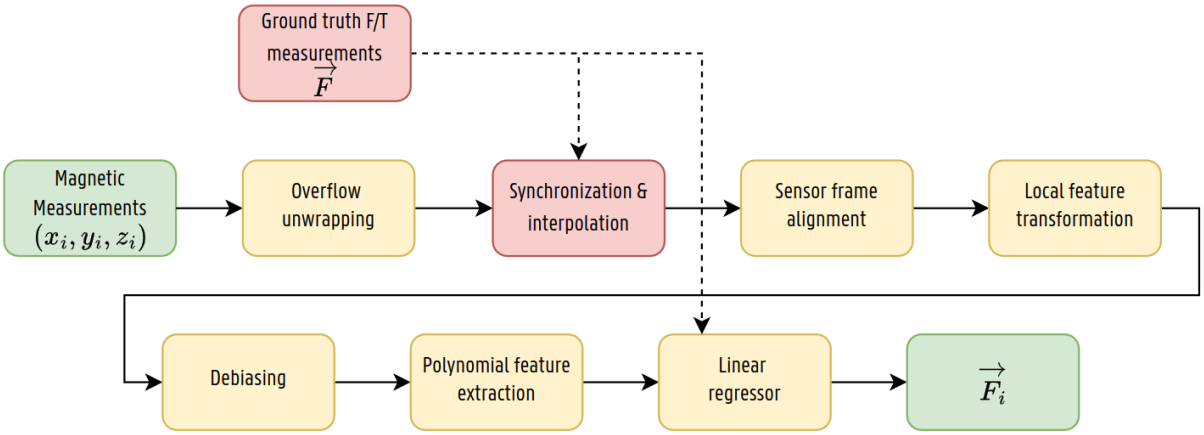


Figure 3.15: An overview of the first Stage for taxel i, the first part of the two-stage force model, see 3.1. Most of the steps are preprocessing. The red blocks, synchronization & interpolation and the measurements from the F/T-sensor, are only used during training.

3.4.1 Overflow unwrapping

The MLX90393 sensors report their measurements as a two-byte unsigned integer per measurement axis. Based on the selected sensitivity (Chapter 2) this gives us a range of approximately 15.9 mT for the Z axis and approximately 9.83 mT for the X and Y axes. While these ranges are large enough, sometimes depending on the position of the magnets above the taxels, the measurements are fairly close to 0 or 65535, the maximal two-byte unsigned integer. So when applying a force that moves a magnet, an overflow from 65535 to 0 or vice versa might occur. This overflow causes discontinuities which negatively influence the fitting of the linear model.

To fix this, per datastream, so per axis per taxel, an integer counter is kept. When an absolute jump of 65535 is detected, the counter is increased or decreased by one, depending on the direction of the jump. Finally, the counter is multiplied with 65535 and added to the original signal.

3.4.2 Synchronization & interpolation

The first step of the preprocessing pipeline involves synchronizing the timestamps of the raw sensor data with those of the ground truth measurements from the F/T sensor. Synchronizing the two datastreams is essential, since this allows us to drop the time dimension altogether, drastically simplifying the model. While it would be possible to estimate this latency difference by further analysing the communication aspects of the system like in Section 2.3.3, instead a data-drive approach is proposed.

Lag hysteresis

A non-linearity found in a lot of different engineering domains is hysteresis. Hysteresis is a dependence of the state of a system on history. In other words, the system has memory that influences the system output. This effect is typically unwanted and can also be quite challenging to model mathematically.

One type of hysteresis is caused by lag between the input and output. In other words, this hysteresis will occur when an input signal $X(t)$ is processed by a general system $\mathcal{S}$ taking t_0 time to process. This system will produce an output $Y(t) = \mathcal{S}(X(t - t_0))$.

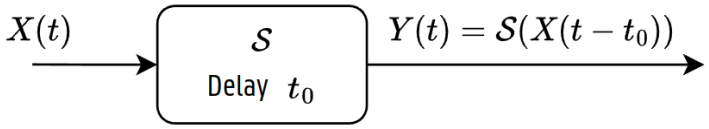


Figure 3.16: A general system $\mathcal{S}$ producing a delayed output $Y(t)$.

As an example, the descending sine wave from the calibration procedure, shown in Figure 3.7, was delayed by 100ms and 500ms and plotted against the original non-delayed version in Figure 3.17. Notice how when no delay is applied ($t_0 = 0$) the plot is a straight line. For non-zero delays hysteresis loops start to form and the higher the delay, the larger the loops become.

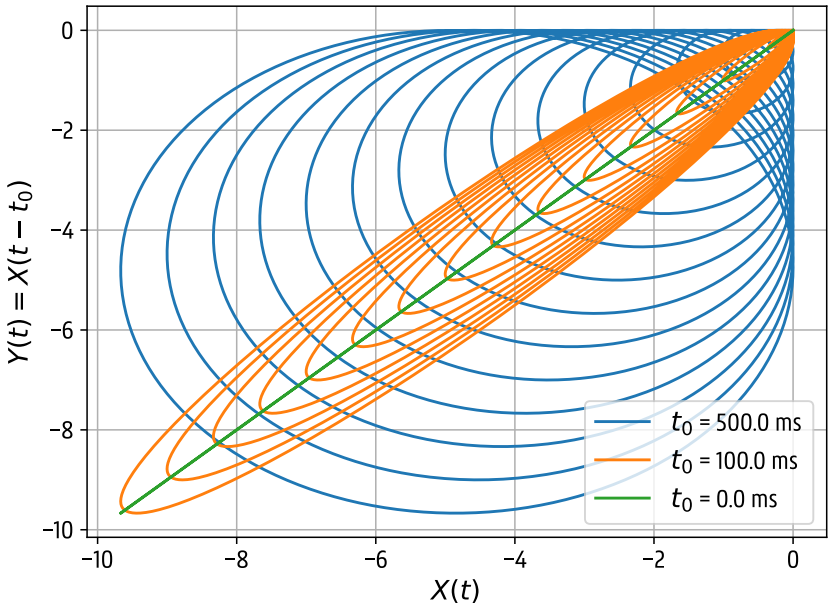


Figure 3.17: An example of lag hysteresis. Here the descending sine wave of the calibration procedure with a 0.5Hz was used as an input, delayed by different amounts and then plotted against itself. Notice how for non-zero delays loops are formed, clearly indicating hysteresis. The system $\mathcal{S}$ represents an ideal delay system.

Also notice in Figure 3.18 how this kind of hysteresis applied to the descending sine wave is also rate dependent. The higher the frequency content of the descending wave, the larger the hysteresis loops for a given delay t_0.

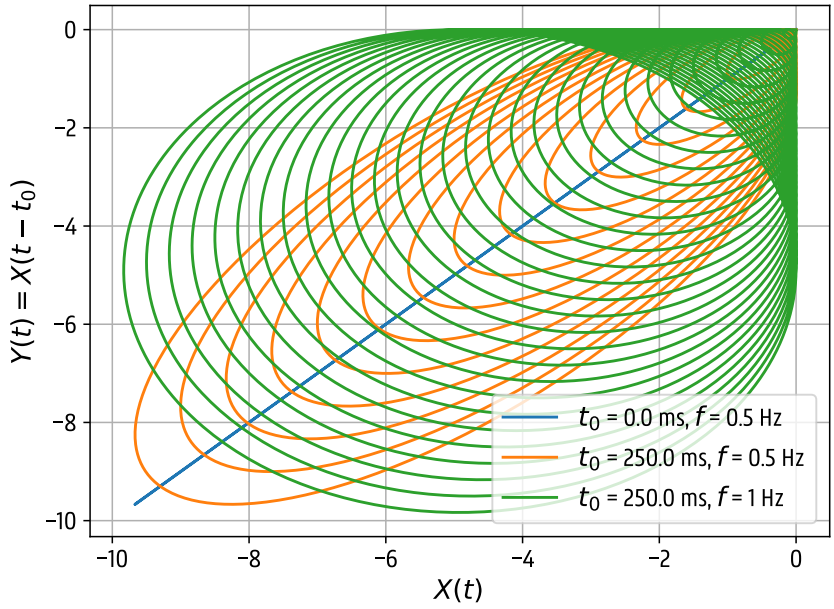


Figure 3.18: An example of the rate-dependency found in lag hysteresis. Notice how the hystersis loops become larger for the 1Hz excitation (green) compared to the 0.5Hz excitation (orange), even when the delay t_0 stays the same. The system $\mathcal{S}$ represents an ideal delay system

This can be explained by looking at what happens when delaying a sinusoidal signal, which forms the basis of the descending wave. Delaying $X(t) = \sin(2\pi f t)$ by a delay t_0, gives us:

$$Y(t) = X(t - t_0) = \sin(2\pi f t - 2\pi f t_0) \tag{3.15}$$

Looking at this expression one can see that the phase lag of the signal is equal to $-2\pi f t_0$. This signal is in essence the memory of the system, or in other words responsible for the hysteresis loops. It is both dependant on the frequency f and the delay t_0.

Applied to time synchronization

Suppose on the system diagram in Figure 3.16 the input is set to the magnetic field measurements from the 4x8 sensor, at the time they are written to disk. Furthermore, the model $\mathcal{S}$ is a black box able to convert these measurements into the measured force by the F/T-sensor, also at the moment it is written to disk. Then, the delay of the system t_0 can be interpreted as the latency difference between the two datastreams. Note that we thus also have $Y(t)$.

A third block in the diagram can be added to compensate this delay. This was done on Figure 3.19. The block $\mathcal{D}$ was added which produces a signal that leads ahead by t_1 time. Note that this system is not possible in real-time, but this is not applicable here. When the lead time t_1 equals the system delay t_0, the lag hysteresis should disappear.

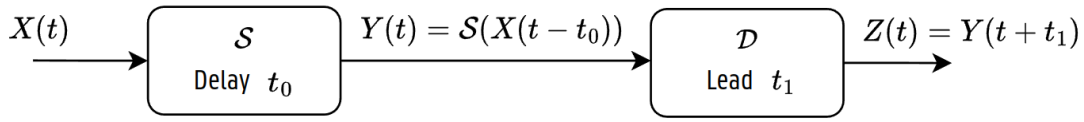


Figure 3.19: A general system $\mathcal{S}$ producing a delayed output $Y(t)$, with a delay of t_0. Followed by a unity "lead" system **D** that makes the signal lead forward by t_1 time units. The signals $X(t)$ and $Y(t)$ are known. The key to finding the introduced delay t_0 is to find t_1 that minimizes the hysteresis lag, since then $t_0 = t_1$

For the real sensor the first step in the calibration was used for taxel 0: the descending sine wave applied as normal force, at 3Hz. After these simulations, the latency was determined to be around 80ms, where t_1 was positive, meaning that the F/T-sensor lags behind. A comparison between $Y(t)$ and $Z(t)$ plotted against $X(t)$ is given in Figure 3.20. Note that even when lag hysteresis would be the only non-linear effect in $\mathcal{S}$, the plot will not show a straight line in this case. Indeed, the normal force and the inverse of the Z-component of the magnetic measurements $1/z_0$ have a cubic relationship. The main key is to look at the width of the hysteresis loops. Further validation of this latency will be presented in the results chapter, since the original system $\mathcal{S}$ is actually the two-stage model we are trying to create. The key takeway is that during training, the F/T-sensor measurements are delayed with 80ms.

Interpolation

The two data streams are sampled at different rates, resulting in varying lengths, which complicates model fitting and necessitates retaining the time dimension. To address this, a unified set of timestamps is created by combining all timestamps from both data streams. Subsequently, linear interpolation is applied to ensure every timestamp corresponds to a data point in both streams.

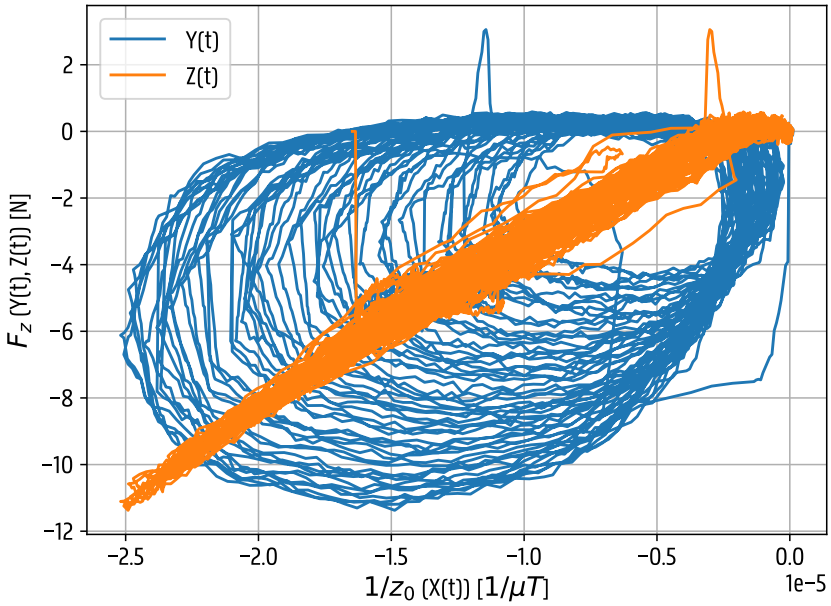


Figure 3.20: This plot shows the signals $Y(t)$ and $Z(t)$ plotted against the input $X(t)$. The plotted signal $Z(t)$ is the one where the lag hysteresis is the least visible, meaning the plot represents a third order polynomial the most. This was used to estimate the latency t_0 between the sensor array and F/T-sensor to be around 80ms.

3.4.3 Sensor frame alignment

The robotic arm executes a homing procedure to automatically find the positions of the taxels in the sensor array. An extra transformation from TCP to the sensor frame can also be set to accommodate for non-default orientations during calibration. Throughout the calibration experiments, setup errors may occur causing the applied force frame (ie. the base frame of the robot, see the coordinate frame on Figure 3.5) to not match the local sensor frame as shown in Figure 2.11. More specifically, the difference between the frames is typically a rotation around the Z-axis of the base frame. This rotation error can also be seen in Figure 3.6 by noticing the 4x8 grid is not perfectly aligned with the X and Y axis of the plot. The fit scores will not be affected since this is just a linear transformation, similar to the magnet not being perfectly aligned. But when someone applies a force in the actual sensor frame, this force will decompose into an X and Y component along the slightly rotated sensor frame used during calibration. It also makes it difficult to cross-validate different experiments with different setups.

Luckily, the calibration data also includes the X and Y movement of the robotic arm with respect to the base frame. This meant it is possible to calculate and correct the rotation error. This preprocessing step thus aims to rotate the X and Y components of the measurements by angle θ, for all taxels i:

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} * \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{3.16}$$

The rotation angle θ is derived from the calculated taxel positions during the homing procedure, seen in Figure 3.6. The angle was determined to be 1.82 deg.

3.4.4 Local feature transformations

Based on the design of the taxel domes the idle distance between the center of the magnet and the sensor z_0 is found to be 3.27mm. During automatic calibration the TCP was also recorded, which means the actual movement of the magnet was also recorded. Under the applied forces, the data shows that the magnet does not move more than 1mm up and down along the Z-direction. Furthermore, the magnet does not deviate more than 0.5mm from the idle position in the planar direction. In other words: $z_0 = 3.27mm$, $\Delta z < 1mm$ and $\Delta\rho < 0.5mm$. Under the assumption of Hooke's law, the conclusion can be made that the measurements of the Z-component will be inversely proportional to the cube of the applied normal force.

