MULTICOIL READER FOR WIRELESS MEASUREMENTS OF STRAIN

By

PAUL N. FERRI

A thesis submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Mechanical and Aerospace Engineering

Written under the direction of

Aaron D. Mazzeo

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October 2019

ABSTRACT OF THE THESIS

MULTICOIL READER FOR WIRELESS MEASUREMENTS OF STRAIN

by PAUL FERRI

Thesis Director:

Aaron D. Mazzeo

This thesis presents a new multicoil reader to interrogate and locate an embeddable wireless sensor through electromagnetic induction to detect small strain. The passive sensor employs a piezoelectric crystal oscillator with a resonant frequency of ~1.8 MHz to detect strain and a pair of planar coils to communicate wirelessly through inductive coupling. Operating at these low frequencies limits the effective distance of wireless communication to ~2 inches due to the quasi-static electromagnetic field generated by the reader coil. The accuracy of the passive strain sensor decreases when there exists an offset in the relative position between the pair of inductively coupled coils. This work presents a new multicoil reader capable of locating the sensor coil to remove measurement error associated with misaligned coils. The multicoil reader consists of four rectangular reader coils that independently induce electric potentials/currents into the sensing coil and measure the frequency response. From the frequency response, the multicoil reader calculates the mutual inductance between each pair of inductively coupled coils. To locate the sensor coil, this work employs high-fidelity multiphysics simulations to solve the

"forward problem" of determining the mutual inductance between a pair of inductively coupled coils as a function of relative position. From the multiphysics simulations and the kinematics of the multicoil reader, this research creates backward conversion libraries that correlate the location of the sensor coil as a function of the simulated mutual inductances in each coil of the multicoil reader. Using MATLAB scripts, the multicoil reader imports the measured mutual inductances, assembles the backward conversion libraries of simulated data, and searches libraries using Simulink's n-D lookup tables that match the measured mutual inductances to the simulated and the corresponding sensor coil location.

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Aaron Mazzeo, for guiding me and continuously pushing to be better. He has helped me achieve the skills necessary to become a professional researcher.

I would like to thank NASA and Marshall Space Flight Center for providing the funding that made this research possible, and Dr. Patrick V. Hull for his guidance and support throughout the project.

I would like to thank Ajay Dusane along with my fellow graduate students Xiyue Zou, Ramendra Pal, and Tongfen Liang for their support and guidance. I would also like to thank my undergraduate researchers Brendan Dowling and Mark Orzeszko for helping me cultivate this work.

I would also like to mention my doctors at Fox Chase Cancer Center, specifically Dr. Margret von Merhen, Dr. Stephanie Weiss, Dr. Marcin Chistwick, and Kate Murphy. Without your help, I would have never been able to accomplish this.

I would like to thank my family for their constant support throughout my endeavors.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

Figure 4.2.2.1.: (a) Equivalent circuit model for measuring the mutual inductance between two inductively coupled coils, with the reading coil (coil 1) connected to a source and the sensing coil (coil 2) connected to a load. (b) Experimental setup for the one-port reflection measurement to find the self-inductance of a single coil (c) Experimental setup depicting the VNA connected to the two cylindrical coils in the three-port transmission configuration for the first vertical separation of 2 mm.

Figure 4.2.2.4.: (a) Experimental setup duplicated from published reference paper to find the mutual inductance between two inductively coupled coils. (b) COMSOL multiphysics depiction of magnetic flux density within the reader coil. (c) Self-inductance of a single multi-turn coil with an inner diameter of 0.9 mm and containing 11 turns, as to closely match the cross-sectional area of the coil used in the reference experiment. (d) 3-port transmission measurement of the gain between the two coils as a function of relative position using the Omicron Labs Bode™ 100 Vector Network Analyzer. (e) Confirmed the mutual inductance of the fabricated coil with an analytical approximation and the COMSOL results. (f) Comparison of mutual inductance from the reference experiment, reference analytical calculation, our experimental setup, our COMSOL simulation, and our analytical calculation for adjusted geometry.

Figure 4.2.3.1.: (a) The overall configuration of the experimental setup. (b) The experimental setup mounted on an optical table for precise position control. Bode 100 VNA connected to two rectangular planar coils for gain measurement. (c) Coil alignment with respect to one another and VNA leads. (d) Displacement platforms for the x, y and z directions.

Figure 4.2.3.3.: (a) The swept meshing of the rectangular coil. (b) Lumped port analysis of the magnetic flux streamlines at 200 kHz for a position offset of 20 mm in the x-direction, 20 mm in the y-direction, and 10 mm in the z-direction. (c) Magnetic flux density distribution of the reader coil from the top-view for a vertical separation of 5 mm. (d) Magnetic flux density distribution of the reader and sensor coils for a vertical separation of 5 mm.

Figure 4.2.3.4. The plotted results from the experiment, analytical solution, and COMSOL simulation for the rectangular coaxial coil geometry

Figure 4.2.4.1.: (a) COMSOL simulation results compared against experimental results for a vertical separation of 5 mm, x-direction offset of 0 mm, and a y-direction offset ranging from 0 mm to 40 mm in increments of 5 mm. (b) COMSOL simulation results compared against experimental results for a vertical separation of 5 mm, an x-direction offset ranging from 0 mm to 40 mm in increments of 5 mm, and a y-direction offset ranging from 0 mm to 40 mm in increments of 5 mm (c) Comparison of experimental and simulation results for all position at a vertical offset of 5 mm with error bars in white. (d) COMSOL simulation results for all x-y positions ranging from 0 – 40 mm in increments of 5 mm for a range of vertical separations from 5 – 40 mm in increments of 5 mm.

Figure 4.3.1.1.: (a) Experimental setup of the multicoil configuration. (b) X, Y, and Z position control using three displacement platforms. (c) Mutual inductance data formed into libraries to predict the magnitude of the phase, (d) Experimental results for coil 1 at a location in the x-direction of 0, y-direction ranging from 0 – 40 mm in increments of 5 mm, and z-direction of 2 mm.

Figure 4.3.2.1.: (a) Isometric-view of the rotary reader with mounting plate for the single coil (b) Rotary Reader with top cover hidden to depict set of reduction gears that counter rotates the mounted coil at the same rate of rotation. (c) Bottom-view of the rotary reader depicting distance, exactly 47 mm, between the center locations of each coil position. (d) Bottom-view showing the four measurement positions and the order of the measurements.

Figure 4.3.3.1.1.: (a) Plotted interpolated mutual inductance in coil 1, 2, and 3 as a function of the x-coordinate, colormap, using "Scattered Interpolant" for the entire range of displacements. (b) Plotted mutual inductance in coil 1, 2, and 3 as a function of the x-coordinate, (c) y-coordinate, and (d) z-coordinate, colormap, representing the assembled libraries for the modified range. This allowed the n-D lookup tables to locate the sensing coil given measured mutual inductances at the three coil locations.

Figure 4.3.3.1.2. n-D lookup tables ran by importing data from the workspace in MATLAB and run in Simulink. Where m1, m2, m3 are the measured mutual inductances; assembled libraries for M1, M2, and M3 versus x-, y-, and z-directions; and xc, yc, and zc are the corresponding locations of the sensor coil.

Figure 5.1.: (a) Results from simulated experimental data for the 1^{st}, 2^{nd}, and 3^{rd} coil locations using the n-D lookup tables. Depicted the residual less than 0.5 mm from simulated experimental data in the x-direction (b) y-direction, and (c) z-direction.

Figure 5.2.1. (a) Experimental setup for the rotary reader taking impedance measurements at the four locations each 47 mm apart in a square configuration. Error results from measurements taken at coil 1, 2, and 4 for the 13 different offsets in relative position

between the rotary reader and sensor coil in the (b) x-direction, (c) y-direction, and (d) z-directions.

Figure 5.2.1.1.: Implemented the MATLAB script and Simulink file to search and interpolate the location of the sensor coil using three different interpolation techniques for the simulation data; “linear”, “spline”, and “makima”. The results for the three different interpolation techniques used for interpolating the simulation data to locate the sensor coil in the (a)x-direction, (b)y-direction, and (c) z-direction. Discovered the optimal interpolation technique for the “scattered interpolant” of the two techniques; “linear” and “natural”. The results for the two different interpolation techniques used by the “scattered interpolant” to locate the sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction.

Figure 5.2.2. Error results from measurements taken at coil locations L1, L2, and L3 for the 13 different offsets in relative position between the rotary reader and sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction.

Figure 5.2.3. Error results from measurements taken at coil locations L1, L2, and L4 for the 13 different offsets in relative position between the rotary reader and sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction

Figure 5.2.4. Error results from measurements taken at coil locations L4, L2, and L3 for the 13 different offsets in relative position between the rotary reader and sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction

LIST OF TABLES

corresponding directions. Only four absolute errors in the x-, y- and z-directions exceeded 1 mm for the combination of coil locations 1, 2, and 3.

Table 5.2.3. Experimental results from measurements taken at coil locations L4, L2, and L3 for the 13 different offsets, labelled Data Points, in relative position between the rotary reader and sensor coil. Where x, y, and z are the chosen offsets in relative position; C4-, C2, and C3-Phase are the phase amplitudes from the impedance measurement; m4, m2, and m3 are the measured mutual inductances converted from the phase amplitude data; xc, yc, and zc are the locations found using the n-D lookup tables; and Error-x, -y, and -z are the errors in the corresponding directions. Seven absolute errors in the x-, y- and z-directions exceeded 1 mm for the coil locations L4, L2, and L3.

Table 5.2.4. Average of experimental results from measurements combinations of: coil locations L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. Results found the 13 different offsets, labelled Data Points, in relative position between the rotary reader and sensor coil. Where x, y, and z are the chosen offsets in relative position; xc-, yc-, and zc-Mean are the average locations found using the n-D lookup tables for the three different combinations of coil locations; and Error-x, -y, and -z are the errors in the corresponding directions. Five absolute errors in the x-, y- and z-directions exceeded 1 mm for the coil locations 4, 2, and 3.

3. Introduction

3.1. Motivations and Objectives

The objective of this thesis is to present a new multicoil reader to interrogate and locate an embeddable passive wireless strain sensor for long-term structural health monitoring (SHM). The passive sensor employs a piezoelectric crystal oscillator, with a resonant frequency of ~1.8 MHz, for strain sensing and a pair of inductively coupled coils for wireless communication. The piezoelectric oscillators are off-the-shelf crystals sensitive to external physical contact. When a mount holding a crystal experience bending strain, the boundary conditions and physical contact on the oscillating crystal change. This change shifts the resonant frequency of the crystal due to an imbalance in the electromechanical equilibrium. The change in resonant frequency is detectable through inductive coupling at low frequencies by measuring the magnitude and phase of the impedance. Operating at these low frequencies limits the effective distance of wireless communication to ~2 inches due to the quasi-static electromagnetic field generated by the reading coil. The accuracy of the passive strain sensor depends on the relative position between the reader and sensor coils. Misaligned coils, which utilize near-field coupling, observe substantial decreases in mutual inductance and the phase amplitude. The decrease in mutual inductance/phase amplitude negatively affects the resonant frequency/strain measurement. To remove error associated with an offset in the relative position between the pair of inductively coupled coils, this work presents a new multicoil reader capable of interrogating and locating the sensor coil.

This thesis presents a new multicoil reader capable of locating the sensor coil to remove measurement error associated with misaligned coils. The multicoil reader consists of four rectangular reader coils that independently induce electric potentials/currents into the sensing coil and measure the frequency response. From the frequency response, the multicoil reader calculates the mutual inductance between each pair of inductively coupled coils. To locate the sensor coil using the mutual inductance measurements, this work employs high-fidelity multiphysics simulations to solve the "forward problem" of determining the mutual inductance between a pair of coupled coils as a function of relative position. For simplification, the multicoil reader operates with three degrees of freedom in the x-, y-, and z-directions; excluding pitch, roll, and yaw. To solve the "backward problem", this work utilizes the multiphysics simulations of the mutual inductance as a function of relative position to create a backwards conversion that determines the location of the sensor coil as a function of the mutual inductances in three reader coils. This work assembles three libraries of backward conversions for each combination of three reader coils. From the mutual inductance measurements and backward conversions, this work composes a MATLAB script to import the measured mutual inductances, assemble the backward conversion libraries, and search libraries using n-D lookup tables locate the sensor coil.

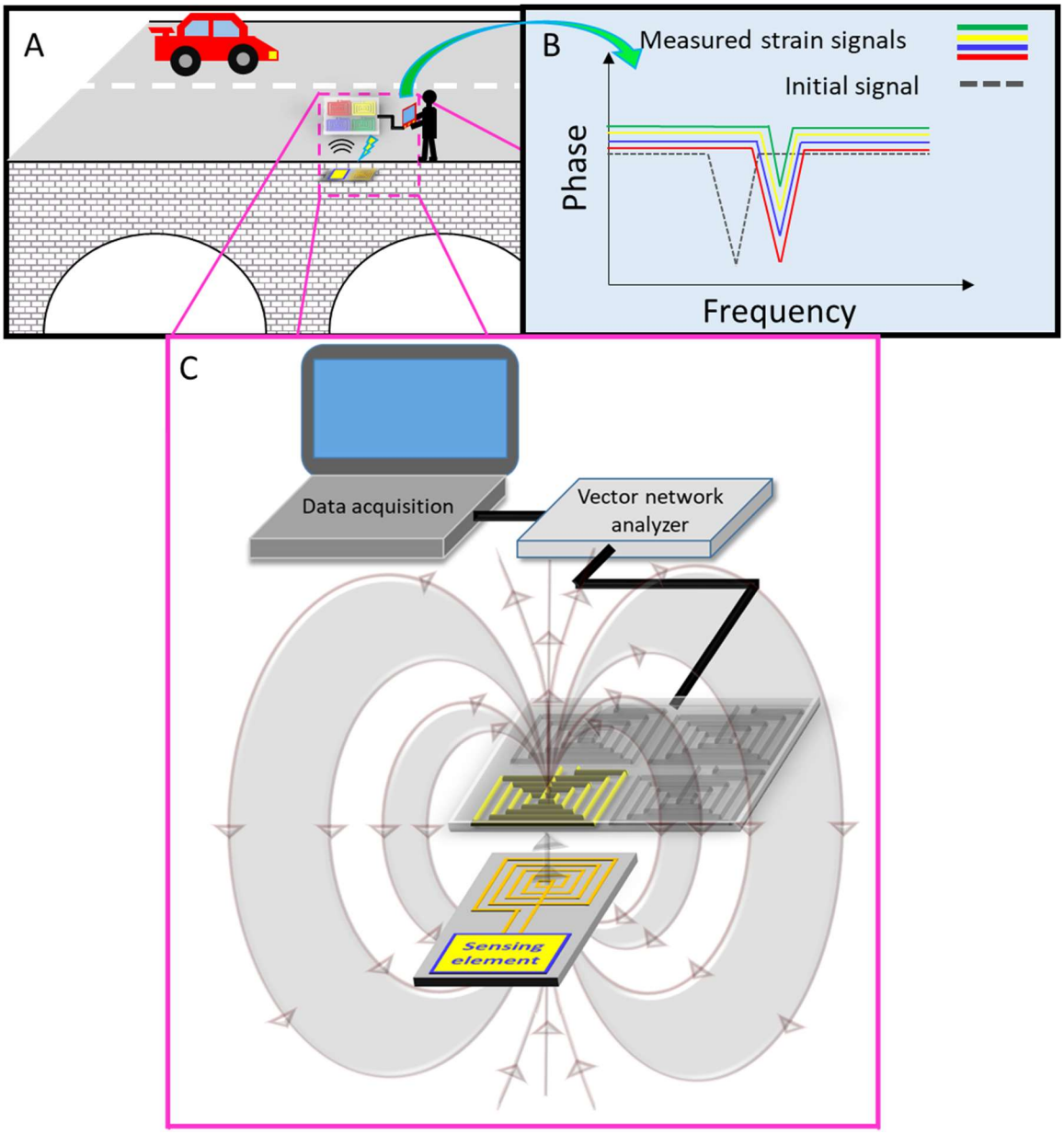


Figure 3.1: Overview of SHM system using passive wireless sensor and multicoil reader. (a) Depiction of a passive sensor embedded into a bridge and an engineer using the multicoil reader to locate the embedded sensor and measure the strain. (b) Measured frequency responses from the piezoelectric sensor using the multicoil reader. (c) Rendering of the multicoil reader consisting of four independent coils transferring power and measuring frequency response one at a time.

3.2. Background

3.2.1. Wired Strain Sensors

Damage due to fatigue, impacts, or defects threaten structures from bridges to spacecraft and can result in catastrophic failures. Structural health monitoring (SHM) is evolving into the primary technique to identify, locate, and characterize damage within a structure or vehicle [1]. To detect damage for structural health monitoring, engineers utilize strain sensors for non-destructive evaluation (NDE) to measure deformation within a structure. Wired strain sensors utilizing photo-elastic, fiber optic, piezoresistive, and resistive methods [2]–[7] measure small strain accurately down to 10s of microstrain; however, they are often expensive, require heavy wiring, and add degrading penetrations.

3.2.2. Wireless Strain Sensors

To avoid the complications associated with wired sensors, engineers turn their attention to wireless sensors to offer an analogous measurement of strain. These sensors either use passive communication with electromagnetic induction or active communication with battery-powered microelectronics. Active sensors use established strain-sensing techniques such as surface acoustic waves (SAW) [8] and acoustic emissions (AE) [9]. The advantage of active sensors is a large effective communication range; however, they require onboard power which limits the lifespan of the sensors.

Passive sensors enable engineers to measure strain using wireless communication and power transfer [10], thereby eliminating the drawbacks and limitations associated with wired and active wireless sensors. There are two categories, based on operating frequency,

of passive wireless sensors: the first is magnetic inductive coupling (MIC), and the second is radio-frequency (RF) coupling.

RF coupling operates from ultra-high to super-high frequencies extending from 300 MHz to 30 GHz, and it has an effective communication distance ranging from two to eight meters. This communication range is because the electric component of the electromagnetic field dominates. To detect strain using RF, researchers use techniques that include patch antennas [11]–[13], SAW [14]–[18], and RF cavities [19]. The large transmission distance and high sensitivity of the RF-based sensors are attractive for passive wireless strain sensing. However, the electric field dominated RF method is susceptible to interference from noise, metallic enclosures, and backscattering [20]. It is also vulnerable to interference from other forms of RF communication and limitations from FCC bandwidth restrictions. Lastly, the RF method requires more power for operation compared to low-power magnetic inductive coupling.

Magnetic inductive coupling (MIC) is an advantageous near-field coupling technique that, unlike RF, experiences little to no interference from external sources. MIC operates in the low to mid-range frequencies ranging from 10 kHz to 3 MHz, which are unassigned for use by the FCC, making them ideal for wireless communication. However, MIC has a smaller effective communication distance than RF, because the magnetic component of the electromagnetic field dominates resulting in a quasi-static magnetic field around the reader coil. To detect strain using MIC, researchers use either an inductive- or capacitive-based sensors that function as a resonant *LC* circuit. The resonant *LC* circuit functions under an applied strain that changes the inductance or capacitance of the sensor resulting in a shift in its fundamental resonant frequency of the *LC* circuit [21]–[26]. The majority of current

inductive- or capacitive-based strain sensors that use MIC for wireless communication measure large strain, in the milli-strain regime.

This paper continues previous work [27] that used low-frequency MIC for wireless communication and a piezoelectric sensing element to measure small-strain. The MIC operates at low-power consumption, has little interference due to other forms of communication, and functions at frequencies undesignated by the FCC when compared to the RF counterpart. However, MIC has a smaller effective communication range ~2 inches that resulted in the relative position between the reader and sensor coils decreasing the magnitude of the frequency response and the accuracy of the strain measurement.

3.2.2.1. Wireless Readers

This issue of an offset in relative position between the reader and sensor coils affecting measurements of wireless sensors is described in other works [21], [22], including sensors that used interdigitated capacitors and resonant *LC* circuits to detect strain. The first case used an interdigitated capacitor [21] for measuring strain and a single loop antenna inductively coupled to a planar sensing coil for wireless communication. This paper reported that an offset in the relative position between the two coupled coils affected the magnitude of the frequency response. However, it did not change the resonant frequency of the sensor because the reader monitored the resonant frequency of a capacitive-based sensor as opposed to inductive-based. In the second case, researchers used a resonant *LC* circuit with solenoidal inductors [22], an offset in the relative position between the reading and sensing coils shifted the resonant frequency of the *LC* circuit. The *LC* circuit shifted its resonant frequency because the mutual inductance between the coils

changed thus changing the overall inductance of the *LC* circuit. Our work resembles the first case, but instead of interdigitated capacitors, we used a piezoelectric crystal oscillator. The resonant frequency of the crystal oscillator, as expected, did not change due to an offset in relative position between the inductively coupled coils. However, similar to the first case, we saw a decrease in the amplitude of the frequency response and in turn the accuracy of the strain measurement. This problem of an offset in the relative position between the two inductively coupled coils makes it advantageous to be able to measure the location of the sensing coil for embeddable applications.

This thesis presents a new multicoil reader capable of locating the sensor coil to remove measurement error associated with misaligned coils. The multicoil reader consists of four rectangular reader coils that independently induce electric potentials/currents into the sensing coil and measure the frequency response. From the frequency response, the multicoil reader calculates the mutual inductance between each pair of inductively coupled coils. To locate the sensor coil using the mutual inductance measurements, this work employs high-fidelity multiphysics simulations to solve the "forward problem" of determining the mutual inductance between a pair of coupled coils as a function of relative position. For simplification, the multicoil reader operates with three degrees of freedom in the x-, y-, and z-directions; excluding pitch, roll, and yaw. To solve the "backward problem," this work utilizes the multiphysics simulations of the mutual inductance as a function of position to create a backward conversion that determines the location of the sensor coil as a function of the mutual inductances in three reader coils. This work assembles three libraries of backward conversions for each combination of three reader coils. From the mutual inductance measurements and backward conversions, this work

employs MATLAB scripts, written by Ajay Dusane and me, to import the measured mutual inductances, assemble the backward conversion libraries, and search libraries using n-D lookup tables locate the sensor coil.

3.3. Thesis Overview

The remainder of this thesis is organized as follows. First, we report the piezoelectric sensor developed to measure small strain, and how a misalignment of the inductively coupled coil affects the accuracy of the strain measurement. To overcome the shortfalls of the wireless interrogator, we developed the multicoil reader to interrogate and locate the sensor coil. To solve the "forward problem", we first validated an experimental procedure and COMSOL simulation for finding the mutual inductance between two inductively coupled coils as a function of relative position by replicating published results for a cylindrical coil. From there, we applied the simulation and experiment to the rectangular coil geometry chosen for the multicoil reader along with an analytical approximation. Satisfied with the validity of the experiment and simulation, we applied the COMSOL simulation to numerically calculate the mutual inductance for a range of position offsets in the x-, y-, and z-directions to solve the "forward problem" represented as $M(x, y, z)$. We solved the "backward problem" by creating three functions of simulated mutual inductance data from three coils representing locations in space and corresponding values at each location for the x-, y-, and z-displacements represented by $X(M_1, M_2, M_3)$, $Y(M_1, M_2, M_3)$, and $Z(M_1, M_2, M_3)$. We assembled the backward conversion libraries using a "scattered interpolant" based on the kinematics of the multicoil reader for the three combinations of three coils: coils 1, 2, and 3; 1, 2, and 4; and 4, 2, and 3. The coil

combination 3, 2, and 1 would yield identical results to coils 1, 2, and 3 as such the remaining combinations are neglected. We then implemented n-D lookup tables in Simulink to search the assembled libraries corresponding locations in the x-, y- and z-directions given measured mutual inductances for three coils. To validate the n-D lookup tables, we used interpolation to simulate experimental data and ran the data through the lookup tables. Satisfied with the accuracy of the n-D lookup tables, we tested the multicoil reader. The multicoil reader failed to function properly due to the overlapping coils, but separating the coils edge-to-edge would reduce the mutual inductance to undetectable levels. So, we designed and tested a rotary reader that used one coil to take measurements at the four locations of the multicoil reader. Lastly, the rotary reader successfully located the sensor coil for 13 different offsets in relative position.

4. Approach to Work

4.1. Piezoelectric Strain Sensor Design

To measure strain, Xiyue Zou and I exploited the ability of the piezoelectric crystal oscillator to convert mechanical vibrations into a detectable electrical signal at resonance. The resonant frequency of the off-the-shelf crystal oscillator is a function of its boundary conditions and geometry. When the length of the crystal oscillator is larger than its thickness, the relationship between its resonant frequency and thickness is as follows:

$$\omega^2 = \frac{c_{66}}{\rho}\left(\frac{\pi}{2b}\right)^2 \tag{1}$$

where ω is the resonant frequency, b is the thickness of pthe late, ρ is the density, and c_{66} is the component of elastic modulus parallel to the plate. The resonant frequency increased at a rate inversely proportional to the thickness. From this relationship, we designed a sensor mount that converted bending strain into external physical contact on the crystal oscillator.

The mount of the bending sensor consists of two acrylic supports in a step configuration on either side of the aluminum baseplate. The step configuration simply supports the piezoelectric crystal leaving it free to oscillate. To apply external contact, we fabricated an aluminum bridge and bonded it to the top of each acrylic support. During this bonding process, we applied the initial preload to the crystal oscillator. The preload allowed a compressive bending strain to remove the external contact, thicken the crystal, and

decrease the resonant frequency, as seen in **Figure 4.1**. Under a tensile bending strain, the external contact thins the crystal oscillator resulting in an increase in resonant frequency.