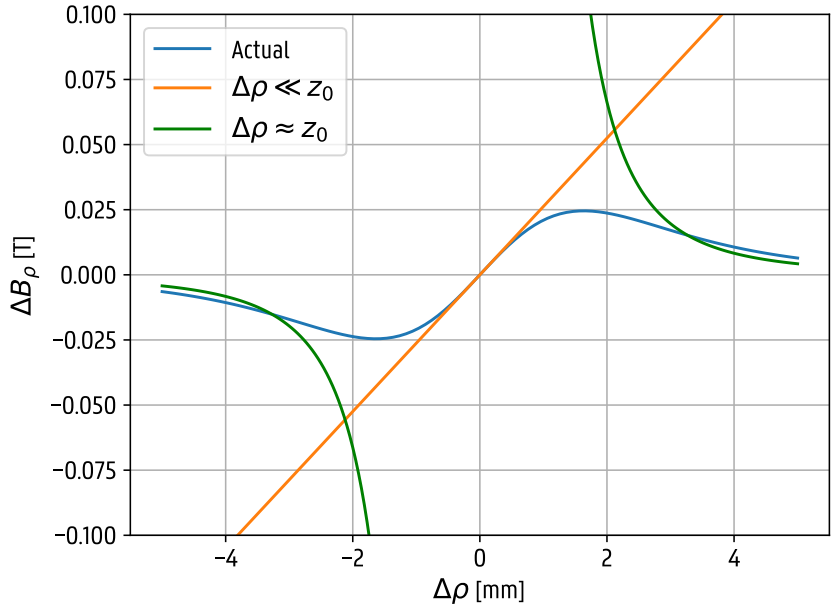


Figure 3.21: The change in magnetic field along the planar direction ΔB_ρ for changes in ρ around the idle point of $P_0 = (3.27, 0.0)$mm. The approximations in both assumptions are plotted as well. Notice how the linear approximation holds up for the small deviations of $\Delta\rho$ measured during calibration.

Because of the small $\Delta\rho$ compared to z_0, the magnet is working in the linear region of the planar magnetic field changes. Plotting the change of the planar magnetic field $\Delta B_z'$ around the idle position $P_0 = (3.27, 0.0)$mm when shifting the magnet along the planar direction confirms this. Shown in Figure 3.21, notice how the linear assumption holds for small values of $\Delta\rho$. The other inverse cubic approximation comes indeed into play when the magnitude of $\Delta\rho$ is similar to $z_0 = 3.27$mm. Zooming in on the region of interest (Figure 3.22, $\Delta\rho = 0.5$mm,

further confirms that the linear assumption holds. Concretely, a maximum error of 7.4mT is calculated, which is a deviation of 5.9%.

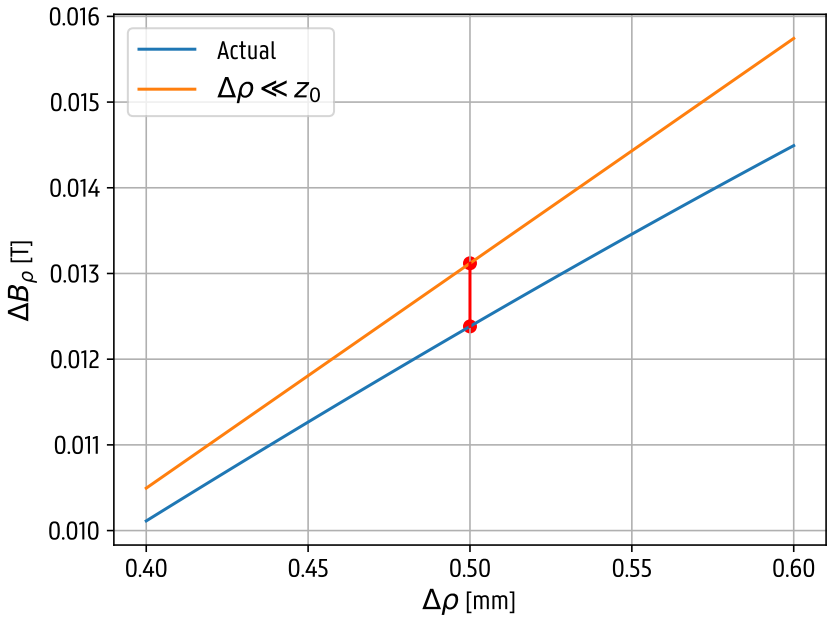


Figure 3.22: A zoomed in version of Figure 3.21. The maximal deviation from the linear approximation is shown in red. It is calculated to be a deviation of 7.4mT or 5.9%.

Because Stage I is a per-taxel linear regressor, there should exists a linear relationship between the measurements and the applied forces which are used as labels. This is the reason why the Z-component of the magnetic measurements are inverted. In other words, per taxel i this feature transformation is applied:

$$\begin{cases} x_i^{'} &= x_i \\ y_i^{'} &= y_i \\ z_i^{'} &= \frac{1}{z_i} \end{cases} \tag{3.17}$$

No special attention has to be given to the fact that the inversion of the Z-component might introduce numerical instabilities such as division by zero, because the magnetic field will always be non-zero under normal operation. An example will be shown in the results chapter that further shows why this feature transformation is actually necessary.

3.4.5 Debiasing

After the feature transformation, the features are all individually debiased. Debiasing is implemented on feature-by-feature basis by subtracting the mean of the initial 100 samples, corresponding to about 1 to 2 seconds, from all samples of that feature. This step is primarily implemented to make sure the models are trained on the variations of the magnetic field when applying a force instead of on the absolute value, which is not that important. Also, this step negates all drift-related effects of Hall sensors, such as temperature, (material) environment and initial pressures as a result of (different) surface layers. In practice, this debiasing step can be a part of the sensors boot procedure.

3.4.6 Polynomial feature extraction

As shown previously, the relationship between the Z-measurements and the applied normal force is (inversely) cubic, while the relationship between the planar force and the planar measurements is linear. Additionally, extra nonlinearities are likely introduced by errors in assumptions such as Hooke's law and the models used for the magnetic fields themselves. Instead of directly fitting a polynomial regressor to this data, it is more effective to generate new polynomial features.

An input feature $\overrightarrow{M}$ of dimension 3 ($\overrightarrow{M} = (x, y, z)$) given to the polynomial feature extractor produces a new feature vector $\overrightarrow{N}$ including all powers of the features up to a certain degree k and their cross-products, ie. $\overrightarrow{N} = (1, x, y, z, x^2, y^2, z^2, xy, xz, xyz, x^3, ...)$, depending on the degree. It is important to be cautious when selecting the maximum degree k of the polynomial feature extractor. The general formula for the number of features created with polynomial degree n for an input vector with dimensionality d and an output vector of dimension m is given by:

$$\frac{(n+d)!}{n!d!} * m \tag{3.18}$$

Increasing the degree can thus significantly raise the number of features and, consequently, the complexity of the linear regressor. This not only increases the risk of overfitting but also makes the model more difficult to execute. This will be shown in the results chapter.

After the transformation, the relationship between the new feature set $\overrightarrow{N}$ and the labels has now become linear again, allowing the use of a standard linear regressor.

3.4.7 Fitting the linear regressor

The code of the first Stage can be found on the GitHub repository provided with this masters dissertation. A model was trained per taxel. Per taxel, the calibration data (after preprocessing) is randomly split into a train/test set.

The train set encompasses 80% of the total sample count, leaving 20% out for the test set. Since a linear regressor was used (no regularization), the model has no hyperparameters, and thus a third validation set was not required.

The polynomial feature extraction and linear regression models were created using scikit-learn [4]. The polynomial feature extractor produces bias features (1), while the linear regressor is configured to not fit an intercept.

3.5 Stage II: Compensating cross-talk

The purpose of Stage II is to compensate for inter-taxel magnetic cross-talk. It is important to realize that the Hall sensors measure a superposition of the magnetic fields of the magnet directly above the sensor and the magnets of neighbouring taxels. When naively applying the taxel models from Stage I while only exciting a single taxel, ghost forces will be predicted on neighbouring taxels. Those ghost forces are accurate in the sense that the prediction would actually be the force required on the neighbour to produce the same magnetic response from the magnet directly above the neighbouring taxel. However, this effect is undesirable. One could opt for a different approach in Stage I where the measurements of the neighbours are directly taken into account, but this greatly increases the complexity of the model and increases the risk of overfitting to the training data. This results in unstable models. Instead, a different method is proposed. An overview can be seen in Figure 3.23.

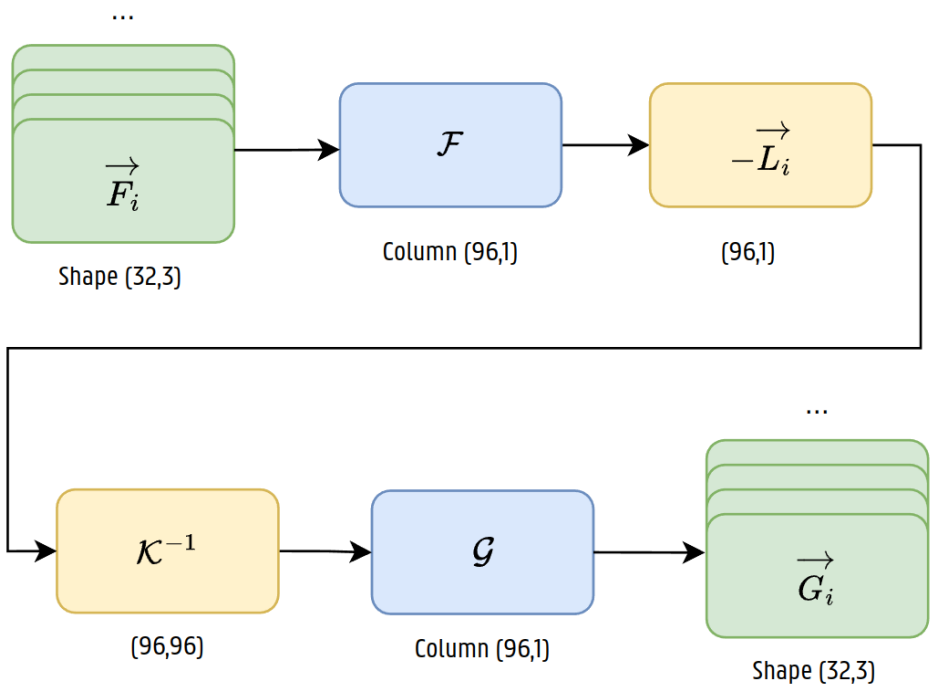


Figure 3.23: An overview of the second Stage for the two-stage model. As will be explained the second Stage boils down to the multiplication of a column containing the results of Stage I with a matrix $\mathcal{K}$

[4] https://scikit-learn.org/stable/

3.5.1 Superposition principle

In Stage I the assumption was made that no cross-talk occurs. In other words, the assumption was made that the output force of taxel i, $\overrightarrow{F_i}(x_i, y_i, z_i) = \overrightarrow{F_i}(\overrightarrow{B_i})$ only depends on the measurements $\overrightarrow{B_i}$ of the taxel i. In reality, this force depends on the measurements of all taxels in the array since the magnetic field of any magnet extends infinitely. In practice, it is shown in Figure 3.24 that from a certain distance the influence of a magnet becomes negligible. For computational reasons, this Stage thus assumes the cross-talk effects can be fully explained and compensated by only taking the measurements of the neighbors with a hop count of at most 1 into account. The hop count of a neighboring taxel is defined as the number of hops that must be made to reach the neighbor. A hop can be taken in the horizontal, vertical, and diagonal directions. See Figure 3.25 for an example.

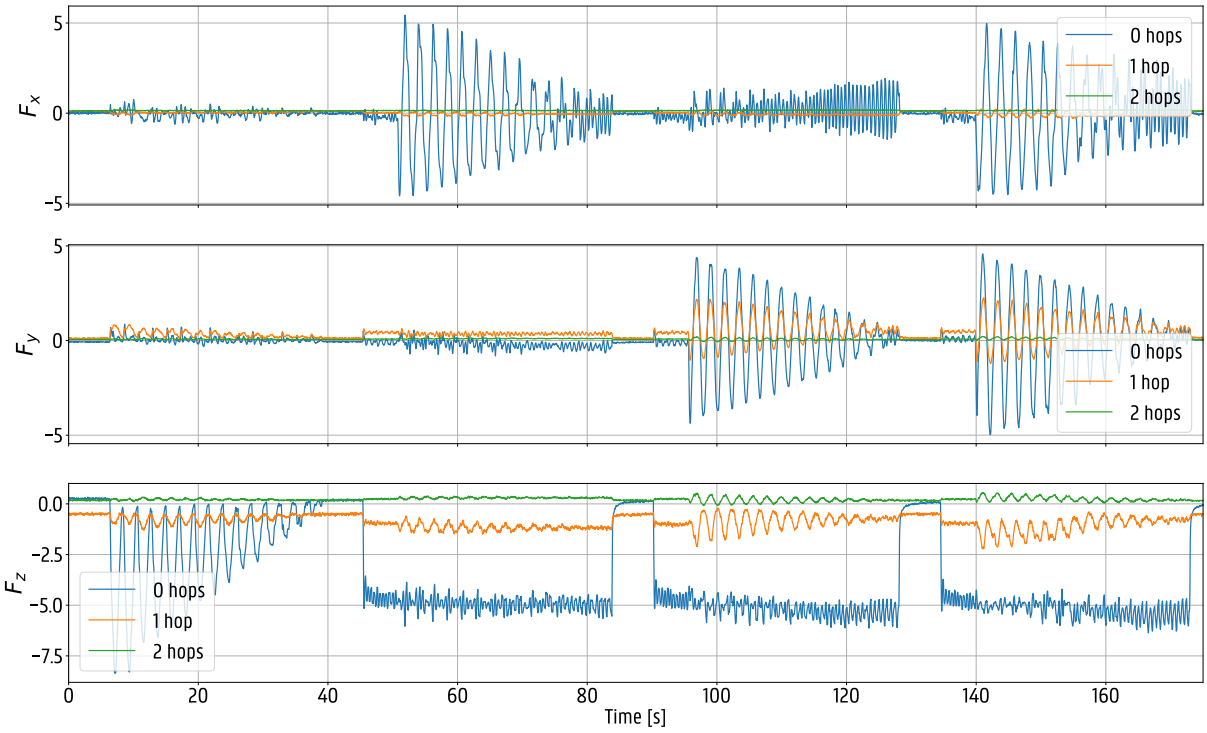


Figure 3.24: The force predictions of Stage I of a taxel compared to two neighbours with hop counts of 1 and 2. Only the main taxel (hop count 0) was actually actuated. Notice how the influence becomes negligible for hop counts of 2 (and more).

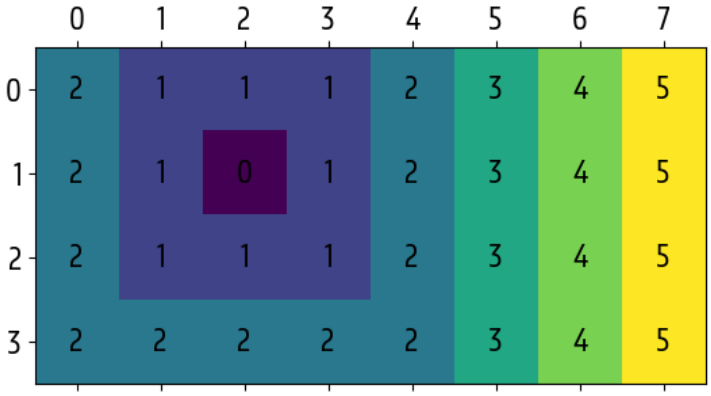


Figure 3.25: The hop count of all taxels, relative to the taxel on the second row and the third column (the square labeled with 0). The second Stage of the two-stage force model only takes the measurements of the neighbours with a hop count of at most 1 into account for crosstalk compensation.