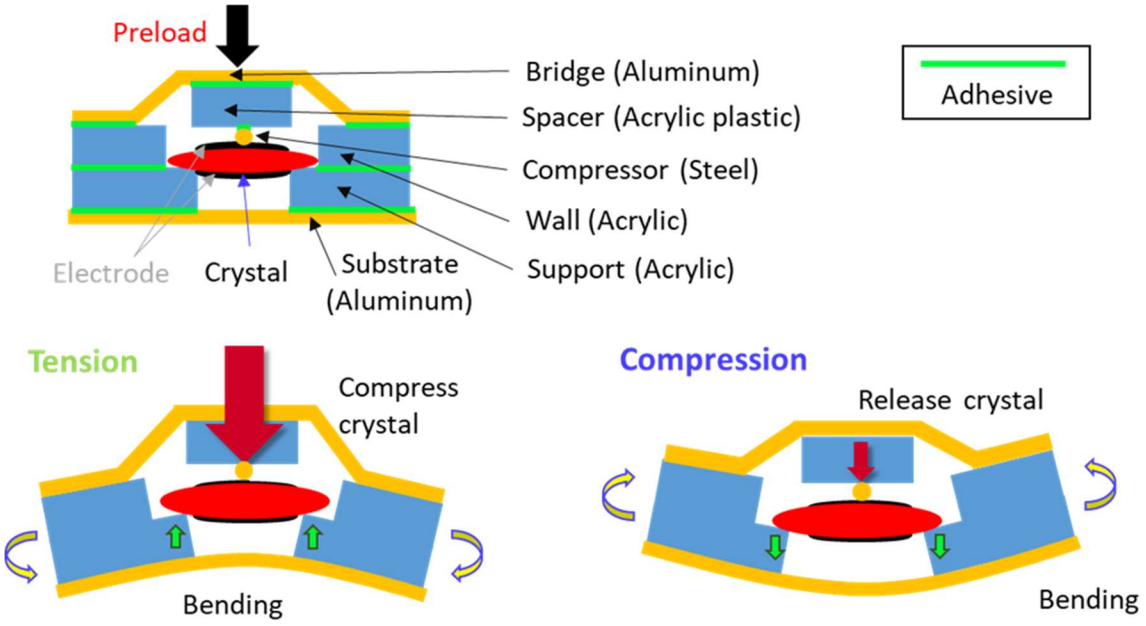


Figure 4.1: (a) Diagram of the piezoelectric strain sensor that we designed to detect bending strain. Depicts the effects of tensile bending strain, which thins the crystal and increases the resonant frequency; and the effects of compressive bending strain, which removes an initial preload decreasing the resonant frequency.

4.1.1. Electromechanical Characterization

The wireless interrogator utilized the magnetic inductive coupling between two rectangular planar coils to transmit a sinusoidal waveform for excitation and reflect the frequency response. The interrogator employed a Bode 100™ Vector Network Analyzer (VNA), which generated the sinusoidal waveform and measured the frequency response of the crystal, by measuring the impedance of the system. The VNA performed an S_{11} one-port reflection measurement that calculated the ratio between the incident and reflected

waveforms to obtain the magnitude and phase of the impedance. The VNA swept a range of frequencies from 1.84 MHz to 1.85 MHz using 401 points, a receiver bandwidth of 300 Hz, and a source power of 13 dBm to reduce the signal to noise ratio. The interrogator exported the impedance data in real-time to a MATLAB GUI, designed by Xiyue Zou, that automatically applied curve fitting, pinpointed the phase minimum, and calculated the resonant frequency of the crystal oscillator. We performed the bending experiment, depicted in **Figure 4.1.1.A,** on the cylindrical compressor sensor design for seven loading cycles ranging from 0 - 30 microstrains by applying weight ranging from 0 – 200 grams in increments of 20 grams. The results for the cylindrical compressor design, displayed in **Figure 4.1.1.B,** show a minimum sensitivity of ~6 με and a frequency versus strain relationship of ~20 Hz/με.

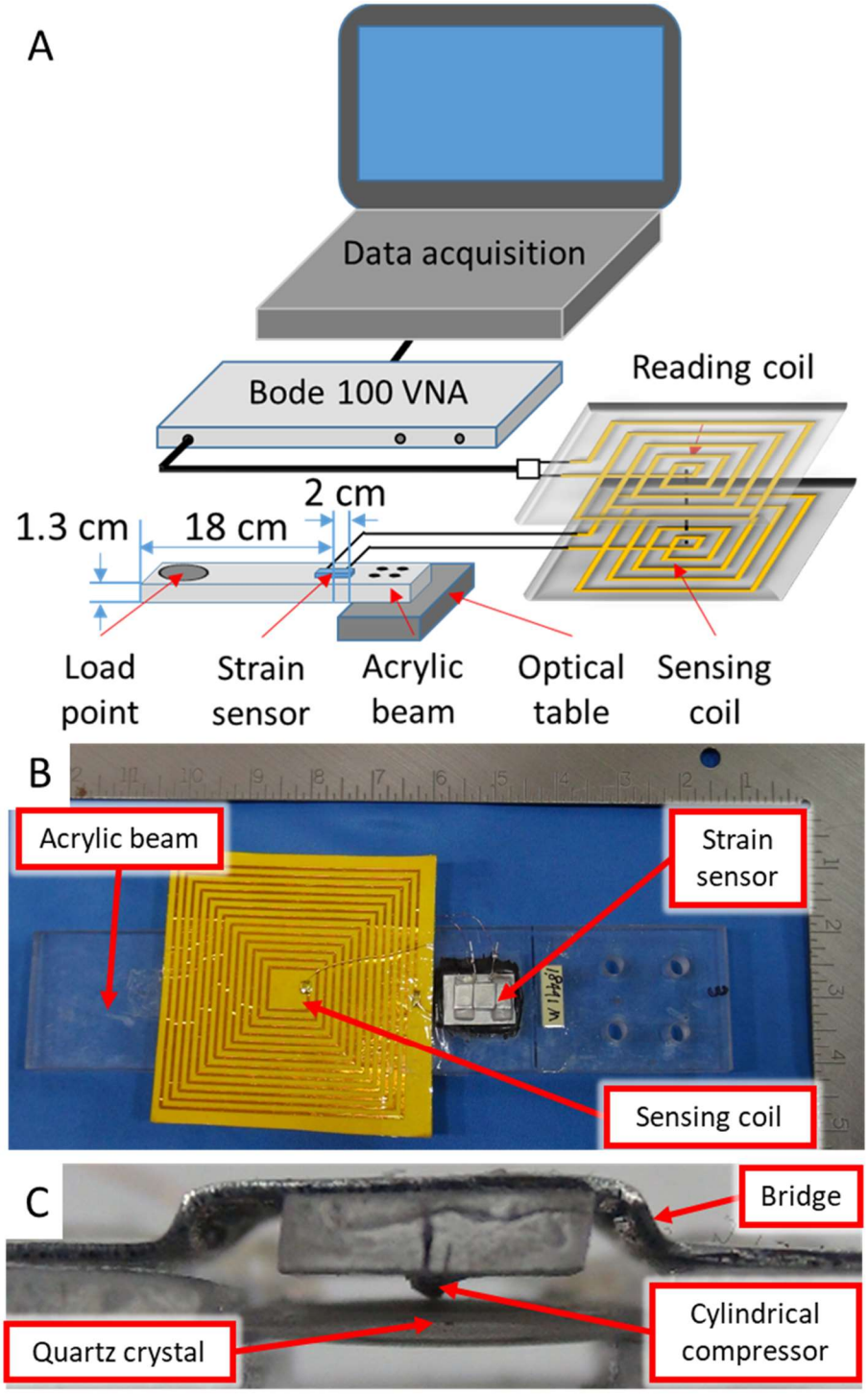


Figure 4.1.1.: (a) Experimental setup for measuring the resonant frequency of the crystal oscillator given an applied weight. Weight ranged from 0 – 200 grams in increments of 20 grams on a ½ inch thick acrylic beam, cantilevered 20 cm, and 5 cm wide. (b) Experimental results for the applied weights ranging from 0 – 200 grams in increments of 20 grams.

4.1.2 Effective Range of Wireless Communication

To find the effective range of wireless communication for the sensor, Xiyue Zou analyzed the influence of a vertical displacement between the reader and sensor coils. **Figure 4.1.2.A** depicts the experimental setup used to test the wireless communication range of the inductively coupled coils for the coaxial case. The setup consisted of the test specimen, with the mounted bending sensor, placed onto a level surface to allow the system to reach equilibrium. Next, we employed ¼ -inch acrylic spacers to apply a vertical displacement between the reader and sensor coils ranging from 0 – 2 inches. To ensure the reader coil remained level, we attached a compression board to the back of the coil. We then interrogated the sensor using the Bode 100 VNA. The VNA swept a range of frequencies from 1.84 MHz to 1.85 MHz using 401 points, a bandwidth of 300 Hz, and a source power of 13 dBm. The source power helped reduce the signal to noise ratio between the two coils.

Figure 4.1.2.B depicts the experimental results for a range of vertical separations, from 0 – 2 inches in increments of 0.25 inches, between the reader and sensor coils. As the vertical separation increased the amplitude of the phase decreased due to the decrease in the mutual inductance between the two coils. Throughout the testing, the resonant frequency of the crystal oscillator remained unchanged. However, as the amplitude of the phase decreased so does the accuracy of the resonant frequency measurement. To remove the error associated with an offset in the relative position between the inductively coupled coils, we designed and built a multicoil reader to locate the position of the sensor coil.

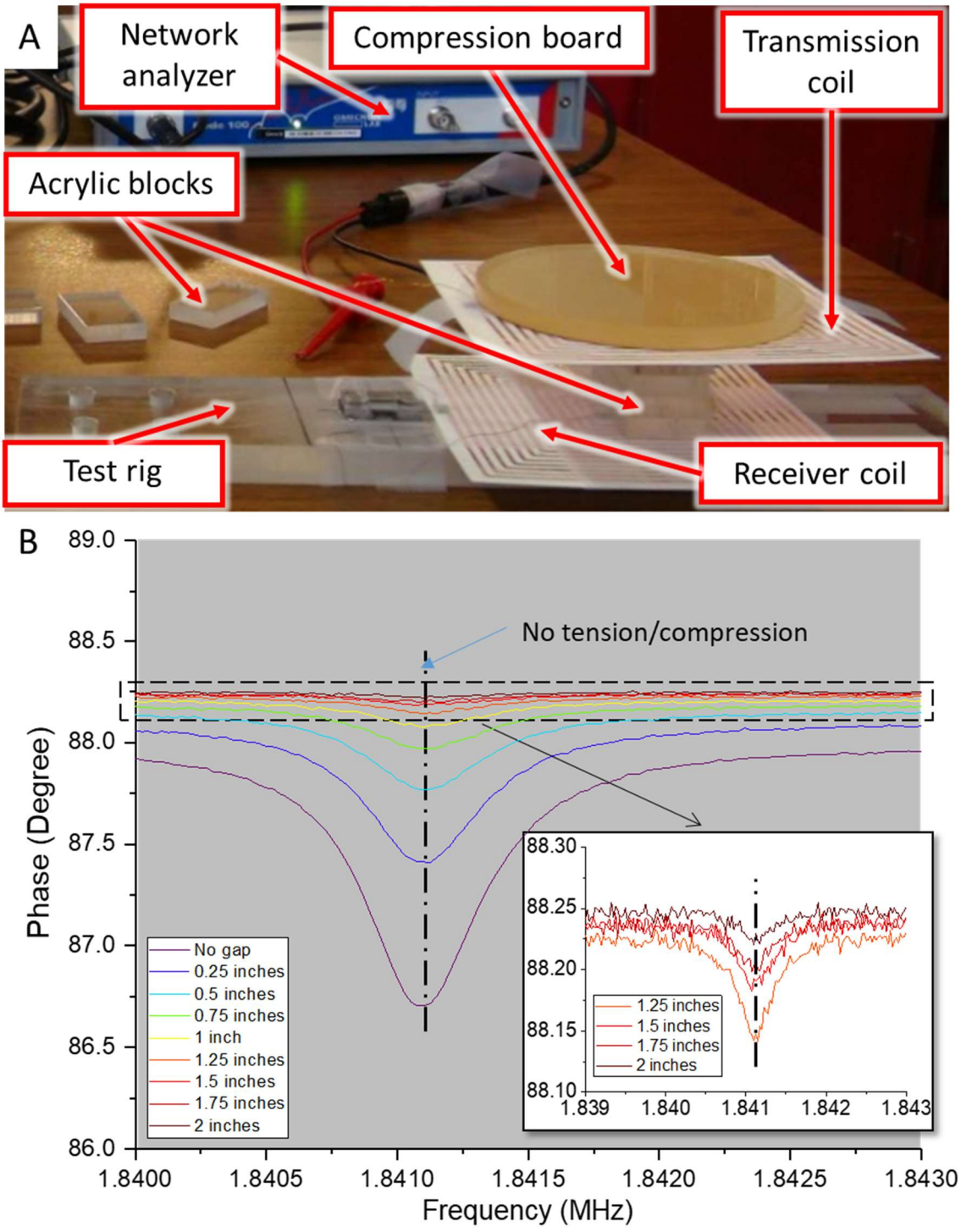


Figure 4.1.2.: The experimental setup and results for the effective wireless communication range testing. (a) Experimental setup to determine the range of effective wireless communication for the coaxial case. (b) Experimental results for the coaxial case by applying a vertical displacement ranging from 0 – 2 inches in increments of 0.25-inches.