A more accurate description of the force model devised in Stage I can be given by using the superposition principle:

$$\overrightarrow{F_i} = \sum_{j \in N_i} \overrightarrow{K}_{j,i}(\overrightarrow{G_j}) \tag{3.19}$$

Where N_i is the set of neighbors of taxel i with a hop count of at most 1, $\overrightarrow{K}_{j,i}$ is a model for the cross-talk component of the force in taxel i caused by taxel j, and $\overrightarrow{G_j}$ is the actual, cross-talk compensated force of taxel j. Note that N_i also includes taxel i, as taxel i has a hop count of zero to itself.

In the above equation Stage I technically gives us the force with cross-talk, $\overrightarrow{F_i}$, while Stage II aims to produce the corrected $\overrightarrow{G_i}$. Note that when only a single taxel is excited, the assumption of Stage I holds and thus the superposition equation simplifies to $\overrightarrow{F_i} = \overrightarrow{K}_{i,i}(\overrightarrow{G_i}) = \overrightarrow{G_i}$. Also because the goal in Stage I is to predict the force on the taxel by exciting that taxel, the model $\overrightarrow{K}_{i,i}$ should always be unity.

During the calibration process the measurements of all other taxels were also recorded. Therefore it is possible to determine the other models $\overrightarrow{K}_{j,i}$. The complexity of this problem heavily depends on the chosen structure of the models $\overrightarrow{K}_{j,i}$. In what follows a few options are discussed.

3.5.2 Assumption of linear cross-talk

When the assumption is made that cross-talk caused by neighbouring taxels is linear, the model $\overrightarrow{K}_{j,i}$ can be written as a 3x3 matrix. As stated before the model used for the main (non-crosstalk related) contribution $\overrightarrow{K}_{i,i}$ should be unity, so under the linear assumption it should be the 3x3 identity matrix.

The superposition equation 3.19 can be rewritten as a sum of matrix products between the models $\overrightarrow{K}_{j,i}$ and the 3x1 vector $\overrightarrow{G_j}$:

$$\overrightarrow{F_i} = \sum_{j \in N_i} \overrightarrow{K}_{j,i} * \overrightarrow{G_j} \tag{3.20}$$

By combining all the taxels into a single set of equations, the problem can be written as a single block matrix:

$$\begin{bmatrix} \overrightarrow{F_0} \\ \overrightarrow{F_1} \\ \vdots \\ \overrightarrow{F_{31}} \end{bmatrix} = \begin{bmatrix} I_3 & \overrightarrow{K}_{0,1} & \dots & \overrightarrow{K}_{0,31} \\ \overrightarrow{K}_{1,0} & I_3 & \dots & \overrightarrow{K}_{1,31} \\ \vdots & & \ddots & \vdots \\ \overrightarrow{K}_{31,0} & \overrightarrow{K}_{31,1} & \dots & I_3 \end{bmatrix} * \begin{bmatrix} \overrightarrow{G_0} \\ \overrightarrow{G_1} \\ \vdots \\ \overrightarrow{G_{31}} \end{bmatrix} \tag{3.21}$$

Which can be notationally simplified as:

$$\mathcal{F} = \mathcal{K} * \mathcal{G} \tag{3.22}$$

A compensated force model $\mathcal{G}$ can then be found by inverting the matrix $\mathcal{K}$:

$$\mathcal{G} = \mathcal{K}^{-1} * \mathcal{F} \tag{3.23}$$

Note that depending on the structure of the sensor array and the chosen influence sphere of the taxels, some, if not most, block matrices $\overrightarrow{K_{i,j}}$ may be artificially forced to be the zero matrix. Under the linear crosstalk assumption there is on a computational level no benefit of zero-forcing certain blocks by setting an upper limit on the influence sphere of a taxel during inference. That being said, from a certain distance the influence of a taxel will be small enough that it might be better to actually ignore said influence.

A note about bias

A more general approach would be to include a bias term in the models. That is, rewriting equation 3.19 as:

$$\overrightarrow{F_i} = \overrightarrow{L_i} + \sum_{jeN_i} \overrightarrow{K}_{j,i} * \overrightarrow{G_j} \tag{3.24}$$

In theory it can be shown that the bias term should always be zero. Indeed, when equating $\overrightarrow{G_j} = \overrightarrow{0}$ for all j, meaning no actual force is applied to any taxel, Stage I should also produce zeroes as predictions, eg. $\overrightarrow{F_j} = \overrightarrow{0}$ for all j.

That being said, due to errors in the prediction of Stage I the above assumption might not hold up. While strictly not being the issue being tackled in Stage II, the chosen method lends itself to also correcting this bias. Indeed, During the calculation of $\mathcal{K}$, a bunch of linear regressors are fit, including the intercepts (see implementation below). The fitted intercepts $\overrightarrow{A_j}$ of all neighbours of taxel i are then averaged to become the actual bias term:

$$\overrightarrow{L_i} = \frac{1}{\aleph_i} \sum_{jeN_i} \overrightarrow{A_j} \tag{3.25}$$

Where $\aleph_i$ represents the number of neighbours of taxel i with a hop count of at most 1 (thus including itself as usual) and A_j is the intercept term as fitted using the proposed model below. Note that $\aleph_i$ can range from 9 for the center taxel to 6 or 4 for respectively the edge and corner taxels. Finally, the bias terms can be subtracted from the predicted Stage I force, which results in the known form:

$$(\overrightarrow{F_i} - \overrightarrow{L_i}) = \sum_{jeN_i} \overrightarrow{K}_{j,i} * \overrightarrow{G_j} \tag{3.26}$$

And:

$$\mathcal{G} = \mathcal{K}^{-1} * (\mathcal{F} - \mathcal{L}) \tag{3.27}$$

In this thesis the forces $\overrightarrow{F_i}$ from Stage I are implicitly assumed to be compensated to have zero bias, such that in general the simpler (non-bias) notation can be used.

Implementation

The problem boils down to finding the square matrix $\mathcal{K}$ and inverting it. Before this inversion is possible, all block matrices $\overrightarrow{K}_{j,i}$ should be calculated.

To show how this can be done using the same calilbration data as in Stage I, the calculation of $\overrightarrow{K}_{2,1}$ is used as an example. During the 32 calibration procedures used for creating the Stage I models the response of the other sensors are also recorded. Thus it is possible to predict $\overrightarrow{F_2}$ using the calibration data used to create the Stage I model of taxel 1. Furthermore, the robotic arm only applies a force to a single taxel, meaning that the vector $\mathcal{G}$ during a calibration procedure only has up to a single non-zero block. To make sure the diagonal elements of block matrix $\mathcal{K}$ are still identity matrices, the actual force $\overrightarrow{G_1}$ is actually the predicted force $\overrightarrow{F_1}$ instead of the training labels as measured by the robotic arm. Note that because the force $\overrightarrow{F_1}$ was predicted while the robot was actually applying force to taxel 1, there would be no inter-magnetic cross-talk, justifying the identity matrices. Finally, having $\overrightarrow{F_2}$ and $\overrightarrow{G_1}$ while assuming there are no forces applied to the taxels (meaning all other blocks of $\mathcal{G}$ are zero) allows us to single out $\overrightarrow{K}_{2,1}$:

To illustrate the process using the same calibration data as in Stage I, let's consider the calculation of $\overrightarrow{K}_{2,1}$ as an example. During the calibration procedures used for creating the Stage I models, which result in $\overrightarrow{F_i}$, the responses of the other sensors are also recorded. Thus, it is possible to predict $\overrightarrow{F_2}$ using the calibration data that created the Stage I model of taxel 1 and the Stage I model of taxel 2. Additionally, since the robotic arm only applies force to a single taxel during calibration, the vector $\mathcal{G}$ during this procedure has at most a single non-zero block.

To ensure the diagonal elements of the block matrix $\mathcal{K}$ are identity matrices, the actual force $\overrightarrow{G_1}$ is considered to be the predicted force $\overrightarrow{F_1}$ rather than the training labels measured by the robotic arm. Because $\overrightarrow{F_1}$ is predicted while the robot is applying force to taxel 1, there is no inter-magnetic cross-talk, which justifies using identity matrices for the diagonal elements.

Finally, having $\overrightarrow{F_2}$ and $\overrightarrow{G_1}$ (while assuming no forces are applied to the other taxels, meaning all other blocks of $\mathcal{G}$ are zero) allows us to single out $\overrightarrow{K}_{2,1}$:

$$\overrightarrow{F_2} = \sum_{j \epsilon N_i} \overrightarrow{K}_{j,i} * \overrightarrow{G_j} = \overrightarrow{K}_{2,1} * \overrightarrow{G_1} \tag{3.28}$$

Then the same methods of Stage I can be used to estimate $\overrightarrow{K}_{2,1}$. In practice, a two-step model was used: a polynomial features extractor followed by a linear regressor. The polynomial feature extractor just like in Stage I produces bias features, while the linear regressor is configured to not fit an intercept. The weights multiplied with the bias features are then used to compensate the bias error coming from Stage I. The other weights are inserted in $\mathcal{K}$. Finally, the matrix $\mathcal{K}$ is inverted.

3.5.3 Higher-order model

The linear assumption of cross-talk can be extended to higher-order models, such as a third order polynomial, also used in Stage I. Similar to a polynomial feature extractor that was used in Stage I as a preprocessor for the linear regressor, the original Stage I outputs $\overrightarrow{F_i}$ are used to generate new polynomial features $\overrightarrow{F_{i,j}} = \overrightarrow{F_i}^j$ where the exponent j is applied in an element-wise manner.

As a result the superposition equation 3.19 can then be rewritten as:

$$\overrightarrow{F_{i,k}} = \sum_{j \epsilon N_i} \overrightarrow{K}_{j,i,k} * \overrightarrow{G_j} \tag{3.29}$$

with the following block matrix as a result, where the polynomial degree was limited to k:

$$\begin{bmatrix} \overrightarrow{F_{0,1}} \\ \overrightarrow{F_{0,2}} \\ \vdots \\ \overrightarrow{F_{0,k}} \\ \overrightarrow{F_{1,1}} \\ \vdots \\ \overrightarrow{F_{31,k}} \end{bmatrix} = \begin{bmatrix} I_3 & \overrightarrow{K}_{0,1,1} & \dots & \overrightarrow{K}_{0,31,1} \\ 0_3 & \overrightarrow{K}_{0,1,2} & \dots & \overrightarrow{K}_{0,31,2} \\ \vdots & & \ddots & \vdots \\ 0_3 & \overrightarrow{K}_{0,1,k} & \dots & \overrightarrow{K}_{0,31,k} \\ \overrightarrow{K}_{1,0,1} & I_3 & \dots & \overrightarrow{K}_{1,31,1} \\ \vdots & & \ddots & \vdots \\ \overrightarrow{K}_{31,0,k} & \overrightarrow{K}_{31,1,k} & \dots & I_3 \end{bmatrix} * \begin{bmatrix} \overrightarrow{G_0} \\ \overrightarrow{G_1} \\ \vdots \\ \overrightarrow{G_{31}} \end{bmatrix} \tag{3.30}$$

Notice that the block matrix 3.30 containing the models is no longer a square matrix of size $32*3 = 96$. Instead, the matrix has shape $(32 * M, 96)$. Here M is the total amount of polynomial terms per input component, excluding the bias. Under the linear assumption, M is equal to three: the direct contributions of F_x, F_y and F_z. When we take a third order model, M becomes 19. The block matrices $\mathcal{K}_{j,i,k}$ have the shape $(M, 3)$. Because the matrix is not square anymore, a pseudo-inverse should be used such as the commonly used Moore-Penrose inverse.

3.6 Implementation on a Raspberry Pi Pico W

Both the first Stage and second Stage of the two-stage force model was implemented on the Raspberry Pi Pico W. A Python script was written to convert the coefficients of the Stage I models and the matrix $\mathcal{K}^{-1}$ to C-style arrays using floating point numbers. Then functions were written to extract polynomial features and to apply the first and second stages to data. Floating point arithmetic was used, which is compute intensive, but techniques exist to make this easier to execute.

Serial communication

After the first and second stages have been applied, the result is an array of 96 float values, $\overrightarrow{G_i}$. These have to be communicated to the computer as usual. The same protocol as with raw measurements is used. The protocol starts with sending a start byte OxAA, followed by the raw values of the taxel measurements (which here are the force predictions) in the order the taxels have been read out. The difference is the datatype. The force predictions have the float32 datatype with a length of 4 bytes. We could just send the floats, but this doubles the communication overhead and as will be shown in the results chapter, this has an effect on the performance. Because the predictions should be in a rather small application range, the choice was made to convert them to two-byte unsigned integers. Mathematically this operation can be written as follows:

$$x_{\text{uint16}} = f(x_{\text{float32}}, x_{\text{min}}, x_{\text{max}}) = \begin{cases} 0, & x_{\text{float32}} < x_{\text{min}} \\ 65535, & x_{\text{float32}} > x_{\text{max}} \\ \text{round}(\frac{x_{\text{float32}} - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} * 65535), & \text{else} \end{cases} \tag{3.31}$$

Where round$(\cdot)$ rounds to the nearest integer value.

For the X and Y components of the predicted force the range was set to $[-10, 10]$N and for the Z component the range is set to $[-30, 10]$N. This operation is lossy, since a float32 consisting of 4 bytes is compressed into only 2 bytes. But because we know the expected range the forces should be in and these ranges are quite small, in practise there will be no noticeable loss in precision. Indeed, the sensitivity of the Z component is 0.62 mN and for the X and Y component 0.31 mN per LSB.

The advantage is that now we only have to send two bytes per axis per taxel, resulting in a total packet of 193 bytes (including start byte) like before.

3.6.1 Readout Graphical User Interface (GUI)

A script has been created to read out the sensor and to visualize it into a quiver plot using opencv [5]. First the incoming two-byte unsigned integers are converted back to float values by inverting equation 3.31. Then the values are used to create the quiver plot. The code can be found in the Github repository [6].

[5] https://opencv.org/

[6] https://github.com/LowiekVDS/masters-thesis

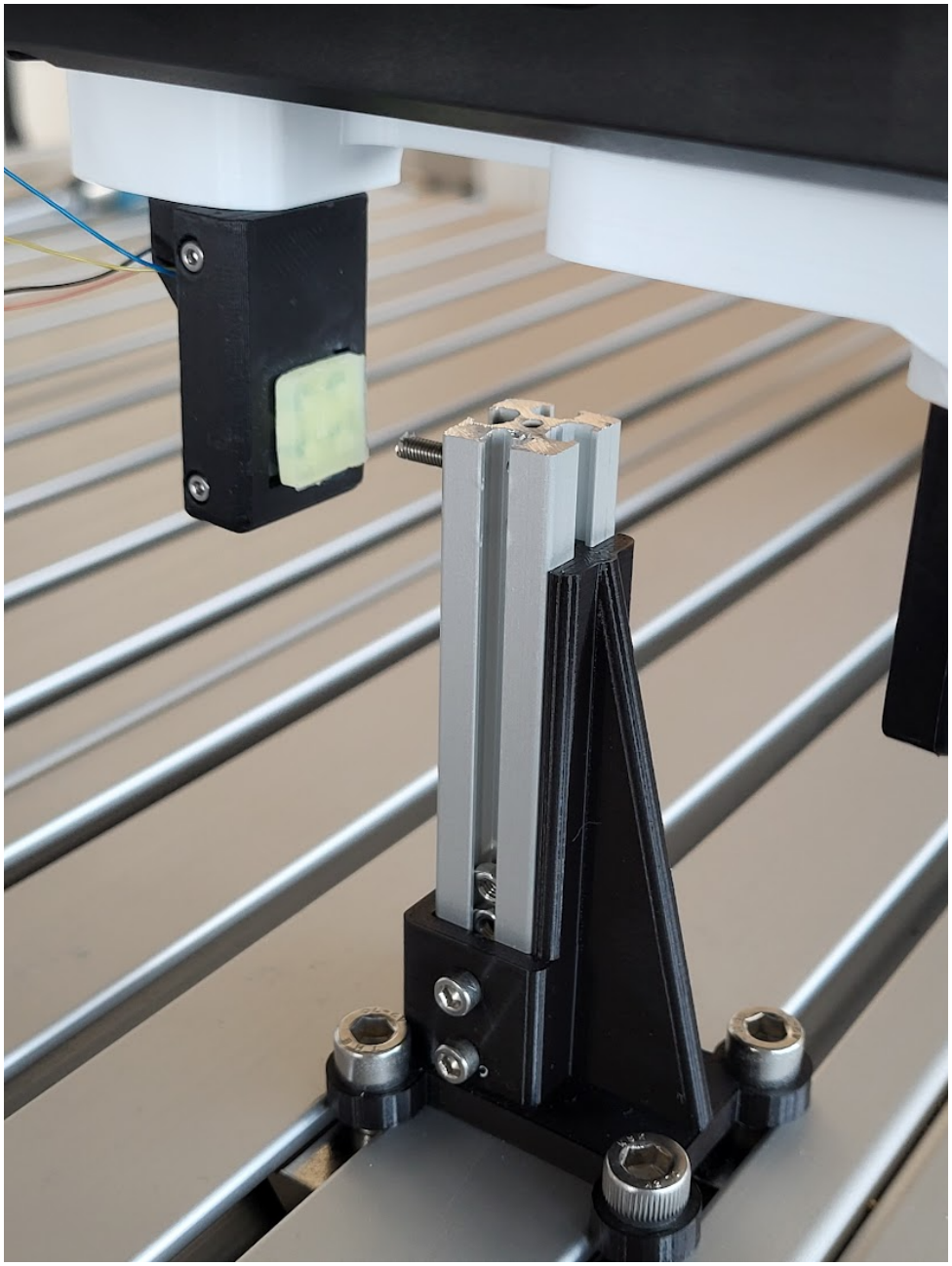


Figure 3.13: The reversed automatic calibration procedure. The sensor is mounted on a gripper, while the probe used is mounted on the table.

4

Results and discussion

4.1 Finger top silicone cover

During testing I noticed that the top silicone cover could use some improvements. The sides of the cast were designed to have a certain thickness to allow the silicon to rest on the top casing. When applying a force to this layer or placing for example a cable on this layer, these sides act a support structure, heavily influencing the measurements. See Figure 4.1 for an example. A tapered version, seen in Figure 4.2 should solve this issue.

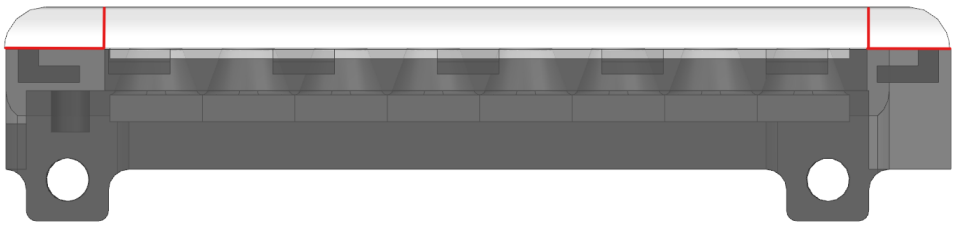


Figure 4.1: The silicone regions outside the red lines are solid and much thicker compared to the center silicone supported by the taxel domes. These solid regions act as a support structure when for example a cable lies on top of it, influencing the measurements in a negative way.

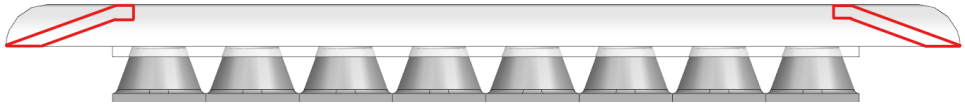


Figure 4.2: A tapered top silicone layer. This should solve the issues caused by the support on the sides, annotated in Figure 4.1 ,when grasping for example a cable.

4.2 Evaluation metrics

To make sure the different approaches and steps of the pipeline can be properly compared to each other, a few evaluation metrics are used.

4.2.1 Normalized Mean Squared Error (NMSE)

In the context of this dissertation, the Normalized Mean Squared Error of a variable x and the estimate $\hat{x}$ is defined as:

$$\text{NMSE}(x) = \frac{1}{N} \frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{\max_i(x) - \min_i(x)} \tag{4.1}$$

Where N is the number of samples over which the NMSE is caluclated. Note that there are multiple ways to normalize the mean squared error. In the case the variable is a feature vector $\overrightarrow{X}$ of length K, the (N)MSE per component X_i is calculated. The final actual (N)MSE is then the mean across all features, ie.:

$$\text{NMSE}(\overrightarrow{X}) = \frac{1}{K} \sum_{i=1}^{K} \text{NMSE}(X_i) \tag{4.2}$$

The NMSE is a number between 0 and 1. Note that the NMSE puts more emphasis on large deviations. A low NMSE score means the model performs well in both the spatial and time dimensions. Meaning a high NMSE score does not necessarily mean that the model doesn't perform well. The model might actually be able to accurately predict the applied force, but might still produce high NMSE scores because of time-related side effects such as hysteresis.

4.2.2 Root Mean Squared Error (RMSE)

The Root Mean Squared Error of a variable x and the estimate $\hat{x}$ is defined as:

$$\text{RMSE}(x) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - \hat{x}_i)^2} \tag{4.3}$$

Where N is the number of samples over which the RMSE is caluclated. In the case the variable is a feature vector $\overrightarrow{X}$ of length K, the RMSE per component X_i is calculated. The final actual RMSE is then the mean across all features, ie.:

$$\text{RMSE}(\overrightarrow{X}) = \frac{1}{K} \sum_{i=1}^{K} \text{RMSE}(X_i) \tag{4.4}$$

In contrast to the NMSE, the RMSE is not normalized and has the unit of the target variable, in this case Newtons. Just like with the NMSE, greater emphasis is put on the large errors. But, because of the absence of normalization, the RMSE can be used to compare the performance of two models or against some reference.

4.2.3 Coefficient of determination R^2

The coefficient of determination, R^2, represents the proportion of variance (of the model output) that has been explained by the independent model variables. It provides an indication of how well unseen samples are likely predicted by the model, through the proportion of explained variance. It is an extension of the explained variance metric with the difference that the explained variance metric does not take bias errors into account, while R^2 does. The score is calculated as follows [23]:

$$R^2(y, \hat{y}) = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4.5}$$

Where $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$, y is the actual applied force and $\hat{y}$ is the prediction from the model, over n samples. When y is multidimensional, i.e., the model predicts vectors instead of real values, the final R^2 score is the average across all outputs.

The value of R^2 can be anything from 1.0 (a perfect fit) to 0.0 (model predicts the average $\bar{y}$) to even $-\infty$, because a predictor can be made arbitrarily worse. Note that the R^2 metric alone is not enough to evaluate the performance of a fit. Indeed, when the model is too complex, it will tend to overfit, producing high R^2 values but not being performant in an actual real-world application.

4.3 Stage I

The goal of the first stage is to convert relative magnetic field measurements on a taxel basis to the force applied to it. Mathematically speaking, we want a model that estimates the applied forces $F_{i,x}$, $F_{i,y}$ and $F_{i,z}$ on taxel i using its measurements x_i, y_i and z_i. The calibration and training procedure is laid out in Chapter 3.

4.3.1 Linear regression

As described in Section 3.4, the model proposed was a linear regressor after a preprocessing pipeline which includes feature transformations and polynomial feature extraction. The supervised linear regressor does not have any hyperparameters, which is why only a training / test split was used and the metrics shown in this section are based on the test set, which encompasses about 20%. The degree n of the polynomial feature extractor was chosen to be 1, 3 or 5. A total of 32 models (one per taxel) were fitted.

Fitting scores and NMSE

In table 4.1 the fitting scores (coefficients of determination R^2) and NMSE values can be seen for all 32 models over all polynomial degrees. The mean and standard deviations across all taxels are also given. The results of the

model from taxel 0 can be seen in Figure 4.3 for visual inspection of the fit.

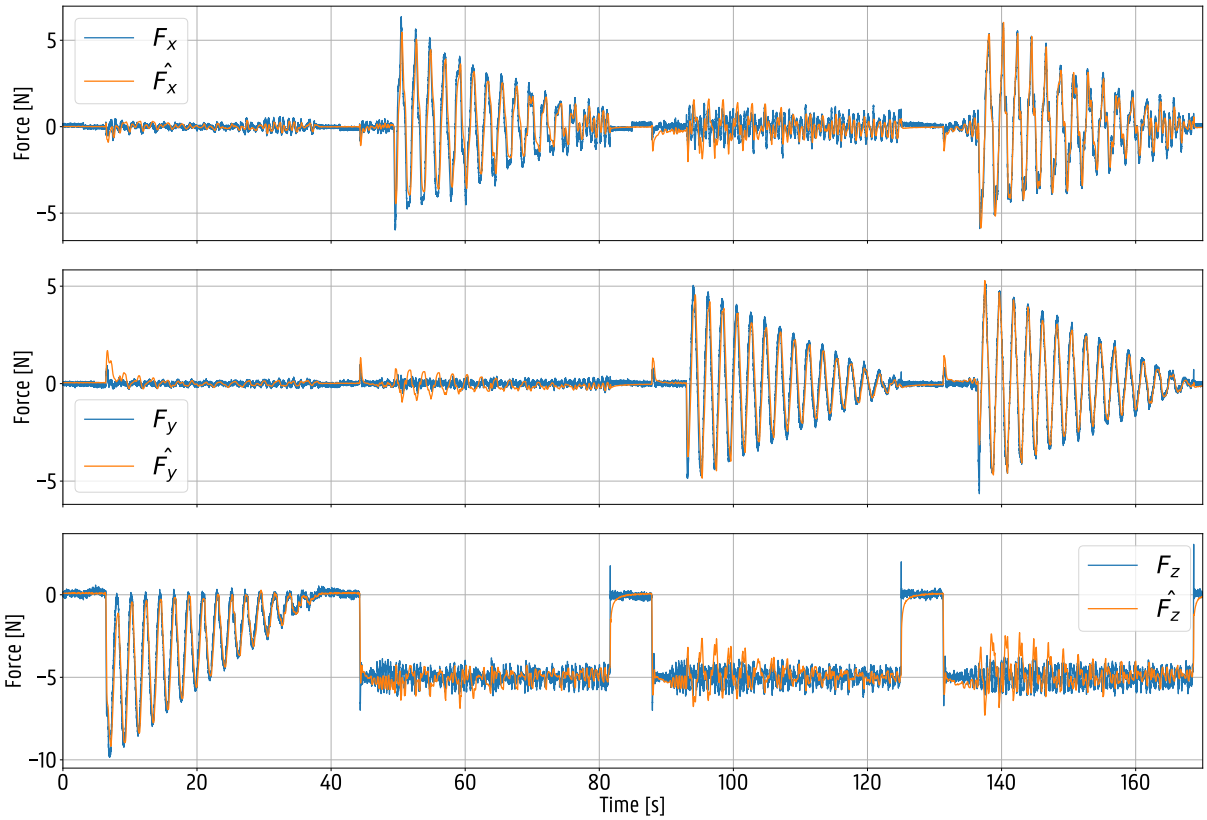


Figure 4.3: The actual applied forces (blue) and predicted forces (orange) of taxel 0. The model was trained on a 80/20% split using a linear regressor with third-order polynomial features

From these metrics it is clear that the third degree model on average performs the best, with a mean cross-taxel R^2 of 0.88134 and NMSE of 0.02849. Notice how the model of the fifth degree has noticably worse scores and also higher variability across all taxels. Especially the significantly higher model variance indicates that the model fails to generalize, in other words, it tends to overfit on the training data. Inversely, the linear model (degree 1) does not perform that much worse compared to the third order model. Especially considering the fact that a third degree model has $\frac{(3+3)!}{3!3!} * 3 = 60$ parameters compared to only $\frac{(3+1)!}{3!1!} * 3 = 12$ parameters in a linear model. This parameter reduction of 80% can be crucial for performance when integrating this force model in an actual application, where the model is probably going to run on some kind of microcontroller. Furthermore, by improving manufacturing techniques the models might perform even better and the model variance will certainly become smaller.

For stage I of the force model the third degree model was chosen, as put forward in the analytical approach in Section 3.1.

Table 4.1: The fitting scores and NMSE values of the 32 taxels for the models fitted on polynomial degrees of $k = 1, k = 3, k = 5$.

	$k = 1$		$k = 3$		$k = 5$	
	R^2	NMSE	R^2	NMSE	R^2	NMSE
Taxel 0	0.87848	0.04866	0.94732	0.01157	-23.38015	0.63823
Taxel 1	0.90572	0.03300	0.95640	0.00924	-0.20548	0.11800
Taxel 2	0.85287	0.05879	0.95354	0.01111	0.84354	0.04993
Taxel 3	0.89846	0.04086	0.95350	0.01073	0.84786	0.05537
Taxel 4	0.92066	0.03114	0.80363	0.07759	0.72956	0.06135
Taxel 5	0.94265	0.01723	0.95487	0.01319	0.76836	0.13990
Taxel 6	0.89509	0.02894	0.95717	0.01047	-13.65787	0.40517
Taxel 7	0.91667	0.01928	0.95966	0.00889	-13.46541	0.37199
Taxel 8	0.92455	0.02719	0.95449	0.01199	-88.43000	0.71898
Taxel 9	0.93054	0.03106	0.97100	0.00970	0.82200	0.07698
Taxel 10	0.94265	0.01536	0.97821	0.00457	0.47881	0.05108
Taxel 11	0.93972	0.02388	0.96871	0.00866	0.82562	0.05754
Taxel 12	0.67363	0.10427	0.71334	0.07168	0.55657	0.16483
Taxel 13	0.92020	0.02846	0.96717	0.01031	0.80212	0.09653
Taxel 14	0.94508	0.01924	0.97837	0.00487	0.75067	0.13179
Taxel 15	0.87549	0.04318	0.95805	0.01114	0.66167	0.09451
Taxel 16	0.92429	0.02176	0.95734	0.01040	0.77126	0.07853
Taxel 17	0.92367	0.03109	0.96357	0.01201	0.71190	0.20006
Taxel 18	0.95263	0.01865	0.98345	0.00384	0.74406	0.18202
Taxel 19	0.94760	0.01509	0.96986	0.00863	0.85772	0.05956
Taxel 20	0.90123	0.03688	0.95591	0.01243	-15.09137	0.43070
Taxel 21	0.85457	0.06897	0.94533	0.02119	-4.45711	0.18839
Taxel 22	0.95175	0.01722	0.98140	0.00428	0.74216	0.16560
Taxel 23	0.86911	0.03323	0.89755	0.02714	0.84022	0.04052
Taxel 24	0.91153	0.02696	0.83481	0.04005	-4.14018	0.26334
Taxel 25	0.92459	0.01215	0.96929	0.00949	-206.46893	0.29398
Taxel 26	0.94912	0.01586	0.97988	0.00456	0.72298	0.23189
Taxel 27	0.90610	0.02943	0.97048	0.00735	-60.68236	0.50180
Taxel 28	0.89634	0.03629	0.83300	0.05420	-12.24119	0.33008
Taxel 29	0.87784	0.04318	0.94151	0.01486	0.54762	0.12788
Taxel 30	0.87335	0.04398	0.93814	0.01810	0.70805	0.18241
Taxel 31	0.91720	0.02809	0.95766	0.01126	-35.25353	0.34425
Mean	0.90448	0.03279	0.93921	0.01705	-14.46065	0.21416
Std.	0.05013	0.01812	0.05894	0.01798	39.53760	0.17211

4.3.2 The importance of the feature transformer

On of the main steps in the preprocessing pipeline described in Section 3.4 is the feature transformer. In this step the measurements of the Z-axis are inversed:

$$\begin{cases} x_i^{'} & = x_i \\ y_i^{'} & = y_i \\ z_i^{'} & = \frac{1}{z_i} \end{cases}$$

An experiment was conducted in which the data used for training was limited to normal forces with an absolute values less than 6N. The model obtains am R^2 of 0.87837 and an NMSE of 0.02937. A visual inspection of the model on Figure 4.4 shows that it is still able to predict the full range of the applied forces. Then the feature transformer was left out for the calibration of taxel 0, still using a linear regressor with third order polynomial features. Even though the model still fits well with an R^2 of 0.89111 and an NMSE of 0.02680, visual inspection of the model on Figure 4.5 when including the complete range of applied forces clearly shows that the model fails to generalize outside the limited range.