The larger the vertical displacement the smaller the phase amplitude of the impedance, which decreases the accuracy of the resonant frequency measurement.

4.2. Magnetic Inductive Coupling

4.2.1. Background

The AC signal generated by the VNA provides a continuously alternating electric potential at a designated operating frequency ~1.84 MHz that creates an electromagnetic field around the reader coil. The magnetic flux is the perpendicular component of the magnetic field that passes through a closed surface, such as the surface of the sensor coil. The amount of magnetic flux that passes through the surface determine the mutual inductance between the two coils. The amount of magnetic flux that transmits from the reader coil through the sensor coil is a function of the relative position between them. To minimize the loss of mutual inductance between the reader and sensor coil, this work seeks to create a multicoil reader capable of locating the sensor coil. With the location of the sensor coil, we can align one of the coils from the multicoil reader with the sensor coil for accurate resonant frequency measurement.

4.2.2. Cylindrical Multiturn Coil Geometry

4.2.2.1. Experimental Procedure

This work began by first solving the “forwards problem” of determining the mutual inductance as a function of relative position. To accomplish this, this work employs an experimental procedure and high-fidelity multiphysics simulation. We begin by replicating

published results [28] to validate the experimental procedure and multiphysics simulation used for finding mutual inductance between two inductively coupled coils. The published work started by developing an equivalent circuit model for the coupled coils. The reader and sensor coils are denoted as coil 1and coil 2, respectively. For measuring the mutual inductance of two inductively coupled coils using the three-port transmission measurement the equivalent circuit diagram is depicted in **Figure 4.2.2.1.A**

In **Figure 4.2.2.1.A** R_i, L_i, V_i and I_i are the resistance, inductance, voltage and current in each respective coil. From the equivalent circuit, Heinzle derived a transfer function using Kirchhoff's Law as the following:

$$\frac{V_2}{V_1} = \frac{i\omega M R_L}{(R_1 + i\omega\ _1)(R_2 + R_L + i\omega L_2) + \omega^2 M^2} \tag{2}$$

where R_L is the load resistance is set to ~ 10 Ω; $R_1 . V_1$ and L_1 are the resistance, voltage, and inductance for coil 1; $R_2 . V_2$ and L_2 are the resistance, voltage, and inductance for coil 2; and $\omega = 2\pi f$ where f is the frequency. R_s is the source resistance inside the Bode 100 VNA. Heinzle simplified the transfer function because $R_L \gg R_{1,2}$ and $R_L \gg \omega L_{1,2}$, thus the transfer function simplifies to

$$\frac{V_2}{V_1} = \frac{M}{L_1} \tag{3}$$

Equation (3) provides a method to calculate the mutual inductance based on the transmitted voltage. To measure the transmitted voltage, the published work used a Bode 100 VNA and a 3-port transmission measurement. The VNA measured the gain between the two coupled coils over a range of frequencies. Heinzle graphed gain versus frequency and clearly

defined a gain plateau that they used for the mutual inductance calculation. Heinzle utilized the value of the gain at the plateau because at high frequencies the skin effect takes over, turning the inductive coil into a capacitor.

To confirm the validity of the experimental procedure, this work replicated the mutual inductance experiment using a similar coil geometry as employed by Heinzle in the published work. The cylindrical coil in the published work had 2 layers of 9 turns each for a total of 18 turns using a 1 mm diameter wire. The fabricated cylindrical coils in this work had a wire diameter of 0.9 mm, 2 layers of 10 turns for a total of 20 turns. This configuration most closely matched the cross-sectional area of the cylindrical coil in the published work. The inner diameter of the cylindrical coils in the published work matched the inner diameter of the fabricated cylindrical coils in this work at 13.5 mm. We employed the same measurement parameters as the published work for the Bode 100 VNA to measure the gain. The VNA swept a range of frequencies from 1 kHz to 10 MHz, with 401 points, a power level of 13 dBm, attenuator set to 0/20 dB for channels 1/2 respectively, a receiver bandwidth of 30 Hz, and set to measure gain in dB format. To fabricate the coils, we wrapped the 0.9 mm diameter wire around a 3D printed PLA center with an inner diameter of 13.5 mm. **Figure 4.2.2.4. A** display the fabricated coils wound around the PLA center in the coaxial configuration**.** We began the procedure by measuring the self-inductance of a single coil using the one-port reflection measurement using the experimental setup depicted in **Figure 4.2.2.1.B.** From the self-inductance measurement, we determined the self-inductance of the coil and location of the gain plateau, as seen in **Figure 4.2.4.C**, which was similar to the published work at ~20 kHz. Then, we employed the Bode 100 VNA in the three-port transmission configuration, as seen in **Figure 4.2.2.1.C,** to measure the gain

between the two cylindrical coils for a range of coil separations from 2 mm to 10 mm. We displayed the gain measurements in **Figure 4.2.2.4.D** and used the gain value at the plateau frequency of 20 kHz**.** From the gain values at the plateau, we calculated the mutual inductance using the simplified transfer function in (3) and the following conversion from gain to a voltage.

$$\frac{V_2}{V_1} = 10^{\frac{gain}{20}} \tag{4}$$

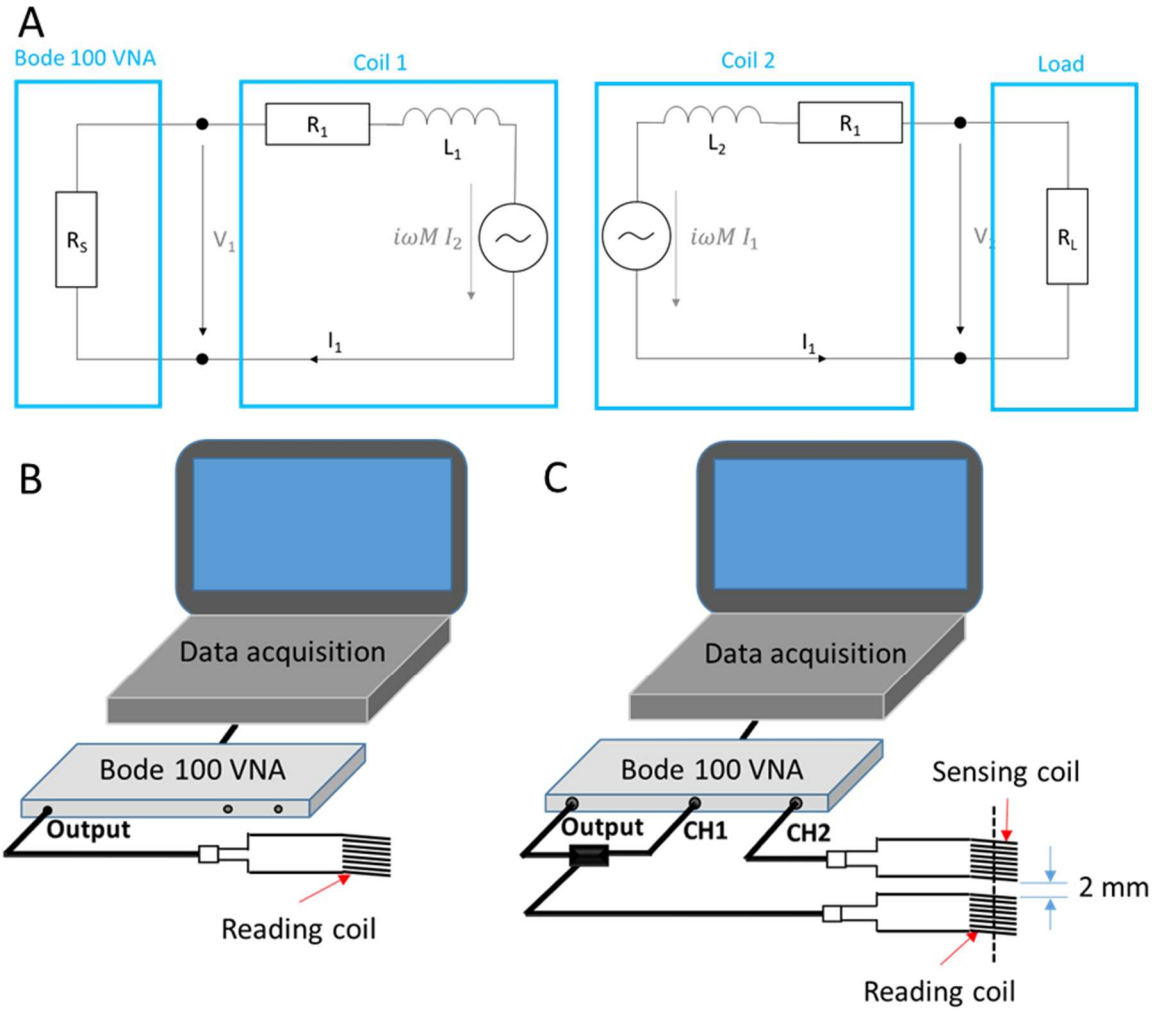


Figure 4.2.2.1.: (a) Equivalent circuit model for measuring the mutual inductance between two inductively coupled coils, with the reading coil (coil 1) connected to a source and the sensing coil (coil 2) connected to a load. (b) Experimental setup for the one-port reflection

measurement to find the self-inductance of a single coil (c) Experimental setup depicting the VNA connected to the two cylindrical coils in the three-port transmission configuration for the first vertical separation of 2 mm.

4.2.2.2. Theoretical Model of Mutual Inductance

The published results investigated the mutual inductance between two multiturn cylindrical coils using an experimental procedure and an analytical approximation. Heinzle derives the theoretical model [29] in three steps. First, they approximate a solution for two infinitesimally thin coaxial cylindrical filaments; second, they extend the infinitesimally thin filaments into filaments with a finite thickness; and third, they use superposition to extend the approximation into a coil with multiple turns and layers. The approximation for the mutual inductance between two multiturn cylindrical coils is as follows:

$$M \approx \pi\mu\overline{r_1}\overline{r_2}N_1N_2\int_0^\infty dmJ_1(m\overline{r_1})\,J_1(m\overline{r_2})e^{-m|\overline{h_1}-\overline{h_2}|} \tag{5}$$

where μ is the magnetic permeability constant; $\overline{r_1}$, $\overline{h_1}$, and N_1 are the average radius average separation, and number of turns for coil 1 respectively; $\overline{r_2}$, $\overline{h_2}$, and N_2 are the average radius, average separation, and number of turns for coil 2 respectively; and J_1 represents a Bessel function of the first kind. Next, we used (4) to calculate the mutual inductance for our coil geometry in MATLAB at the vertical displacements used in our experiment. To confirm the accuracy of the MATLAB script, we also replicated the published results for the analytical approximation using the geometry of the coil in their work. However, this model does not take frequency into account limiting the accuracy of the approximation.

4.2.2.3. High-Fidelity Multiphysics Simulation

To solve the "backward problem" of locating an embedded coil, we must first solve the "forward problem" of finding the mutual inductance between the two coupled coils as a function of relative position. To accomplish this, we employed high-fidelity COMSOL multiphysics simulations to calculate the mutual inductance between two inductively coupled coils. We used the AC/DC module and the magnetic fields node from COMSOL multiphysics to simulate the magnetic field between two cylindrical coils with identical geometries, and we compared the results to the experimental results discussed in the previous section. The simulation employed a lumped port [30] technique with a frequency domain study that calculated the voltage at each port based on the amount of magnetic flux density transmitted from the reader coil through the orthogonal surface of the sensor coil. The simulation calculated the gain between the two coils at the plateau frequency reported from the experiment ~20 kHz using the following equations.

$$\nabla \times \mathbf{H} = \mathbf{J} \tag{6}$$

$$\mathbf{B} = \nabla \times \mathbf{A} \tag{7}$$

$$\mathbf{J} = \sigma \mathbf{E} + j\omega \mathbf{D} + \mathbf{J_s} \tag{8}$$

$$\mathbf{E} = -j\omega \mathbf{A} \tag{9}$$

Where **H** represents the magnetic field intensity, **B** is the magnetic flux density, **J** is the current density, **A** is the magnetic vector potential. **E** is the electric field intensity, **D** is the electric displacement field, and $\mathbf{J_f}$ is the free current density. We performed the COMSOL simulation for coaxial cylindrical coils at a variety of vertical separation distances that are identical to the experiment and analytical calculations. We plotted the magnetic flux density from the reader coil in **Figure 4.2.2.4. B.** The results for the experimental data,

analytical approximation, and COMSOL simulation for the coils fabricated for this work are seen in **Figure 4.2.2.4. E.**

4.2.2.4. Results and Discussion

We observe the results in **Figure 4.2.2.4. E** that our experimental results and COMSOL simulation data for the fabricated cylindrical coil are within ~7% error of one another. The disagreement with the analytical solution is likely due to the approximation not taking frequency into account. **Figure 4.2.2.4. F** shows the results of **Figure 4.2.2.4. E** compared against the published results [28] discussed earlier. We expected some variation between the fabricated coil results and the published work because we did not have the exact diameter of the wire used in the published work. The results, however, show similar trends to the data with an offset in the mutual inductance values. This is likely due to the different geometries between the fabricated coil and coil used in the published work. From this work, we are confident in the validity of the experimental procedure and the COMSOL multiphysics simulation. We continue the work for the rectangular coil geometry, which we will use in the final multicoil reader.