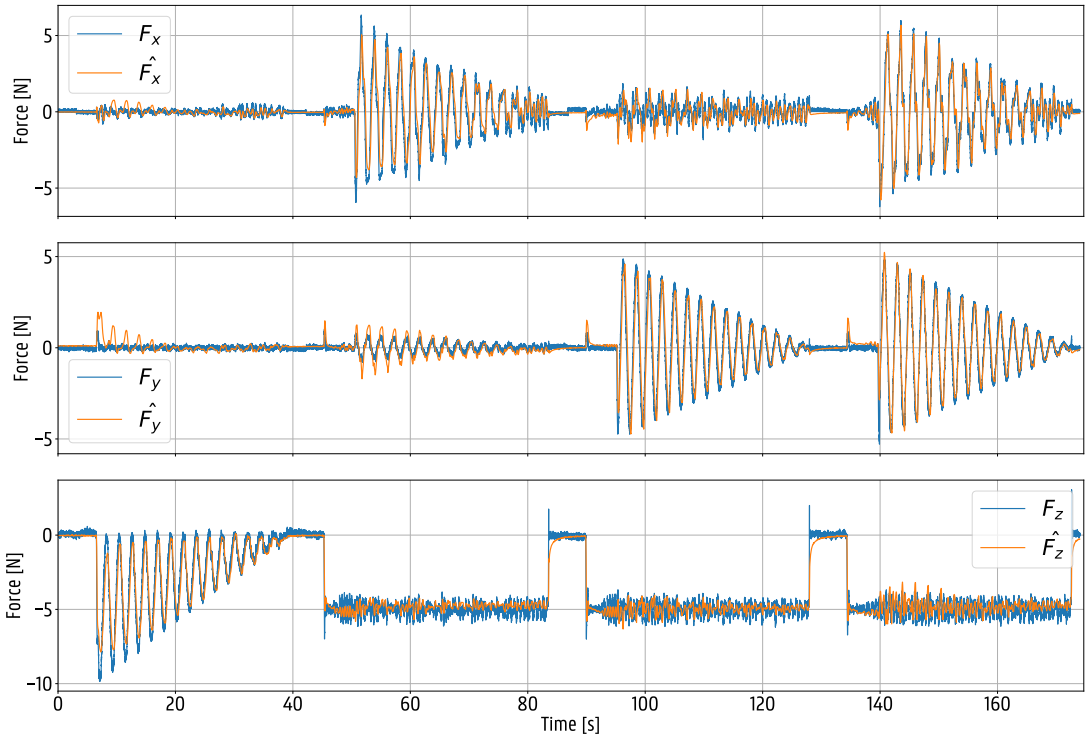


Figure 4.4: The actual applied forces (blue) and predicted forces (orange) of taxel 0. The model was trained on a 80%/20% split with data only containing measurements where the normal force is higher than -6N. The model is a linear regressor with third-order polynomial features and generalizes well.

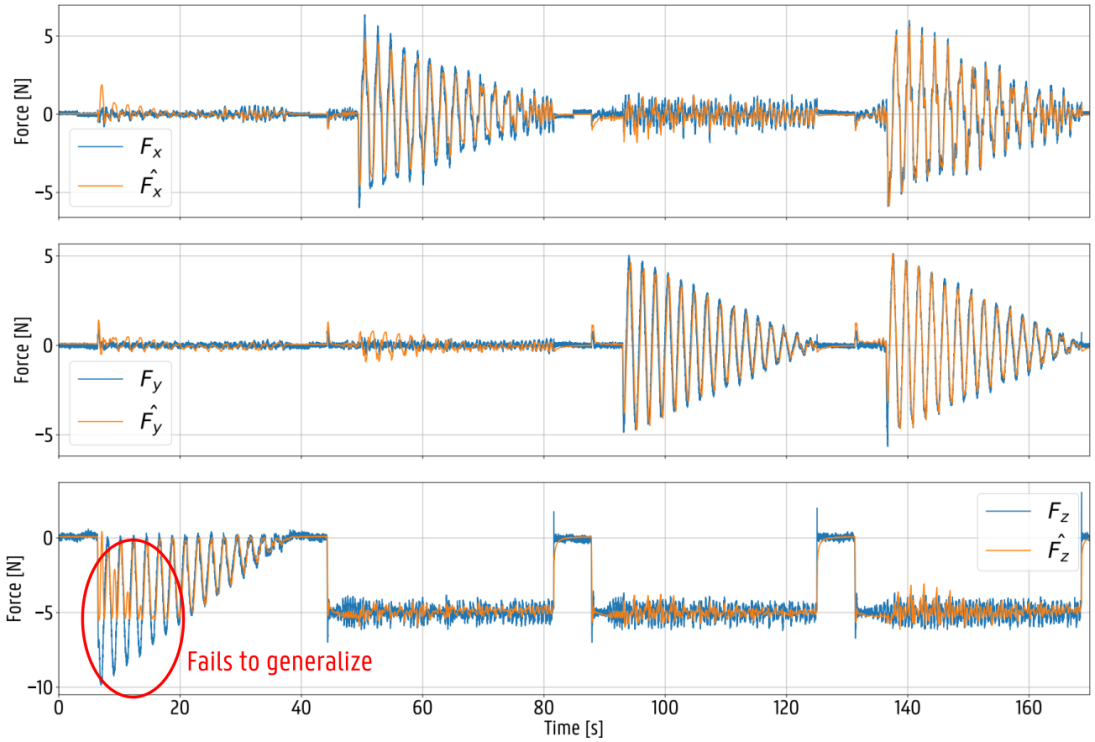


Figure 4.5: The actual applied forces (blue) and predicted forces (orange) of taxel 0. The model was trained on a 80%/20% split with data only containing measurements where the normal force is higher than -6N. The model is a linear regressor with third-order polynomial features. The feature transformer (of the Z-axis) was left out. Note how it fails to generalize below -6N.

This happens because the data structure does not match the prior knowledge used to select a preprocessing pipeline and the regression architecture. This causes the regressor to overfit on the given data instead of learning the relationship between the magnetic measurements and thethe applied forces.

4.3.3 A note on overfitting

When limiting the training data to the samples where one of the planar forces (F_x or F_y) is maximally 2.5N, a result similar to the results in the previous Section can be obtained. As can be seen in Figure 4.6, the model fails to generalize for F_x and F_y. This makes sense, as a third order model is used while it was shown in Section 3.1 that small deviations in the planar direction ρ was linearly proportional to the applied planar force. Thus the third order model is overfitting on the data for the planar directions.

The planar and the normal direction must still be coupled to be able to compensate for magnet misalignments. Therefore, to reduce overfitting the straightforward solution would be to select a second degree or linear model instead, sacrificing some overall accuracy. Possibly a more advanced model can be developed in which some

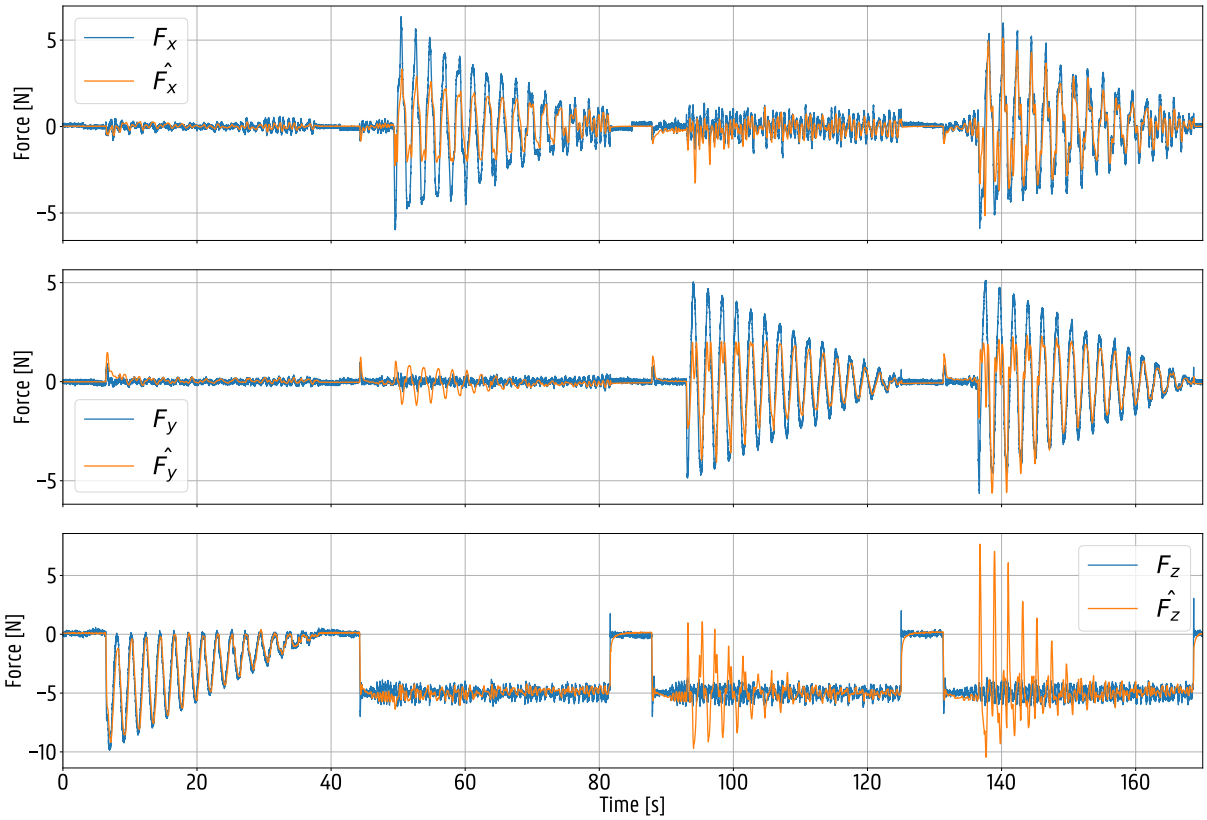


Figure 4.6: The actual applied forces (blue) and predicted forces (orange) of taxel 0. The model was trained on a 80%/20% split with data only containing measurements where the planar forces F_x, F_y are in absolute value smaller than 2.5N The model is a linear regressor with third-order polynomial features.

regularization terms are added to only the higher order coefficients of the planar components. Also, overfitting becomes less of an issue when creating a good all-round dataset that completely overlaps with the application forces.

4.3.4 The spiral calibration procedure

The alternative spiral calibration curve was also tested to see if a shorter procedure where all components are exited at once by executing a kind of spiral movement would also work. The first stage has been trained on all taxels. Similar results are obtained compared to the longer calibration procedure. The first and third order models perform well with average R^2 scores of respectively 0.88387 and 0.92056 and an average NMSE of respectively 0.03519 and 0.02571. The fifth order model again overfits with an average R^2 score of -0.78540 and NMSE of 0.12215. The fit of taxel 0 can be seen in Figure 4.7. Overall, it can be concluded that the spiral calibration curve proposed is actually a viable method of calibrating the taxel.

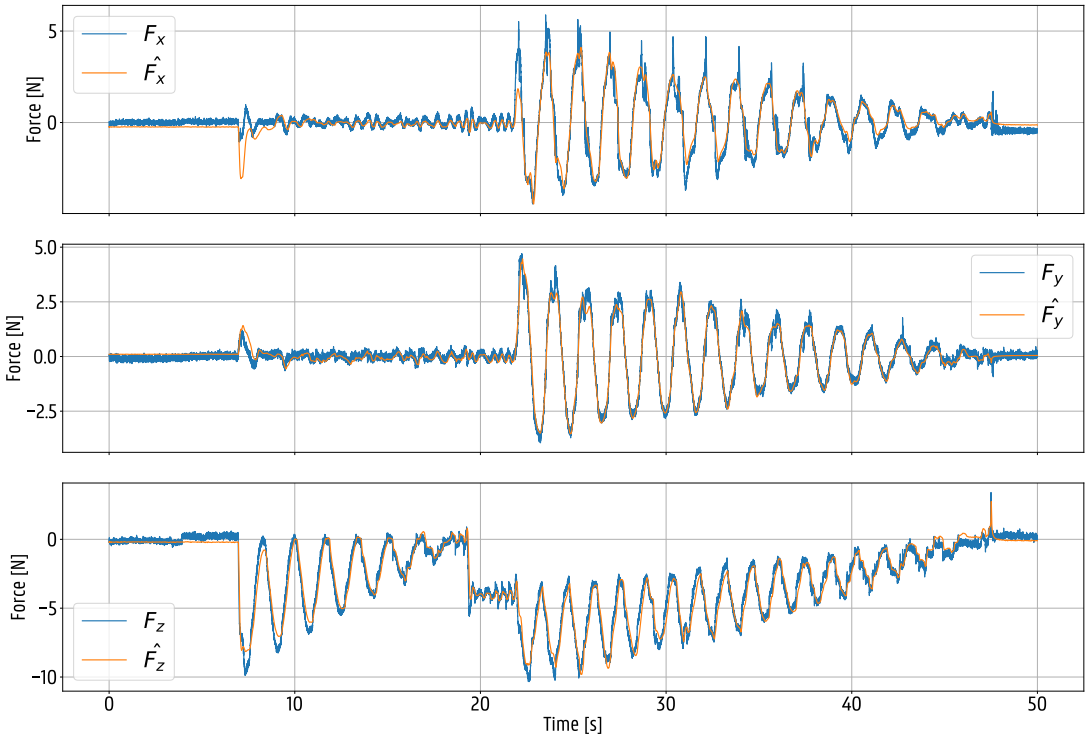


Figure 4.7: The actual applied forces (blue) and predicted forces (orange) of taxel 0 when calibrated using the shorter spiral procecdure. The model was trained on a 80%/20% split dataset. The model is a linear regressor with third-order polynomial features, after some important preprocessing.

4.3.5 Cross-validating the models

The different calibration procedures have different characteristics. As we have data and models for both, it can be interesting to use the data from the other dataset as a second validation. In Figure 4.8 the results can be seen of the original calibration data let loose on both the original model and the alternative models. In Figure 4.9 the same was done for the data from the alternative calibration procedure. In general, it appears both model perform fairly well on the other dataset, with a slight edge for the original calibration procedure.