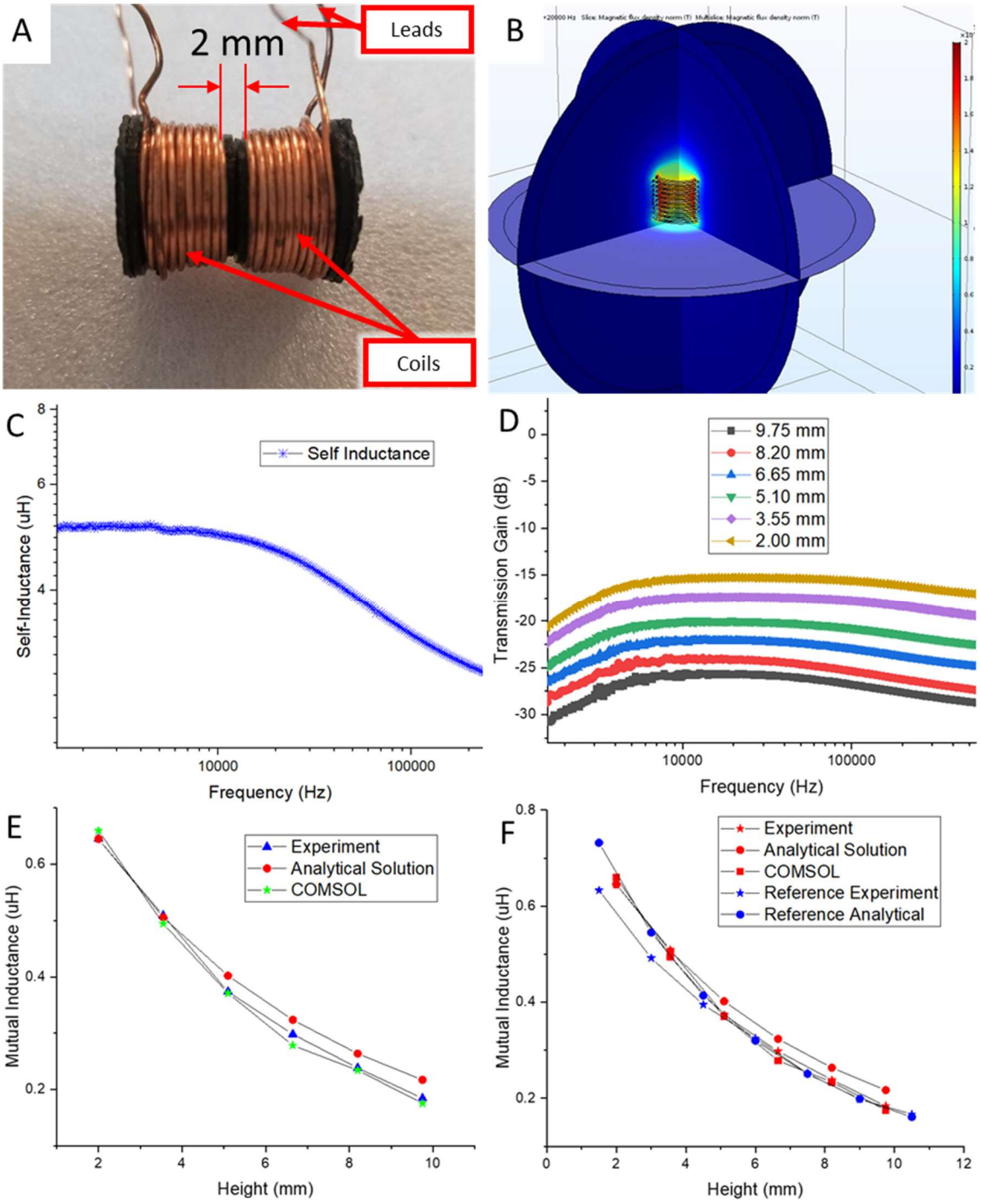


Figure 4.2.2.4.: (a) Experimental setup duplicated from published reference paper to find the mutual inductance between two inductively coupled cylindrical coils. (b) COMSOL multiphysics depiction of magnetic flux density within the reader coil. (c) Self-inductance of a single multi-turn coil with an inner diameter of 0.9 mm and containing 11 turns, as to

closely match the cross-sectional area of the coil used in the reference experiment. (d) 3-port transmission measurement of the gain between the two coils as a function of relative position using the Omicron Labs Bode™ 100 Vector Network Analyzer. (e) Compared the results from the mutual inductance experiment using our fabricated coil geometry against an analytical approximation and COMSOL simulation results. (f) Comparison of mutual inductance results from the reference experiment, reference analytical calculation, our experiment, our COMSOL simulation, and our analytical calculation for the fabricated coil geometry.

Cylindrical Coaxial Coil Geometry				
Verical Separation (mm)	Analytic (μH)	Experiment (μH)	COMSOL (μH)	Error %
2.00	0.64497	0.64516	0.65974	2.23832
3.50	0.50921	0.50582	0.49474	2.92378
5.10	0.37347	0.40207	0.371	0.66556
6.65	0.29809	0.32366	0.27821	7.14592
8.20	0.23816	0.26359	0.23409	1.73893
9.75	0.18421	0.21697	0.17554	4.94057

Table 4.2.2.4. The results from experiment, analytical solution, and COMSOL simulation for the cylindrical coaxial coil geometry with a wire diameter of 0.9 mm. The percent error is between the experimental results and the COMSOL simulation results.

4.2.3. Rectangular Multiturn Coil Geometry

4.2.3.1 Experimental Procedure

The rectangular planar coil was chosen for the final multicoil configuration to locate the position of the sensor coil. To solve the “forward problem” of finding the mutual

inductance as a function of position, we repeated the process for the cylindrical coil by comparing experimental results against multiphysics simulations and analytical approximations. The geometry of the rectangular planar coil is as follows: the inner diameter of the coil was 16 mm, the spacing between each turn was 2 mm, the width of the coil wire was 1.3 mm, the coil had 13 turns, and a thickness of 0.1 mm. We fabricated the coils using a Silhouette™ Vinyl Cutter that accurately cut the coil pattern into a sheet of copper tape. To control the relative position between the two coils with accuracy, we employed three displacement platforms that controlled the x-, y-, and z-directions. Pitch, roll, and yaw are neglected at this moment in time. The measurement process to determine the mutual inductance was the same as described for the cylindrical coil geometry. The Bode 100 VNA swept a range of frequencies from 1 kHz to 1 MHz, with 401 points, a power level of 13 dBm, attenuators set to 0/20 dB for channels 1/2 respectively, a receiver bandwidth of 30 Hz, and set to measure gain in dB. We began by measuring the self-inductance of a single rectangular coil and determining the location of the gain plateau. The gain plateau for the rectangular coil geometry was ~200 kHz, so we performed the gain measurements at this frequency. With the gain plateau located, we calculated the mutual inductance using the simplified transfer function in (3) and the gain conversion to voltage in (4). For the coaxial case, we tested a range of vertical separations from 0 mm to 25 mm in increments of 5 mm. The experimental setup, seen in **Figure 4.2.3.1 A,** provided the gain measurements and accurate position control.

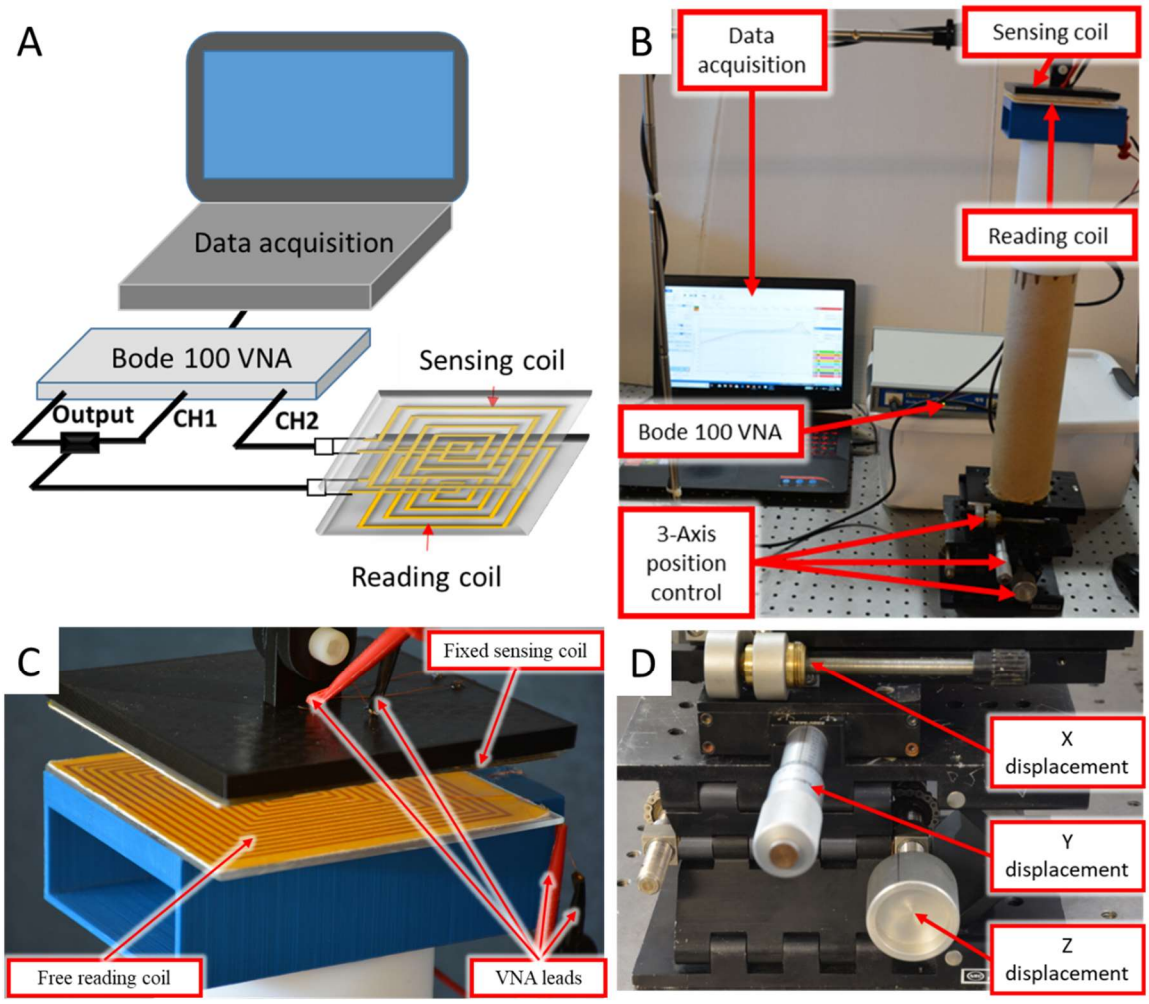


Figure 4.2.3.1.: (a) The overall configuration of the experimental setup. (b) The experimental setup mounted on an optical table for accurate position control. Bode 100 VNA connected to two rectangular planar coils for three-port transmission measurement of the gain. (c) Coil alignment with respect to one another where the fixed coil cannot move and the free coil moves due to the displacement platforms. (d) Displacement platforms controlling the free reader coil in the x, y and z directions.

4.2.3.2. Theoretical Model of Mutual Inductance

To calculate the mutual inductance of a rectangular planar multiturn coil, this work employed an analytical approximation [31], authored by Cheng, based solely on the geometry of the coil. Similar to the approximation for the cylindrical coils, this approximation does not take into account frequency and is for the coaxial case only. This model begins by approximating each turn of the rectangular multiturn coil as multiple concentric single turn coils with the same width and spacing, as seen from **Figure 4.2.3.2. A** to **Figure 4.2.3.2. B.** The Then, the model simplifies the single turn concentric coils to single turn concentric filaments with the gap between the coils being the sum of the original width and spacing, as seen from **Figure 4.2.3.2. B** to **Figure 4.2.3.2. C.**

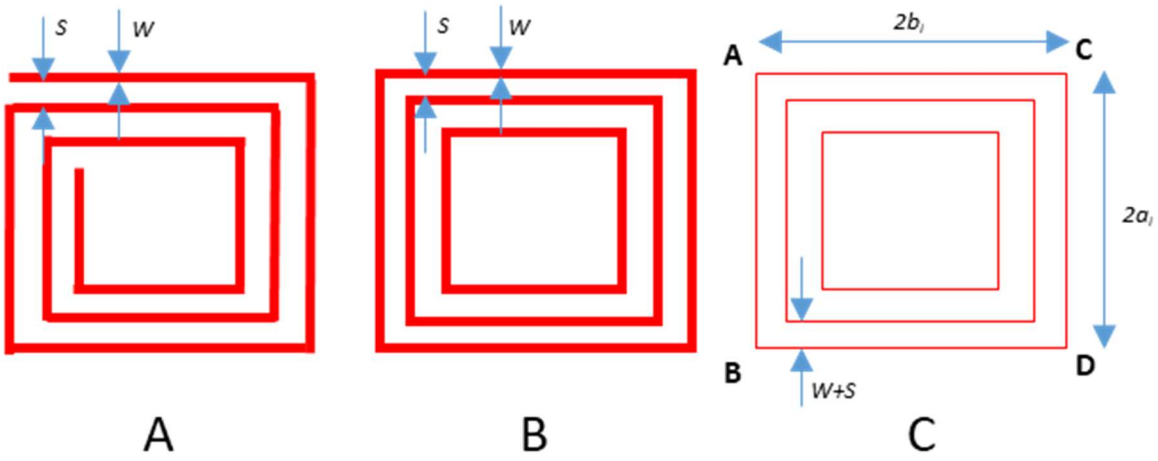


Figure 4.2.3.2.: (a) Multiturn planar rectangular coil. (b) Multiple concentric single turn coils. (c) Multiple concentric single turn filaments

The model denotes the width and length of the reader coil as $2a_i$ and $2b_i$, respectively; and the width and length of the sensor coil as $2c_j$ and $2d_j$, respectively. For our calculation, the length and width of the reader and sensor coils will be the same. The model defines each successive arm of each coil using the following equation.

$$a_i = a_1 - (i-1)(w+s) \quad \& \quad b_i = b_1 - (i-1)(w+s) \tag{10}$$

$$c_j = c_1 - (i-1)(w+s) \quad \& \quad d_j = d_1 - (i-1)(w+s) \tag{11}$$

After simplifying the coil geometry, the model begins using the Biot-Savart law to calculate the magnetic flux produced by each arm of each coil on one another, starting with the following equation

$$\Phi_{ij} = \int_{S_j} \mathbf{B_i} \cdot dS_j \tag{12}$$

From the magnetic flux equation, they implement the simplified geometry and derive an expression for the magnetic flux transmitted due to each arm of each coil in the z-direction. The final expression for the magnetic flux in the z-direction for arm *CD* as seen in **Figure 4.2.3.2. C** is as follows

$$\Phi_{CD-Z} = \frac{u_0}{2\pi}\left(\sqrt{\left(b(i)+d(j)\right)^2+z^2+\left(a(i)+c(j)\right)^2} - \left(a(i)+c(j)\right)\operatorname{atanh}\left(\frac{(a(i)+c(j))}{\sqrt{\left(b(i)+d(j)\right)^2+z^2+\left(a(i)+c(j)\right)^2}}\right) - \sqrt{\left(b(i)+d(j)\right)^2+z^2+\left(a(i)-c(j)\right)^2} + \left(a(i)-c(j)\right)\operatorname{atanh}\left(\frac{(a(i)-c(j))}{\sqrt{\left(b(i)+d(j)\right)^2+z^2+\left(a(i)-c(j)\right)^2}}\right) - \sqrt{\left(b(i)-d(j)\right)^2+z^2+\left(a(i)+c(j)\right)^2} + \left(a(i)+c(j)\right)\operatorname{atanh}\left(\frac{a(i)+c(j)}{\sqrt{\left(b(i)-d(j)\right)^2+z^2+\left(a(i)+c(j)\right)^2}}\right) + \sqrt{\left(b(i)-d(j)\right)^2+z^2+\left(a(i)-c(j)\right)^2} - \left(a(i)-c(j)\right)\operatorname{atanh}\left(\frac{a(i)-c(j)}{\sqrt{\left(b(i)-d(j)\right)^2+z^2+\left(a(i)-c(j)\right)^2}}\right)\right) \tag{13}$$

Where z is the vertical separation distance between the two coils and μ_0 is the permeability of free space. The final equation sums up the magnetic flux from all the arms of each single turn coil and calculates the mutual inductance as the following:

$$M = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} \Phi_{ij} \quad (14)$$

Where N_1 was the number of turns for the reader coil and N_2 was the number of turns for the sensor coil. We performed the calculation in MATLAB due to the complexity and obtained results for the same range of vertical separations, z, used in the experiment.

4.2.3.3. High-Fidelity Multiphysics Simulation

To solve the "forward problem" of finding the mutual inductance as a function of relative position, this work utilized the verified simulation method described for the cylindrical coils but with the rectangular planar coil geometry. The rectangular planar coil consisted of 13 turns, an inner diameter of 16 mm, a coil spacing of 2 mm, a wire width of 1.3 mm, and a wire thickness of 0.1 mm**.** We employed the lumped port technique with the frequency domain study to calculate the gain between the two inductively coupled coils. Next, we applied a mesh to the coils and surrounding air, but due to the thin wire of the coil, the number of mesh elements increased dramatically. To account for the thin edge of the coil wire, we implemented a swept mesh by defining a triangular mesh on the surface of the coil wire and sweeping the mesh through the thickness of the wire to the opposing side. This meshing technique reduced the number of mesh elements, which decreased the computational time from approximately ~1 hour per simulation to ~4 minutes. Additionally, we implemented a swept mesh on the spherical infinite element domain, the entire coil geometry, and the lumped ports. The remaining air domain, between the infinite

element domain and the coil geometry, used a standard tetrahedral mesh, as seen in **Figure 4.2.3.3. A.** We performed the simulation at ~200 kHz to match the frequency used during the experiment. We began with the coaxial case for a variety of vertical separations, which are the same separations used in the experiment and analytical approximation. The results in **Figure 4.2.3.3. B** shows the magnetic field lines in red and how a change in relative position affects the magnetic flux that transfers from one coil to another. **Figure 4.2.3.3. C** and **Figure 4.2.3.3. D** depicts the magnetic flux density produced by the reader coil and how a change in position will affect the amount of magnetic flux that permeates the surface of the sensor coil.

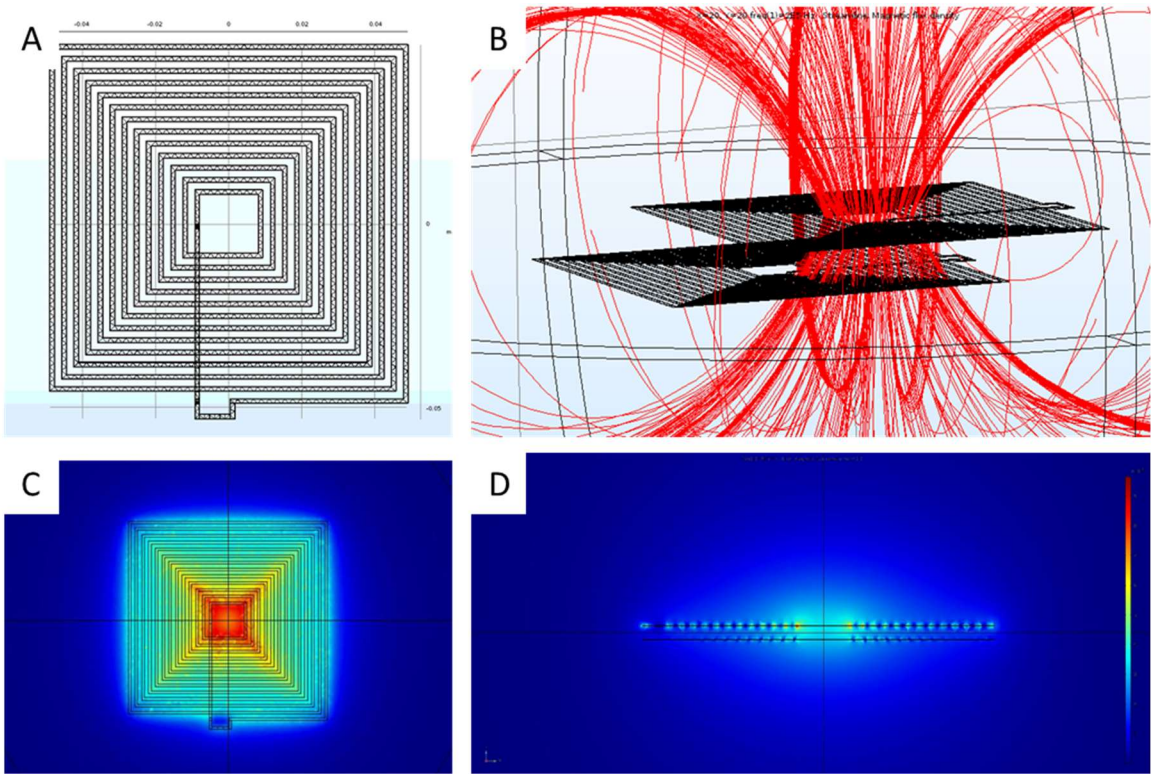


Figure 4.2.3.3.: (a) The swept meshing of the rectangular coil. (b) Lumped port analysis of the magnetic flux streamlines at 200 kHz for a position offset of 20 mm in x-direction, 20 mm in the y-direction, and 10 mm in the z-direction. (c) Magnetic flux density distribution of the reader coil from the top view for a vertical separation of 5 mm. (d)

Magnetic flux density distribution of the reader and sensor coils for a vertical separation of 5 mm.

4.2.3.4. Results and Discussion

To confirm the validity of the simulation for the rectangular coil geometry, we performed a previously verified experiment [28], and a published analytical approximation [31]. The experimental and COMSOL simulation results agreed within 1% everywhere except at 25 mm. When comparing the COMSOL simulation results and the analytical solution, we saw a near identical trend with a small magnitude offset. The offset at every location was ~2% because the analytical approximation did not take frequency into account. We validated the COMSOL simulation to calculate the mutual inductance at a variety of offset locations. The mutual inductance at each position will allow us to locate the embedded coil.

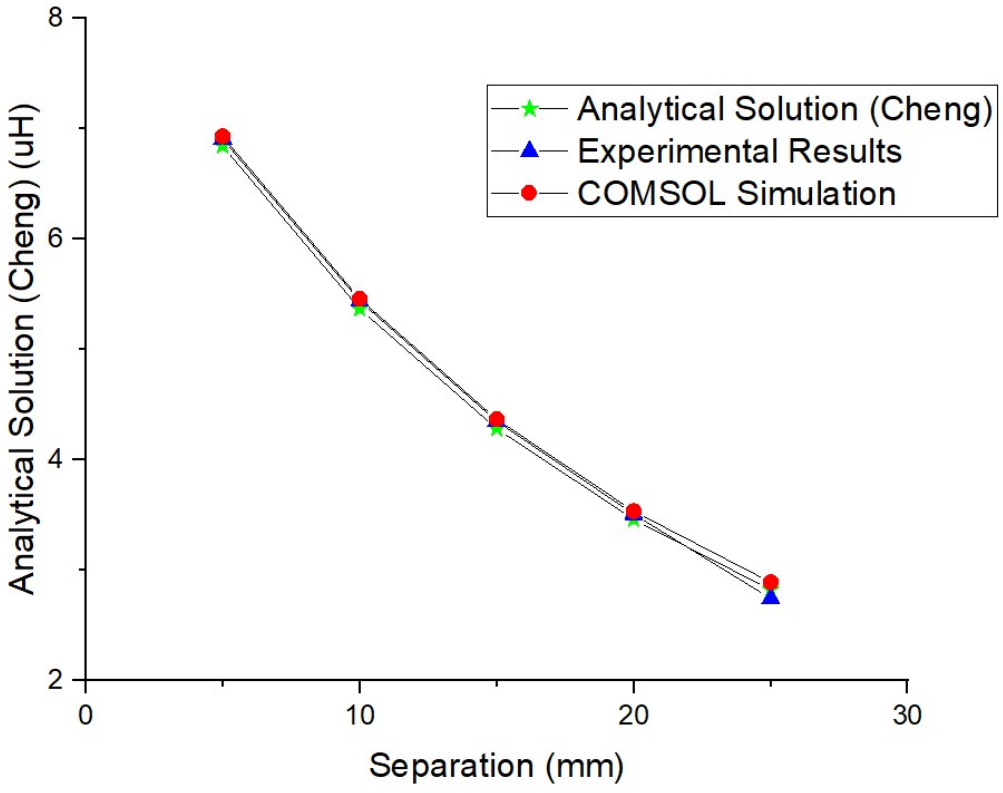


Figure 4.2.3.4. The plotted results from the experiment, analytical solution, and COMSOL simulation for the coaxial case of the rectangular coil geometry

Rectangular Coaxial Coil Geometry				
Verical Separation (mm)	Analytic (μH)	Experiment (μH)	COMSOL (μH)	Error %
5	6.844	6.90695	6.9311	0.34843
10	5.369	5.44047	5.45827	0.32614
15	4.284	4.35197	4.36758	0.35729
20	3.461	3.50933	3.53714	0.78637
25	2.825	2.74615	2.89324	5.0841

Table 4.2.3.4. The results from the experiment, analytical solution, and COMSOL simulation for the rectangular coaxial coil geometry. Percent error compares experimental results against simulation results.