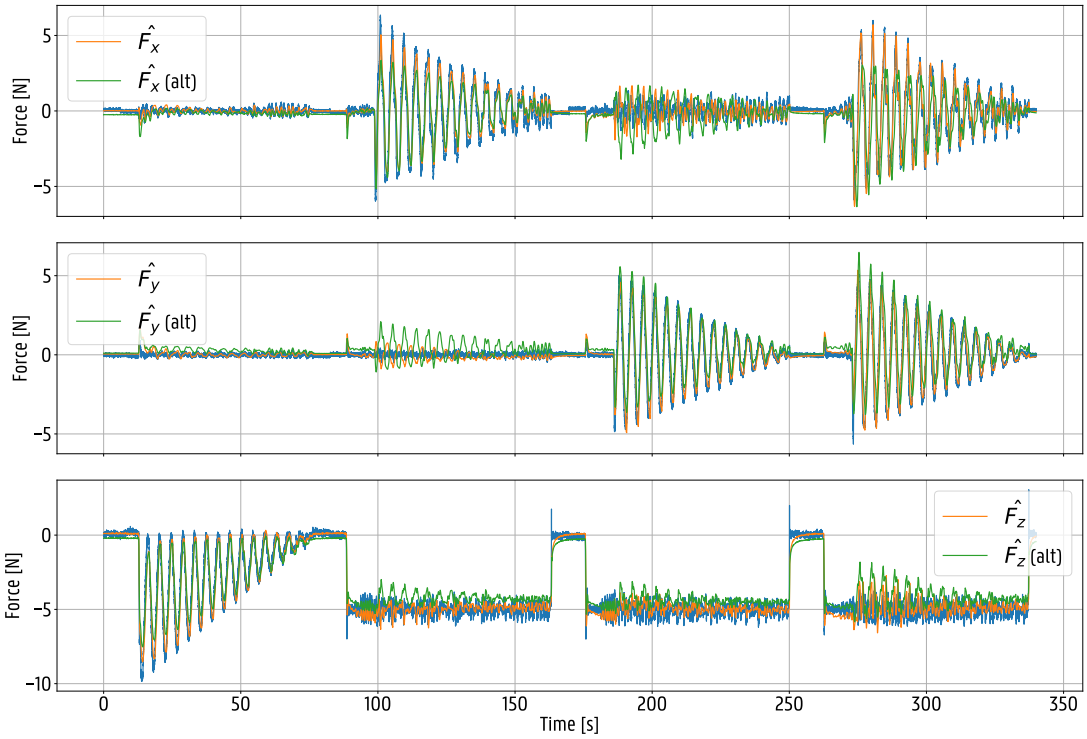


Figure 4.8: The actual applied forces (blue), predicted forces (orange) of the original model of taxel 0 and the predicted forces (green) when using the alternative models, using the shorter spiral procedure. The data used comes from the original calibration procedure.

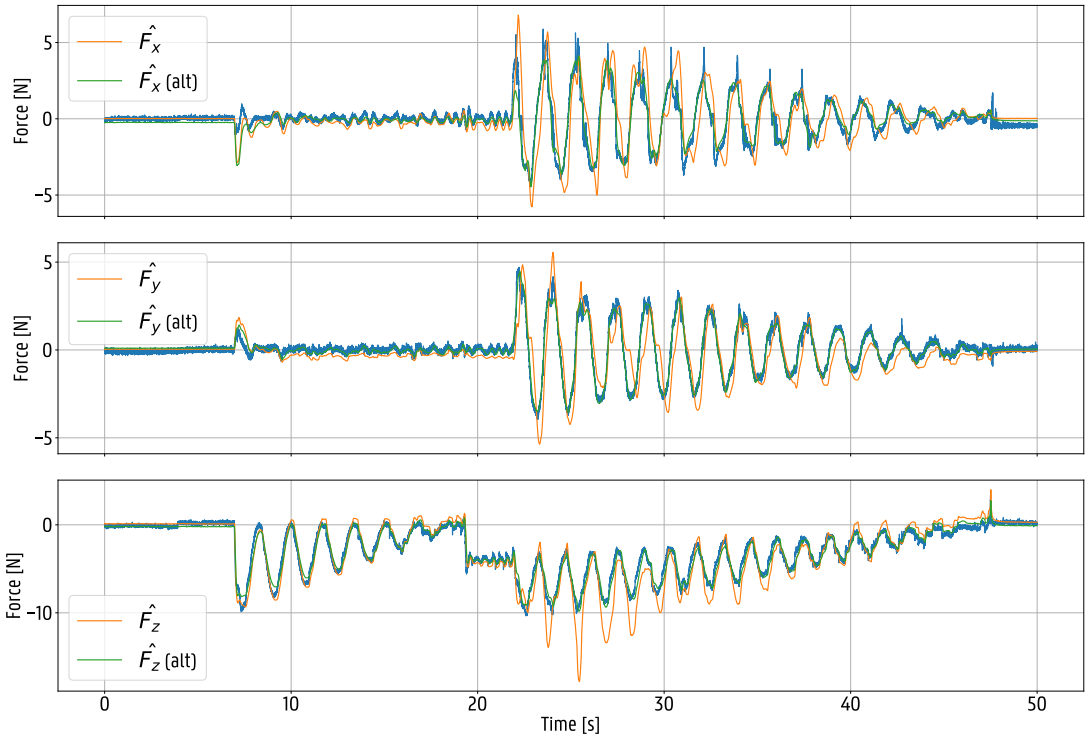


Figure 4.9: The actual applied forces (blue), predicted forces (orange) of the original model of taxel 0 and the predicted forces (green) when using the alternative models, using the shorter spiral procedure. The data used comes from that alternative spiral calibration procedure.

4.4 Stage II

The goal of the second stage is to compensate the predictions in stage I for inter-taxel (magnetic) cross-talk. This is done by finding the matrix $\mathcal{K}$ that models the relation between the predictions of stage I $\mathcal{F}$ and the actual applied force by the robot $\mathcal{G}$, also the output of stage II:

$$\mathcal{G} = \mathcal{K}^{-1}\mathcal{F} \tag{4.6}$$

Evaluation

In order to evaluate the performance of the second stage, the RMSE between $\mathcal{G}$ and $\mathcal{G}_{\text{ref}}$ is calculated. Here, $\mathcal{G}_{\text{ref}}$ is the same as $\mathcal{G}$, but with the components where no force was applied (according to the input data used) set to zero. This evaluation is performed for each dataset (of stage I) separately, as well as for all datasets combined (as if force is applied to all taxels). The same will be done between $\mathcal{F}$ (with compensated bias) and $\mathcal{G}_{\text{ref}}$

4.4.1 Linear cross-talk

First we begin with the linear cross-talk assumption where the matrix $\mathcal{K}$ is invertible. An example of the before and after result where taxel 14 was excited and the cross-talk in taxel 13 is given in figure 4.10.

Visualising the matrix $\mathcal{K}^{-1}$

The matrix $\mathcal{K}^{-1}$ can be visualized using a colormap. The parts of the matrix responsible for the calculation of $\mathcal{G}$ of taxel 14 can be seen in Figure 4.11. As an example, the plot titled $X- > Y$ means that this represents the influence of the uncompensated prediction F_x of the taxels on the Y component of the compensated prediction, G_Y of taxel 14. Notice how the Z-component of a force has a negligible influence on their neighbours.

RMSE

The resulting before and after RMSE values can be seen in table 4.2. The total average RMSE (which equals the power in this case) of the actual neighbours across all datasets went down from 0.041153 to 0.013316.

Table 4.2: The RMSE values of the second Stage for all taxels. Overall, the RMSE (or the power) of the neighbouring signals halves.

	Before compensation				After compensation			
	$RMSE_x$	$RMSE_y$	$RMSE_z$	RMSE	$RMSE_x$	$RMSE_y$	$RMSE_z$	RMSE
Taxel 0	0.01991	0.01349	0.02619	0.01986	0.00706	0.00433	0.01440	0.00860
Taxel 1	0.03282	0.02263	0.03435	0.02993	0.01747	0.01311	0.00966	0.01341
Taxel 2	0.04570	0.02097	0.09666	0.05444	0.01143	0.00829	0.02465	0.01479
Taxel 3	0.00794	0.01032	0.01416	0.01081	0.00220	0.00286	0.00640	0.00382
Taxel 4	0.03224	0.03999	0.08104	0.05109	0.01492	0.02929	0.05161	0.03194
Taxel 5	0.02763	0.04531	0.05733	0.04342	0.00989	0.01238	0.02341	0.01523
Taxel 6	0.04433	0.04953	0.06165	0.05184	0.01093	0.01105	0.02069	0.01423
Taxel 7	0.01677	0.03000	0.03611	0.02762	0.00611	0.00877	0.01446	0.00978
Taxel 8	0.07082	0.05431	0.06372	0.06295	0.01694	0.01469	0.01745	0.01636
Taxel 9	0.06232	0.05344	0.08573	0.06716	0.01192	0.01248	0.02260	0.01567
Taxel 10	0.05620	0.03112	0.05662	0.04798	0.01243	0.01239	0.01948	0.01477
Taxel 11	0.03275	0.02386	0.03937	0.03199	0.00642	0.00627	0.01437	0.00902
Taxel 12	0.02727	0.02689	0.03951	0.03122	0.01489	0.01363	0.02940	0.01931
Taxel 13	0.04558	0.05205	0.05592	0.05118	0.01629	0.01217	0.01740	0.01529
Taxel 14	0.05459	0.04901	0.06037	0.05466	0.00904	0.01027	0.01712	0.01215
Taxel 15	0.05459	0.03262	0.04836	0.04519	0.01337	0.00899	0.01238	0.01158
Taxel 16	0.04627	0.05061	0.04533	0.04740	0.01099	0.01034	0.01349	0.01161
Taxel 17	0.05672	0.04719	0.04437	0.04943	0.01602	0.01349	0.02275	0.01742
Taxel 18	0.06323	0.04408	0.06737	0.05823	0.01114	0.00751	0.01599	0.01155
Taxel 19	0.01668	0.02619	0.04774	0.03020	0.00738	0.00686	0.01543	0.00989
Taxel 20	0.02502	0.02012	0.02727	0.02414	0.00618	0.00463	0.00715	0.00599
Taxel 21	0.04825	0.03745	0.04372	0.04314	0.01009	0.00757	0.01580	0.01115
Taxel 22	0.05374	0.02519	0.06183	0.04692	0.01188	0.00692	0.02065	0.01315
Taxel 23	0.06299	0.03167	0.05454	0.04973	0.01835	0.00722	0.01833	0.01463
Taxel 24	0.02467	0.02379	0.08496	0.04448	0.01272	0.01370	0.05825	0.02822
Taxel 25	0.04435	0.03255	0.05819	0.04503	0.00841	0.00840	0.01965	0.01215
Taxel 26	0.05608	0.03326	0.07555	0.05496	0.01270	0.00943	0.02752	0.01655
Taxel 27	0.03163	0.04009	0.08141	0.05104	0.00788	0.01030	0.01680	0.01166
Taxel 28	0.01115	0.01698	0.01292	0.01368	0.00676	0.00605	0.00915	0.00732
Taxel 29	0.03095	0.02016	0.03967	0.03026	0.01042	0.00652	0.01613	0.01102
Taxel 30	0.02079	0.02918	0.03758	0.02919	0.00876	0.01124	0.01970	0.01323
Taxel 31	0.01071	0.01474	0.02770	0.01772	0.00334	0.00298	0.00755	0.00462
Mean	0.03858	0.03278	0.05210	0.041153	0.01076	0.00982	0.01937	0.013316
Std. Dev.	0.01745	0.01260	0.02053	0.01431	0.00398	0.00473	0.01071	0.00561

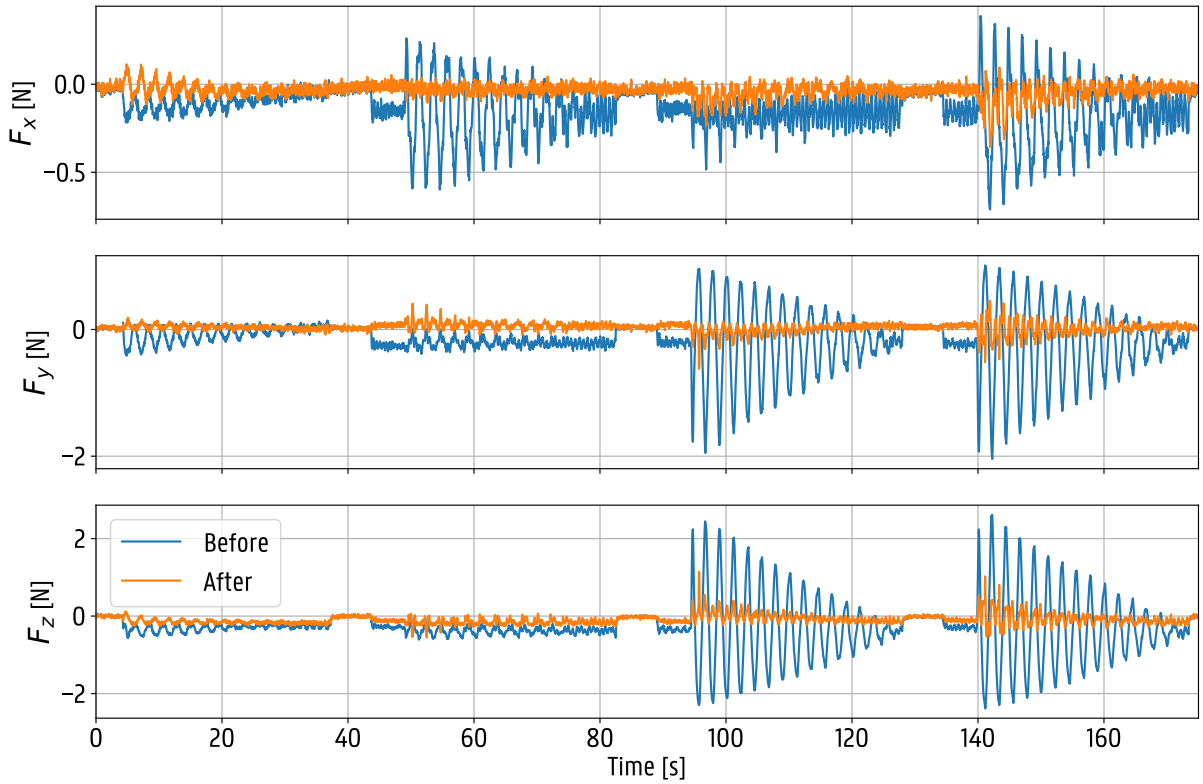


Figure 4.10: An example of the second stage in action. Taxel 14 was excited (using the calibration data) and the output of Stage I of taxel 13 is plotted here. The blue line is the output of stage I, while the orange line is the output after crosstalk compensation.

4.5 Hysteresis analysis

Even after compensating for lag hysteresis caused by the latency difference between the F/T-sensor and the sensor array, some hysteresis still exists. In Figure 4.12, 4.13 and 4.14, respectively, the X, Y and Z components of the predicted force on taxel 0 after the second Stage $\overrightarrow{G_0}$ are shown versus the corresponding applied forces $\overrightarrow{F_x}$, $\overrightarrow{F_y}$ and $\overrightarrow{F_z}$. The plots were created on their respective calibration step. For example, Figure 4.14 was plotted using the first step in which only a normal force was applied, with similar cases for the other two plots.

Note how these look much better compared to Figure **??**. This is not because the model somehow trained how to compensate hysteresis. That would not be possible, since the two-stage force model ignores the time dimension, therefore having no memory and thus is unable to compensate for memory-related effects.

Initially, this thesis was going to look into how to compensate hysteresis using Long Short-Term Memory (LSTM) networks, a type of Recurrent Neural Networks (RNNs), based on work done in [22]. Because LSTMs are quite difficult to execute on low-level hardware and the hysteresis in the sensor itself seems to be relatively small, the choice was made to not do this.

4.6 Readout performance

In this Section, the readout performance is compared across some microcontrollers and configurations. When utilizing an I2C speed of 400kHz and a serial communication rate of 115200bps, the theoretical achievable datarate should be 81.27 Hz. But from the scaleability analysis model using the schedule proposed in Section 2.3.3 it turns out that the serial communication is the bottleneck, reducing the achievable datarate to 36.2 Hz. The Arduino Nano Every was be able to send measurements at a datarate of only 32.0 Hz. This means the microcontroller has a processing overhead of 3.63 ms.

This is why the two-stage force model was implemented on the Raspberry Pi Pico W instead. The RP2040 can do serial communication at a speed of 400kbps, effectively reducing the serial communication bottleneck to a theoretical upper boundary of 63.7 Hz. The actual achieved datarate for sending raw measurements at this communication rate is 60.6 Hz, resulting in a processing overhead of 0.8ms, which is 4.5 times less compared to the Arduino Nano Every.

Enabling the first Stage of the model with third-order polynomials, including the conversion to unsigned integers, reduces the datarate to 50.0 Hz, which results in a Stage I processing overhead of 3.50 ms. When adding the second Stage, the datarate reduces further to 26.0 FPS, or a processing overhead of 18.5 ms. This makes sense, as the second Stage requires $96 * 96 = 9216$ multiplications and $95 * 96 = 9120$ additions because of the 96x96 matrix. In comparison, the first Stage with third order polynomials only requires 26 multiplications for the creation of the polynomial features, followed by $3 * 20 = 60$ multiplications and $3 * 19 = 57$ additions, per taxel. So in total, 2752 multiplications and 1824 additions.

For completeness the communication overhead was tested when sending over the original float values (4 bytes per axis per taxel) instead of the compressed unsigned integers. The datarate dropped from 26.0 Hz to 24.9 Hz.

Finally, when using the linear model, the total amount of multiplications and additions further reduces to $32 * (3 * 4) = 384$ and $32 * (3 * 3) = 288$ additions respectively. Using a linear model in the first Stage thus improves the performance to a datarate of 59.1 Hz without Stage II and 28.2 Hz with Stage II.

Overall, it can be concluded that the first Stage can be ran on a microcontroller without all too much hassle, while the second Stage is a bit trickier. The crosstalk compensation could be done on the central computer instead, or on a second processor / co-processor. Note that even on the Raspberry Pi Pico W, the serial communication speed is still the main bottleneck. So techniques to offload communication from the main or second processor are valuable, especially when wanting to switch to wireless technologies which include quite a significant amount of extra communication overhead compared to serial communication.

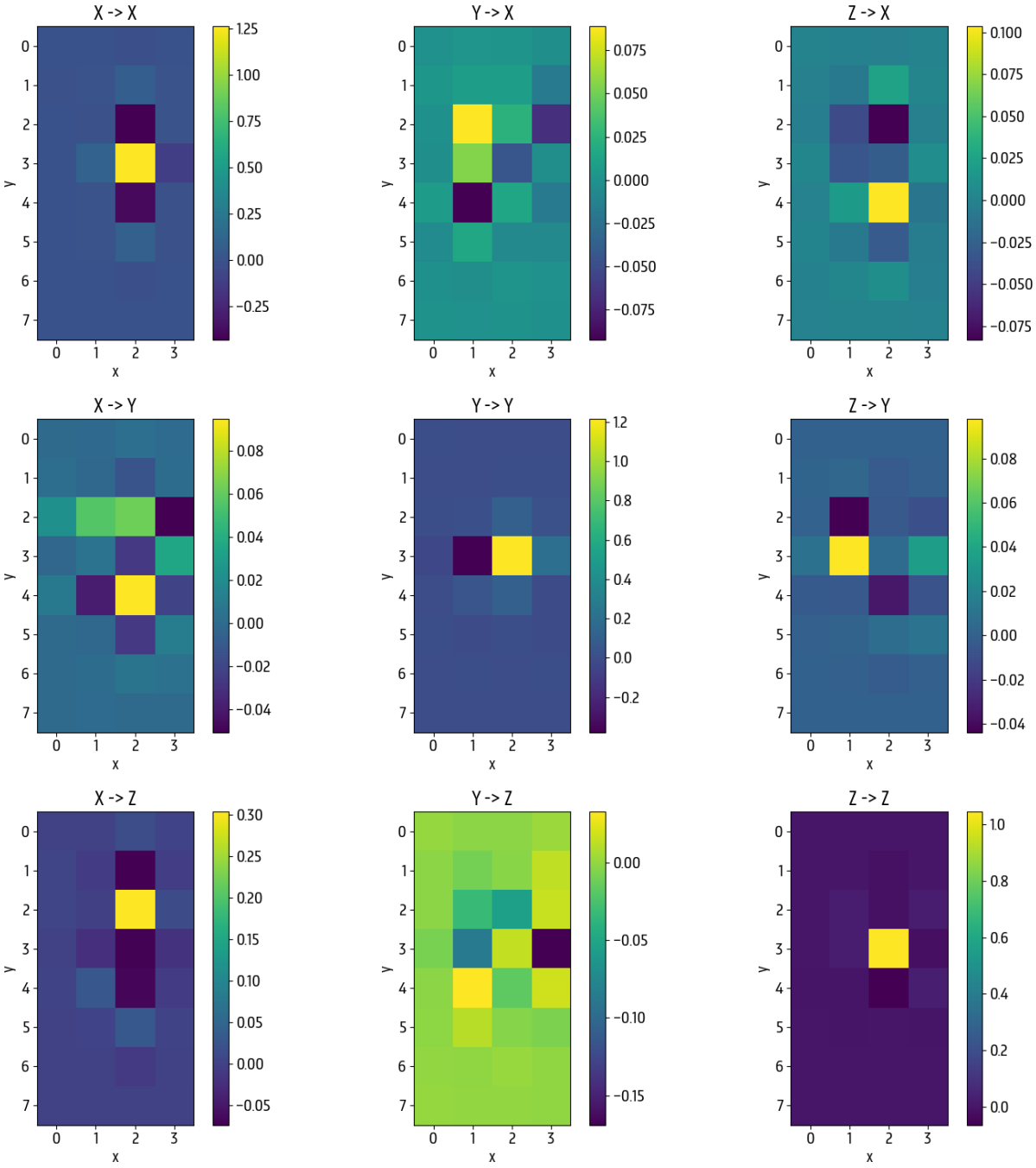


Figure 4.11: A representation of the crosstalk compensation matrix $\mathcal{K}^{-1}$ of taxel 14 (fourth row, third column or coordinate (2,3)). For example, the plot titled $X- > Y$ means that this represents the influence of F_x of the taxels on the Y component of the compensated prediction, G_Y.

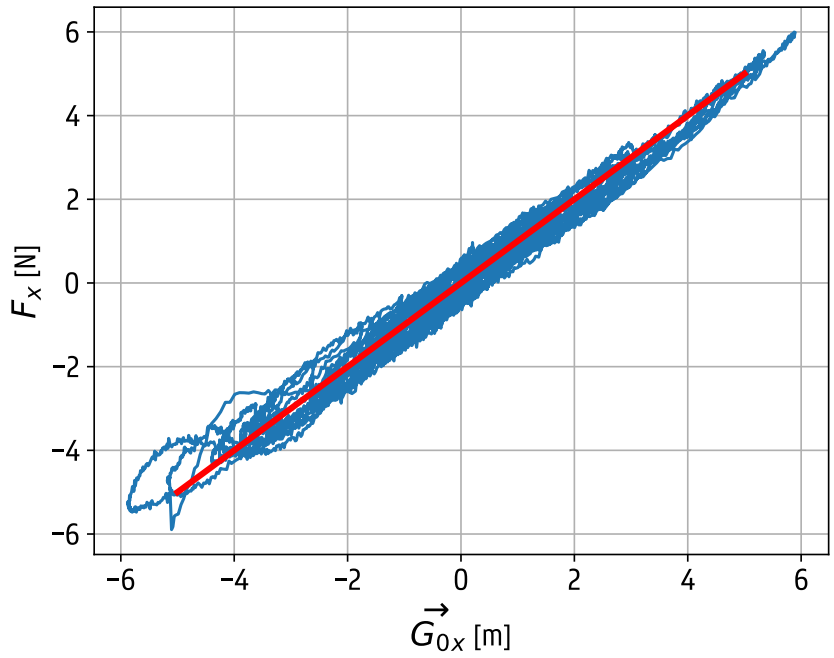


Figure 4.12: The X-component of the predicted force on taxel 0 $\overrightarrow{G_{0x}}$ plotted against the X-component of the applied force, $\overrightarrow{F_x}$. The red line is the ideal case, without hysteresis.

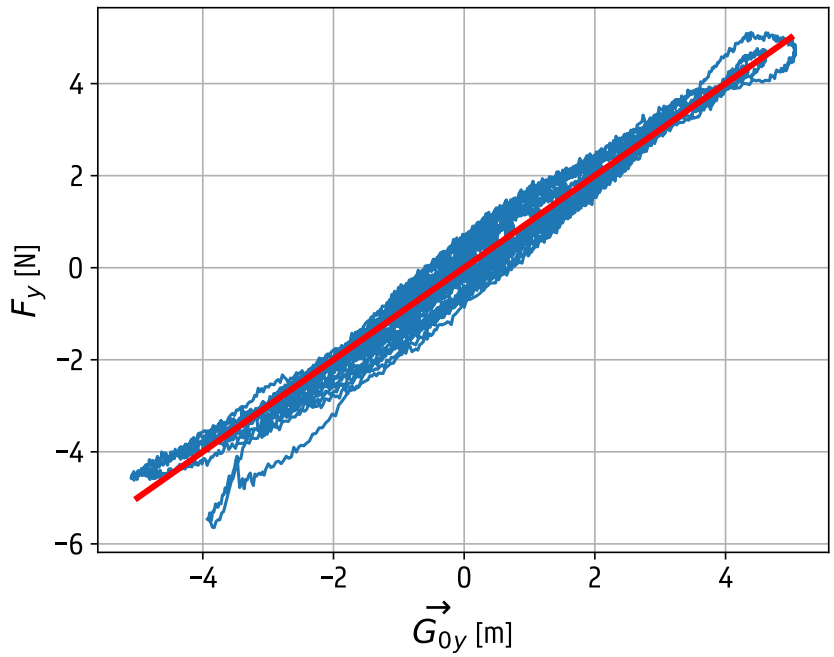


Figure 4.13: The Y-component of the predicted force on taxel 0 $\overrightarrow{G_{0y}}$ plotted against the Y-component of the applied force, $\overrightarrow{F_y}$. The red line is the ideal case, without hysteresis.

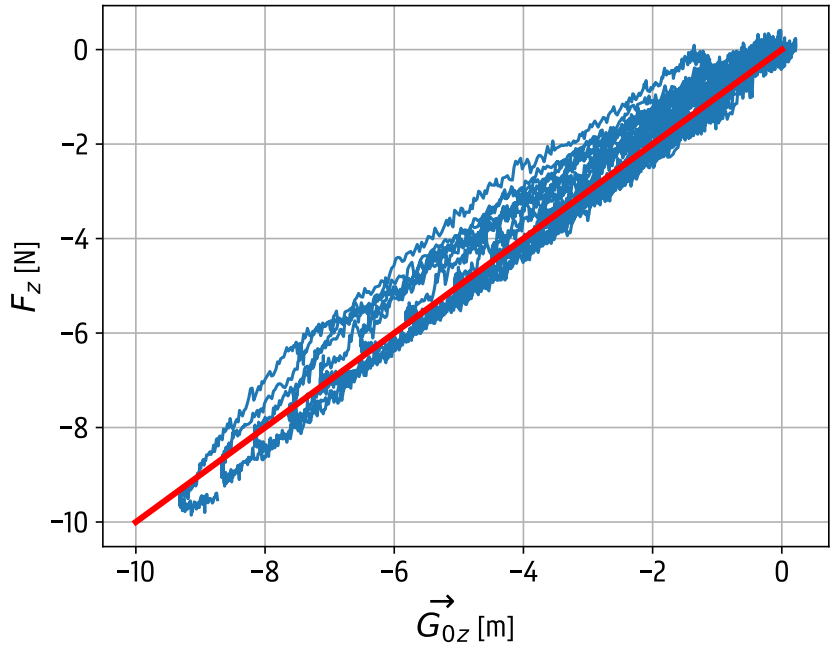


Figure 4.14: The Z-component of the predicted force on taxel 0 $\overrightarrow{G_{0z}}$ plotted against the Z-component of the applied force, $\overrightarrow{F_z}$. The red line is the ideal case, without hysteresis.

Conclusion

In conclusion, this thesis addresses the critical need for advanced tactile sensing in the evolving field of robotics. As robots transition from controlled industrial environments to more dynamic settings where they interact with diverse objects and collaborate with humans, the ability to handle fragile and compliant objects with precision becomes increasingly important. The research presented here focuses on the development of a novel sensor based on magnetic field sensors capable of predicting three-dimensional force vectors on its surface, enhancing the tactile capabilities of robotic grippers such as the Robotiq 2F-85 and Schunk EGU and EGK models.

The key contributions of this work include the construction of the sensor and the creation of an automatic, data-driven calibration method for its two-stage force prediction model. The model was integrated on a Raspberry Pi Pico W, showing potential for actual applications. All resources were also open-sourced on GitHub [1], increasing accessibility to this technology and further incentivizing research.

Ultimately, this research lays the groundwork for future developments in tactile sensing and its applications in robotics using this technology, paving the way for more sophisticated and human-like manipulation capabilities. Further studies and advancements in this area are expected to build on these findings, contributing to the creation of more responsive and adaptable robotic systems.

More specifically, the two-stage force model appears to be promising in its capabilities to predict the force on the taxel, based on measuring the magnetic field differences. However, some issues were discovered in the top layer silicone, reducing the effectiveness of the sensor. Future work should include more advanced mechanical constructions and advancements on the top silicone layer which will help with the overall performance of the sensor. Instead of using a dome structure with a silicone cast top layer, another option would be to combine both structures into one 3D-print. Or instead of using the dome structure principle, more exotic materials that are magnetized could be used as well, like the work done in [12]. Aside from that, the development of a more advanced sensor readout schedule, making use of multiple processors (or other techniques) to eradicate the communication bottleneck, would possibly make wireless communication possible. Finally, this sensor could be used in research focussing on the higher-level algorithms solving tasks like cable grasping, fruit manipulation or cloth folding.

[1] https://github.com/LowiekVDS/masters-thesis

References

[1] Melexis, "Mlx90393 triaxis® micropower magnetometer," https://www.melexis.com/-/media/files/documents/datasheets/mlx90393-datasheet-melexis.pdf, 2023, revision 010, July 2023. [Online]. Available: https://www.melexis.com/-/media/files/documents/datasheets/mlx90393-datasheet-melexis.pdf

[2] D. K. A. Billard, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 364, no. 6446, pp. 1–8, 2019.

[3] R. Yagawa, R. Ishikawa, M. Hamaya, K. Tanaka, A. Hashimoto, and H. Saito, "Learning food picking without food: Fracture anticipation by breaking reusable fragile objects," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 917–923.

[4] L. Zlokapa, Y. Luo, J. Xu, M. Foshey, K. Wu, P. Agrawal, and W. Matusik, "An integrated design pipeline for tactile sensing robotic manipulators," 2022.

[5] E. Judd, B. Aksoy, K. M. Digumarti, H. Shea, and D. Floreano, "Slip anticipation for grasping deformable objects using a soft force sensor," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 10 003–10 008.

[6] R. Proesmans and F. wyffels, "Augmenting off-the-shelf grippers with tactile sensing," 2023.

[7] L. Christiaen, L. V. den Stockt, and W. Goossens, "Laser sharp tactile sensing." [Online]. Available: https://github.ugent.be/wargooss/hop-self-mixing

[8] T. P. Tomo, M. Regoli, A. Schmitz, L. Natale, H. Kristanto, S. Somlor, L. Jamone, G. Metta, and S. Sugano, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2584–2591, 2018.

[9] H. Kristanto, P. Sathe, A. Schmitz, C. Hsu, T. P. Tomo, S. Somlor, and S. Sugano, "Development of a 3-axis human fingertip tactile sensor based on distributed hall effect sensors," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–7.