4.2.4 High-Fidelity Multiphysics Simulation

4.2.4.1. Solve Forward Problem

With the validated simulation and experiment, we solved the "forward problem" of determining the mutual inductance as a function of relative position for all three degrees of freedom. To accomplish this, we implemented the verified COMSOL simulation for the rectangular coil geometry for a range of horizontal and vertical displacements. We began by simulating results for a vertical offset in the z-direction of 5 mm and a range of horizontal displacements in the x- and y-directions from 0 – 40 mm in increments of 5 mm for a total of 81 locations in the positive x-y plane. We then reran the experimental procedure for the mutual inductance measurement at the same 81 locations and compared the results against the simulation results. The graph in **Figure 4.2.4.1. A** depicts the mutual inductance results from experiment and simulation for the vertical displacement of 5 mm,

x-displacement equal to 0 mm, and y-displacement ranging from 0 mm to 40 mm in increments of 5 mm. We observed a close correlation with a maximum error of ~5% at the largest displacement, which we expected. Next, we plotted the experimental and simulation results for the z-displacement of 5 mm; x-displacements 0, 10, and 15 mm; and y-displacement ranging from 0 – 40 mm in increments of 5 mm in **Figure 4.2.4.1. B. Figure 4.2.4.1. C** depicts the entire x-y plane from 0 to 40 mm and a z-displacement of 5 mm with an interpolated surface plot and colormap. When testing every location within the x-y plane at a vertical separation of 5 mm, we observed a maximum error of ~7%**,** when comparing experimental results to the COMSOL simulation seen in **Figure 4.2.4.1. C.** Lastly, we simulated the 81 points of the x-y plane for a vertical displacement for 2, 3, and 4 mm and ranging from 5 mm to 40 mm in increments of 5 mm**.** With the range of displacements in the x-, y-, and z-directions; we used symmetry to expand the mutual inductance data to the other four quadrants of the symmetric coil. From simulations, we built a library of mutual inductances at all the position offsets. **Figure 4.2.4.1. D** displays an interpolated 3D colormap of the mutual inductance values as a function of position for one quadrant.

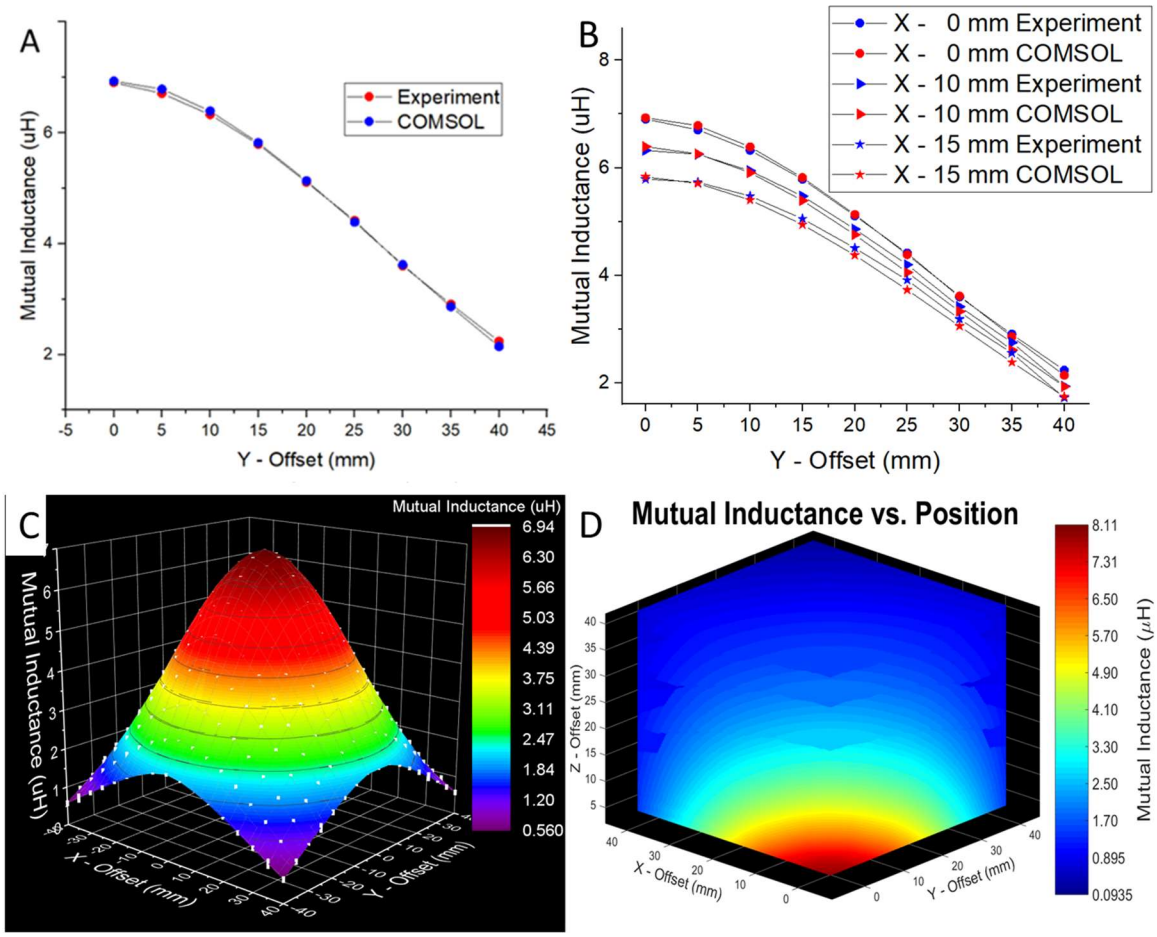


Figure 4.2.4.1.: (a) COMSOL simulation results compared against experimental results for vertical separation of 5 mm, x-direction offset of 0 mm, and a y-direction offset ranging from 0 mm to 40 mm in increments of 5 mm. (b) COMSOL simulation results compared against experimental results for a vertical separation of 5 mm; x-direction offsets for 0, 10, and 15 mm; and a y-direction offset ranging from 0 mm to 40 mm in increments of 5 mm (c) Comparison of experimental and simulation results for all position at a vertical offset of 5 mm with error bars in white. (d) Interpolated colormap of mutual inductance from COMSOL simulation results for all x-y positions ranging from 0 – 40 mm for a range of vertical separations from 5 – 40 mm.

Rectangular Coil Geometry for Z = 5 mm				
X (mm)	Y (mm)	Experiment (μH)	COMSOL (μH)	Error %
0	0	6.90695	6.9311	0.34841
0	5	6.70635	6.7852	1.16205
0	10	6.3261	6.3904	1.00612
0	15	5.79077	5.8194	0.49196
0	20	5.10901	5.1347	0.50041
0	25	4.4176	4.388	0.67457
0	30	3.60028	3.6202	0.55022
0	35	2.90929	2.8643	1.57057
0	40	2.2402	2.1462	4.37974

Table 4.2.4.1.: COMSOL simulation results compared against experimental results for a vertical separation of 5 mm, x-direction offset of 0 mm, and a y-direction offset ranging from 0 mm to 40 mm in increments of 5 mm.

4.3. Multicoil Reader

4.3.1. Prototype Design

This work designed, built, and tested a multicoil reader capable of detecting the location of an embedded coil. First, we fabricated four coils from copper tape with a cellulose backing using the Silhouette™ Vinyl Cutter. Then, we soldered leads to each end of each coil and insulated the coils with a layer of Kapton™ Tape. We mounted each coil in each corner of the four-square cellulose backings, seen in **Figure 4.3.1.1.A**, that was 1.5x the size of the coil. We then layered all four cellulose backings on top of one another so the coils were overlapping with each other, as seen in **Figure 4.3.1.1.B.** To interrogate each coil individually, we employed two four-channel relays that connected the Bode 100 VNA to each coil, one at a time, while leaving the remaining coils in an open circuit

configuration. This prevented the active reader coil from inducing current into any of the reader coils not activated. We then wrote a MATLAB script to operate the VNA and cycle through the relays one at a time and measure the magnitude of the impedance in each coil.

4.3.1.1 Multicoil Reader Operation and Experiment

We performed an experimental procedure, depicted in **Figure 4.3.1.1.C,** to test the multicoil reader and determine the viability of the current configuration. To control the relative position between the multicoil reader and the sensor coil, we again implemented three displacement platforms that controlled the x-, y-, and z-directions**.** We chose the configuration for the multicoil reader because having the coils side-by-side would reduce the mutual inductance to a point that the impedance in each coil would be undetectable. To overcome this, we overlapped the coils on top of each other by half the total width of a single coil. We employed the Bode 100 VNA to perform a one-port reflection measurement of the impedance in each coil. However, using the one-port reflection measurement does not allow us to measure the mutual inductance and the three-port transmission measurement would not work in application because one must be wired to the sensing coil.

To obtain the mutual inductance from the impedance measurement, we first employed an analytical representation of the relationship between impedance and mutual inductance, derived in [27], as the following:

$$Z_{eff} = R_3 + j2\pi f L_3 + \frac{(2\pi f M)^2}{\left(R_1 + j2\pi f L_1 + \frac{1}{j2\pi f C_1}\right) || \left(\frac{1}{j2\pi f C_0}\right) + R_2 + j2\pi f L_2} \quad (15)$$

Where R_x represents the resistance, L_x the self-inductance, C_x the capacitance, M the mutual inductance, and Z the impedance. We assume the real part of the impedance does

not affect the mutual inductance of the system. From this equation, we can relate the amplitude of the phase to the mutual inductance squared. To check the validity of this relationship between mutual inductance and phase amplitude, we devised an experiment using coil 1 from the multicoil reader to measure the phase amplitude for a z-displacement of 5 mm, x-displacement of 0 mm, and y-displacement ranging from 0 to 40 mm in increments of 5 mm. Next, we calculated the expected phase amplitude at the same locations using the relationship between the mutual inductance and phase amplitude in (15). To accomplish this, we used the measured phase amplitude at a single point; x-, y-, and z-displacements equal to (0,0,5); and calculated the expected phase amplitude using the following equation:

$$P_{x,y,z} = \left(\frac{M_{x,y,z}}{M_{0,0,5}}\right)^2 * P_{0,0,5} \tag{16}$$

Where $P_{x,y,z}$ is the phase amplitude at any location, $M_{x,y,z}$ is the mutual inductance at any location from the COMSOL simulations, $M_{0,0,5}$ is the mutual inductance at x-, y,-, and z-displacements (mm), and $P_{0,0,5}$ is the phase amplitude at the same x-, y-, and z-displacements (mm). We took the measured phase amplitudes from coil 1 in the multicoil reader and compared them against calculated phase amplitudes using (16). The results from the experiment, depicted in **Figure 4.3.1.1. D,** show the comparison of the measured phase amplitudes against the calculated phase amplitudes**.** We obtained a close correlation between the expected values and the experimental results for y-displacements under 30 mm. For the y-displacements under 30 mm, we saw a maximum error under 4%, as seen in **Table 4.3.1.1.**

When we repeated the experiment for the second, third, and fourth coils for the center of each respective coil; we were unable to obtain results that matched. For example, we tested the coaxial case of each individual coil at a vertical displacement of 5 mm. We expected the phase amplitudes to be relatively close, but they were substantially different. We discovered that the shielding from multiple overlapping coils affected the magnitude of the phase measurements. For example, the fourth coil that was furthest from the sensor coil saw three overlaps in the top left quadrant, one overlap in the top right and bottom left quadrants, and no overlap in the bottom right quadrant. Moreover, we observed that the shielding due to the overlapping coils changed due to the relative position of the sensor coil. This led to a redesign of the reader to a single coil reader that took measurements in the four locations of the multicoil reader.