[10] P. Sathe, A. Schmitz, T. P. Tomo, S. Somlor, S. Funabashi, and S. Shigeki, "Fingertac - an interchangeable and wearable tactile sensor for the fingertips of human and robot hands," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 813–10 820.

[11] K. Dai, X. Wang, A. Rojas, E. Harber, Y. Tian, N. Paiva, J. Gnehm, E. Schindewolf, H. Choset, V. Webster-Wood, and L. Li, "Design of a biomimetic tactile sensor for material classification," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10 774–10 780.

4 References

[12] Y. Yan *et al.*, "Soft magnetic skin for super-resolution tactile sensing with force self-decoupling," *Science Robotics*, vol. 6, no. 51, 2021.

[13] R. Bhirangi, T. Hellebrekers, C. Majidi, and A. Gupta, "Reskin: versatile, replaceable, lasting tactile skins," in *2021 Conference on Robotic Learning (CoRL)*, 2021.

[14] T. Le Signor, N. Dupré, and G. F. Close, "A gradiometric magnetic force sensor immune to stray magnetic fields for robotic hands and grippers," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3070–3076, 2022.

[15] S. Xie, Y. Zhang, M. Jin, C. Li, and Q. Meng, "High sensitivity and wide range soft magnetic tactile sensor based on electromagnetic induction," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 2757–2766, 2021.

[16] L. V. den Stockt, R. Proesmans, and F. wyffels, "Automatic calibration for an open-source magnetic tactile sensor," 2024.

[17] A. C. Holgado, N. Piga, T. P. Tomo, G. Vezzani, A. Schmitz, L. Natale, and S. Sugano, "Magnetic 3-axis soft and sensitive fingertip sensors integration for the icub humanoid robot," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–8.

[18] P. Sathe, A. Schmitz, T. P. Tomo, S. Somlor, S. Funabashi, and S. Shigeki, "Fingertac - an interchangeable and wearable tactile sensor for the fingertips of human and robot hands," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 813–10 820.

[19] F. Tang, J. Yang, H.-N. Li, F. Liu, N. Wang, P. Jia, and Y. Chen, "Field validation of a magnetic sensor to monitor borehole deviation during tunnel excavation," *Materials*, vol. 11, p. 1511, 08 2018.

[20] H. Kristanto, P. Sathe, A. Schmitz, C. Hsu, T. P. Tomo, S. Somlor, and S. Sugano, "Development of a 3-axis human fingertip tactile sensor based on distributed hall effect sensors," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–7.

[21] T. P. Tomo, M. Regoli, A. Schmitz, L. Natale, H. Kristanto, S. Somlor, L. Jamone, G. Metta, and S. Sugano, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2584–2591, 2018.

[22] D. Wu, M. Ourak, K. Niu, Y. Zhang, M. A. Ahmad, J. Dankelman, and E. Vander Poorten, "Towards modeling of hysteresis in robotic catheters based on lstm," 2020.

[23] "3.4. Metrics and scoring: quantifying the quality of predictions — scikit-learn.org," https://scikit-learn.org/stable/modules/model_evaluation.html#r2-score-the-coefficient-of-determination, [Accessed 03-06-2024].

Appendix A

Automatic Calibration for an Open-source Magnetic Tactile Sensor

Lowiek Van den Stockt[1*], Remko Proesmans[1*] and Francis wyffels[1]

***Abstract*—Tactile sensing can enable robots to perform complex, contact-rich tasks. Magnetic sensors offer accurate three-axis force measurements while using affordable materials. Calibrating such a sensor involves either manual data collection, or automated procedures with precise mounting of the sensor relative to an actuator. We present an open-source magnetic tactile sensor with an automatic, in situ, gripper-agnostic calibration method, after which the sensor is immediately ready for use. Our goal is to lower the barrier to entry for tactile sensing, fostering collaboration in robotics. Design files and readout code can be found at https://github.com/LowiekVDS/Open-source-Magnetic-Tactile-Sensor.**

***Index Terms*—Tactile sensing, Sensor-based Control, Open-source**

I. Introduction

Tactile sensing in robotics is crucial for contact-rich tasks like food picking [1] and in-hand manipulation [2]. Widespread sensor modalities include piezoresistive, capacitive, barometric, optical and magnetic sensing. Among these, magnetic sensors can be made with readily available materials and components while providing accurate three-axis force measurements [3]. They work by suspending small magnets [3]–[7], coils [8], or distributed magnetic substrates [9], [10] above a grid of Hall sensors.

Typically, a data-driven calibration procedure is performed to interpret the data from such sensors. In [3]–[5], a human presses the magnetic sensor against a force/torque (F/T) sensor to capture calibration data. Other work proposes automated procedures: [8]–[10] use a robot equipped with an F/T sensor to press down on the magnetic sensor in a predefined grid, whereas [7] has the magnetic sensor pressing down on a load cell mounted to a moving platform.

Schmitz et al. [3]–[5] have commercialised three-axis tactile fingertips based on magnetic sensing, greatly incentivising further research in tactile sensing. However, the monetary cost can still be a hurdle for other groups to adopt these sensors. In the spirit of democratising tactile sensors, we are developing open-source three-axis force sensors: here, we present a baseline fingertip with a 2-by-2 taxel grid. To make adoption in other research groups as straightforward as possible, we propose an automatic calibration procedure with the sensor already mounted to the robot that will make use of it. This way, the sensor is ready for use immediately after calibration, as opposed to previous work [7]–[10].

[1]Lowiek Van den Stockt, Remko Proesmans and Francis wyffels are with the IDLab-AIRO research lab at Ghent University – imec, Technologiepark-Zwijnaarde 126, 9052 Zwijnaarde, Belgium. Remko Proesmans is a predoctoral fellow of the Research Foundation Flanders (FWO) under grant agreement no. 1S15923N. This work was also partially supported by the euROBIn Project (EU grant number 101070596). remko.proesmans@ugent.be

*Lowiek Van den Stockt and Remko Proesmans made equal contributions to this work.

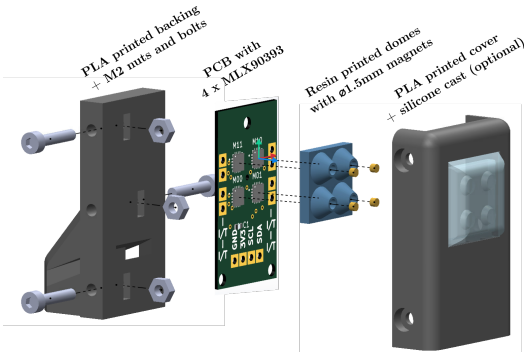


Fig. 1: Fingertip structure. The magnets are glued into the domes, the domes are glued onto the PCB. The PCB is pressed between the cover and backing, and the cover is bolted to the backing.

II. Sensor design

At the heart of the fingertip design, shown in Fig. 1, lies a printed circuit board (PCB) with a 2-by-2 grid of MLX90393 Hall effect sensors connected to an I^2C bus. A dome structure similar to [3], [5] is printed in Flexible FLGR02 resin using a Formlabs Form 2 printer. Cylindrical magnets with a height of $1\,\mathrm{mm}$ and diameter of $1.5\,\mathrm{mm}$ are superglued into the domes, and the dome structure itself is glued to the PCB. A cover for the PCB is 3D printed in PLA using a Prusa i3 MK3. Optionally, a smooth contact surface for grasping can be cast onto the cover using Silicone Addition Colorless 50 by Silicones and More. Fig. 2 shows the mould used. Crucially, a mock PCB is inserted into the cover during curing. The domes on the mock PCB are shorter than the resin domes. This way, when the mock PCB is removed and the resin domes are inserted after curing, an air gap surrounds each dome. The air gap ensures that the force applied to the silicone surface is transferred to the resin domes. Finally, the cover is bolted to a PLA backing (Fig. 1). The backing is designed to fit a Robotiq 2F-85 or 2F-140 gripper, but we also provide a coupling to fit Robotiq fingertips to Schunk EGU and EGK grippers.

III. Automatic Calibration

A. Hardware Setup

The robotic arm on which the gripper is mounted should feature an F/T sensor. In this work, we used a Schunk EGK 40-MB-M-B and a UR3e collaborative arm. A custom probe, see Fig. 3, is mounted externally to the robot. The probe is constructed from an aluminium 1515 profile with D-shaped slots and several 3D-printed PLA parts. The probe tip is shaped such that it fits on top of one of the resin domes.

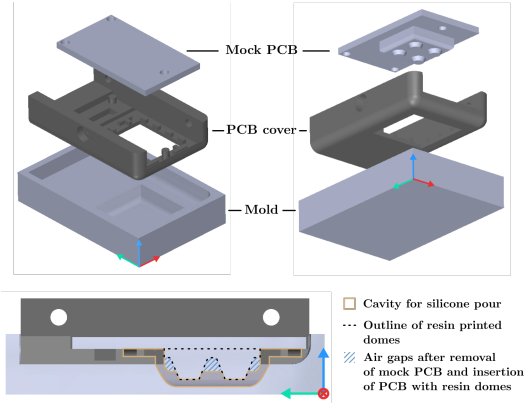


Fig. 2: Mould for silicone pour. The domes on the mock PCB are shorter than the resin-printed domes, such that after removal of the mock PCB and insertion of the PCB with resin domes, an air cavity is present all around each dome.

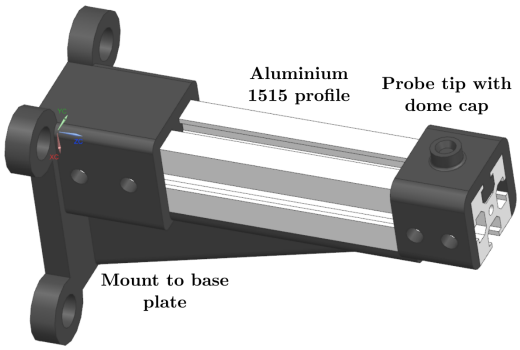


Fig. 3: Calibration probe assembly. The probe tip perfectly fits a taxel dome.

It is important that both the probe tip and the mounting of the probe to the table are rigid.

B. Calibration Methodology

Calibration is done in situ, meaning the fingertip is mounted on the robot that will make use of it. At first, the cover (see Fig. 1) should not have a silicone pad. If desired, the cover can easily be replaced after calibration without removing the fingertip from the gripper. The procedure is gripper-agnostic. Only the direction of the probe tip expressed in the base frame of the robot should be known.

First, the robot is placed in the vicinity of the sensor, with the fingertip facing the probe tip. The robot then touches the probe with the front, the side and the top of the fingertip. Contact is observed from the F/T readings. Knowing the dimensions of the fingertip, the probe can be localised. The robot subsequently presses one of the resin domes into the probe and exerts a predefined set of forces, see the green annotations in Fig. 4, while recording both the F/T and the Hall sensor readings. This is repeated for each taxel. The F/T data is averaged over a moving window of 100 samples or 240 ms. A

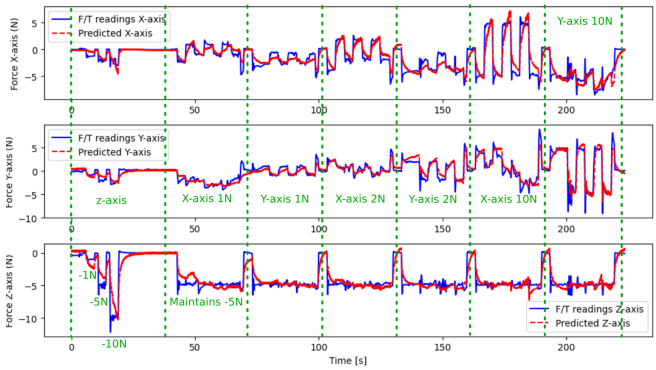


Fig. 4: F/T Readings and model predictions for taxel 3.

linear interpolation is applied to account for different readout frequencies of the Hall and F/T sensors.

Polynomial features up to the third degree are extracted from the Hall readings. An 80%/20% train-test split is randomly sampled and a least squares linear regressor is fitted on the train set. The result is a model per taxel that takes in the X, Y and Z components of the magnetic field and predicts the three components of the force (F_x, F_y, F_z) applied to the taxel.

IV. Results

For each taxel, Table I gives the coefficient of determination (R^2) and the mean squared error (MSE) of the model when applied to the test set. Fig. 4 additionally shows both the true and predicted forces for taxel 3. We note that the predicted force curves rise comparatively slowly to the ground truth. We believe this is due to the mechanical hysteresis of the resin-printed domes.

TABLE I: Test scores for each taxel

Taxel	R^2	MSE [N^2]
0	0.81162	1.10555
1	0.80103	0.97162
2	0.84808	0.88912
3	0.84431	0.92823
Average	0.82626	0.97363

V. Conclusion

We presented an open-source design for a magnetic tactile fingertip and have developed an automatic, gripper-agnostic, in situ calibration strategy. The calibration method only requires a probe to be mounted along a known direction w.r.t. the base frame of the robot. A polynomial model trained on the magnetic sensor data shows an average R^2 of 0.83 and an average MSE of 0.97 N^2. In future work, we will open-source a modular fingertip design featuring tactile surfaces with 4-by-8, 1-by-4, and 2-by-2 taxel grids. In addition, we aim to compensate for hysteresis and to examine a second calibration stage that can account for mechanical and magnetic coupling. Our goal is to lower the barrier to entry of tactile sensors and encourage their adoption across robotics labs.

REFERENCES

[1] R. Yagawa, R. Ishikawa, M. Hamaya, K. Tanaka, A. Hashimoto, and H. Saito, "Learning food picking without food: Fracture anticipation by breaking reusable fragile objects," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 917–923.

[2] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang, "Rotating without seeing: Towards in-hand dexterity through touch," 2023.

[3] T. P. Tomo, M. Regoli, A. Schmitz, L. Natale, H. Kristanto, S. Somlor, L. Jamone, G. Metta, and S. Sugano, "A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2584–2591, 2018.

[4] H. Kristanto, P. Sathe, A. Schmitz, C. Hsu, T. P. Tomo, S. Somlor, and S. Sugano, "Development of a 3-axis human fingertip tactile sensor based on distributed hall effect sensors," in *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, 2019, pp. 1–7.

[5] P. Sathe, A. Schmitz, T. P. Tomo, S. Somlor, S. Funabashi, and S. Shigeki, "Fingertac - an interchangeable and wearable tactile sensor for the fingertips of human and robot hands," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 10 813–10 820.

[6] K. Dai, X. Wang, A. Rojas, E. Harber, Y. Tian, N. Paiva, J. Gnehm, E. Schindewolf, H. Choset, V. Webster-Wood, and L. Li, "Design of a biomimetic tactile sensor for material classification," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10 774–10 780.

[7] T. Le Signor, N. Dupré, and G. F. Close, "A gradiometric magnetic force sensor immune to stray magnetic fields for robotic hands and grippers," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3070–3076, 2022.

[8] S. Xie, Y. Zhang, M. Jin, C. Li, and Q. Meng, "High sensitivity and wide range soft magnetic tactile sensor based on electromagnetic induction," *IEEE Sensors Journal*, vol. 21, no. 3, pp. 2757–2766, 2021.

[9] Y. Yan, Z. Hu, Z. Yang, W. Yuan, C. Song, J. Pan, and Y. Shen, "Soft magnetic skin for super-resolution tactile sensing with force self-decoupling," *Science Robotics*, vol. 6, no. 51, p. eabc8801, 2021.

[10] R. Bhirangi, T. Hellebrekers, C. Majidi, and A. Gupta, "Reskin: versatile, replaceable, lasting tactile skins," in *2021 Conference on Robotic Learning (CoRL)*, 2021.