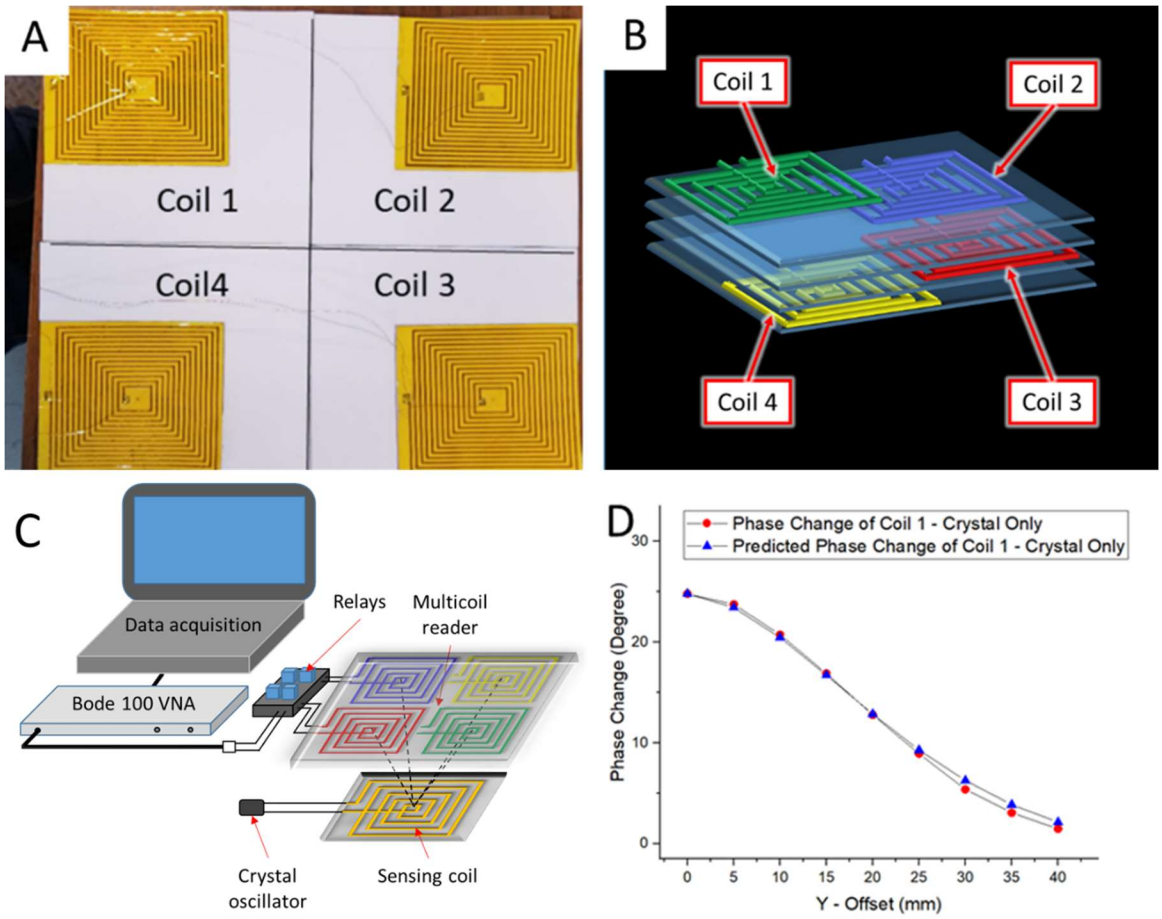


Figure 4.3.1.1.: (a) Each of the four coils mounted on a square cellulose backing, one coil in each of the four corners of the square. (b) coils stacked on top of one another so each coil is overlapping one another by half the width. (c) Experimental setup for testing the multicoil reader using the one-port reflection measurement of the impedance (d) Experimental results, using a piezoelectric crystal only, for coil 1 at a location in the x-direction of 0, y-direction ranging from 0 – 40 mm in increments of 5 mm, and z-direction of 2 mm.

Rectangular Coil Geometry						
X (mm)	Y (mm)	COMSOL Mutual Inductance (μH)	Normalized Mutual Inductance	Experimental Change in Phase (Degree)	Calculated Change in Phase (Degree)	Error %
0	0	8.10734	1	24.80441	24.80441	1.60E-05
0	5	7.88157	0.97215	23.74943	23.44214	1.31081
0	10	7.36597	0.90856	20.75352	20.47538	1.35839
0	15	6.65741	0.82116	16.8926	16.72563	0.99828
0	20	5.84093	0.72045	12.78595	12.87469	0.68928
0	25	4.96514	0.61243	8.93494	9.30327	3.95912
0	30	4.07795	0.503	5.40296	6.27562	13.90565
0	35	3.20879	0.39579	3.07587	3.88557	20.83868
0	40	2.37994	0.29355	1.48441	2.13749	30.55392

Table 4.3.1.1.: Experimental results for Coil 1 in the multicoil reader. We used the first location where x- and y-displacements equal zero, as the known location for mutual inductance and phase amplitude, to calculate the expected phase amplitude at the remaining locations. The error increased for larger displacements due to the curve fitting.

4.3.2. Prototype Design of Rotary Reader

To overcome the shortcomings of the multicoil reader, we designed a reader mount that moved a single coil to the four positions of the multicoil reader as opposed to having one coil at each location. This removed the nonuniform shielding effect the overlapped coils had on each other. The rotary reader consisted of a single coil that measures the impedance of the sensor coil at four locations 47 mm apart in a square configuration. However, the rotary action rotated the orientation of the coil which would affect the mutual inductance between the two coils. To account for this, we designed a system of gears to counter rotate the mounted coil and maintain a constant orientation, as seen in **Figure 4.3.2. B.** We mounted the coil on the bottom of the rotating disc, represented in **Figure 4.3.2 C**

by the square plate. The outer most gear consisted of 60 teeth, the two largest inner gears consisted of 20 teeth each, the two smallest gears 10 teeth each, and the mid-sized gear consisted of 15 teeth. We fixed two sets of 10- and 20- tooth gears for a total reduction from 60 teeth per rotation to 15 teeth per rotation, which is the size of the last gear. The last gear counter-spun to the rotation of the black disc and kept the orientation of the coil relative to a fixed location. The 15-tooth gear, which we mounted the coil to, was mounted at a radius of $47\sqrt{2}$ giving us a spacing of 47 mm between each corner of an overlaid square. We printed the entire rotary reader out of PLA on a Flashforge Creator Pro 3D printer as to not interfere with the magnetic field generated by the reader coil.

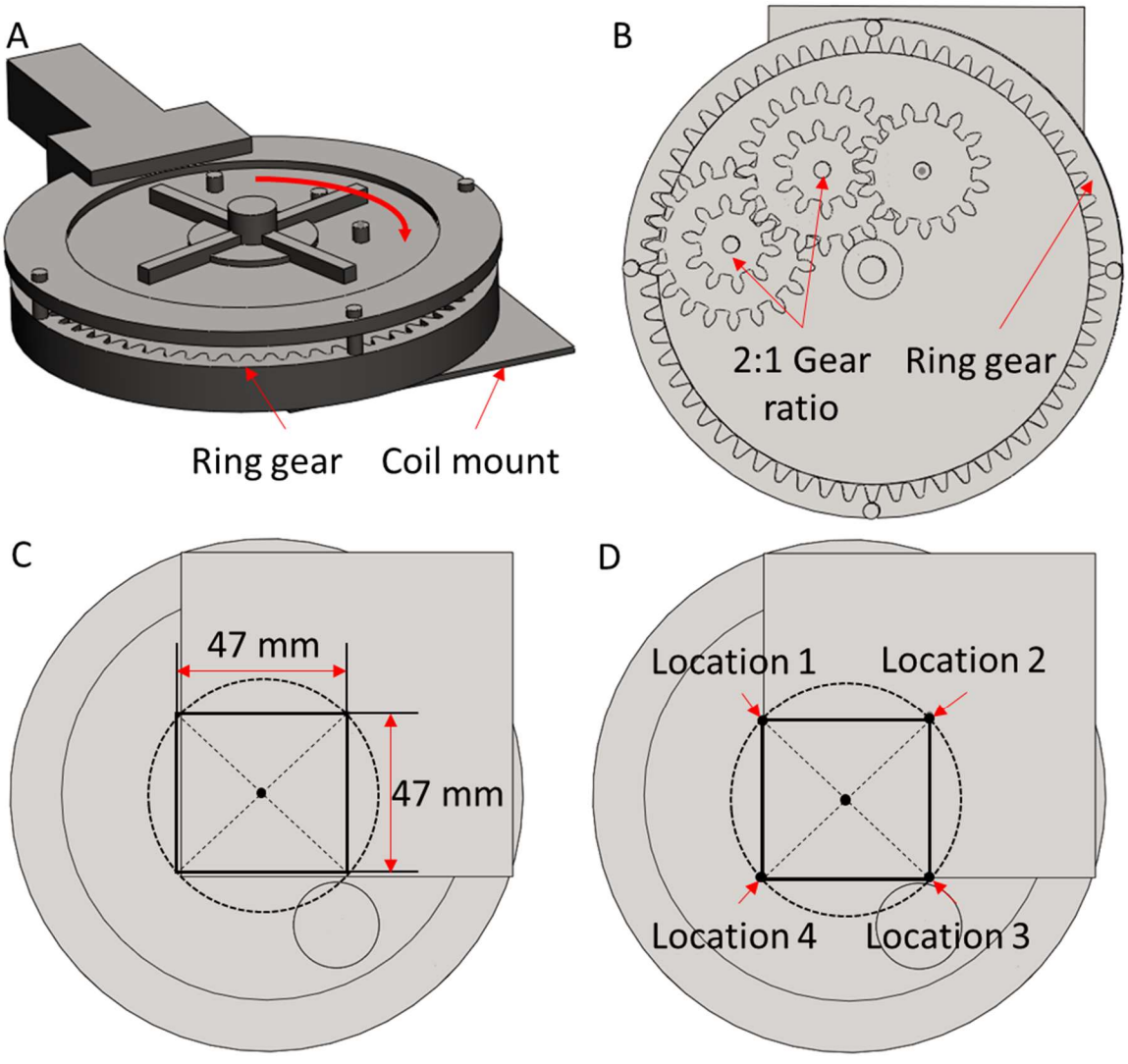


Figure 4.3.2.1.: (a) Isometric-view of the rotary reader with mounting plate for the single coil (b) Rotary Reader with top cover hidden to depict set of reduction gears that counter rotates the mounted coil at the same rate of rotation. (c) Bottom-view of the rotary reader depicting distance, exactly 47 mm, between the center locations of each coil position. (d) Bottom-view showing the four measurement positions and the order of the measurements.

4.3.3. Real-time Locating of Sensor Coil

After completion of the hardware, we adapted a method documented in [25] to convert the measured impedances from the four locations of the reader coil into x-, y-, and z- displacements that represented the location of the sensor coil. We began this work by solving the "forward problem" of determining the mutual inductance as a function of relative position. Next, we solved the "backward problem" of determining the relative position of the sensor coil as a function of the measured mutual inductances. To accomplish this, we implemented a backward conversion technique to reverse lookup the relative position from measured mutual inductances at the four locations. We employed equation (16) to convert the measured phase amplitudes into mutual inductance values. Next, we employed the **interp3** [32] to interpolate the COMSOL simulation data of the mutual inductance values in between the step size of 5 mm. Of the four coil locations, we only required three locations to find the position of the sensing coil. This allowed for multiple measurements of the sensor coil location using different combinations of the four locations. For example, we executed three measurements from the coil locations corresponding to: L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. Each set of three locations had three corresponding libraries that were created using the "scattered interpolant" [33] function and searched using Simulink's n-D lookup tables [34] to locate the sensor coil. With the

help of Ajay Dusane, we wrote MATLAB scripts to sample the rotary reader at the four locations, assemble the libraries, import the data into Simulink, and implement the n-D lookup tables.

4.3.3.1. Assembled Libraries and Simulink n-D Lookup Tables

To determine the location of the sensor coil based on data obtained during the rotary coil experiment, we employed Simulink's n-D lookup tables within MATLAB to search the assembled libraries to locate the sensor coil. To visualize the assembled libraries for coil locations L1, L2, and L3; we implemented **plot3k** [35] to create a 3D plot of the simulated mutual inductance values for each coil location M1, M2, and M3 against the colormap x-displacement seen in **Figure 4.3.3.1.1. A**. We observed a non-cubic dataset when plotting the mutual inductances for the three coil locations against the x-, y- and z-displacements. The n-D lookup tables required cubic datasets and as such did not function in their current form. To make the datasets cubic, we narrowed the range of mutual inductance data for the three coil locations to 2 to 4 μH; as seen in **Figure 4.3.3.1.1 B, C, and D;** for the x-, y-, and z-dimensions respectively. To convert the 3D plots in **Figure 4.3.3.1.1 B, C, and D** into three searchable libraries; we employed the "scattered interpolant" function in MATLAB. The "scattered interpolant" created three libraries that had M1, M2, and M3 as the locations in space and a corresponding value at each location for the x-, y-, and z-displacements. This formed the three libraries that the n-D lookup tables searched to locate the sensing coil. We assembled two additional sets of three libraries for the two remaining coil location combinations: L1, L2, and L4; and L4, L2, and L3. The n-D lookup tables employed its own linear interpolation to interpolate the assembled libraries to more accurately locate the sensor coil. However, narrowing the

range of mutual inductances also narrows the range of detection for the sensing coil. The limiting mutual inductance range limited the detectable range of the sensor coil to +/- 15 mm in the x-y plane and less than 21 mm in the z-direction. Future work could devise a transformation for the mutual inductance libraries into a searchable cubic shape.

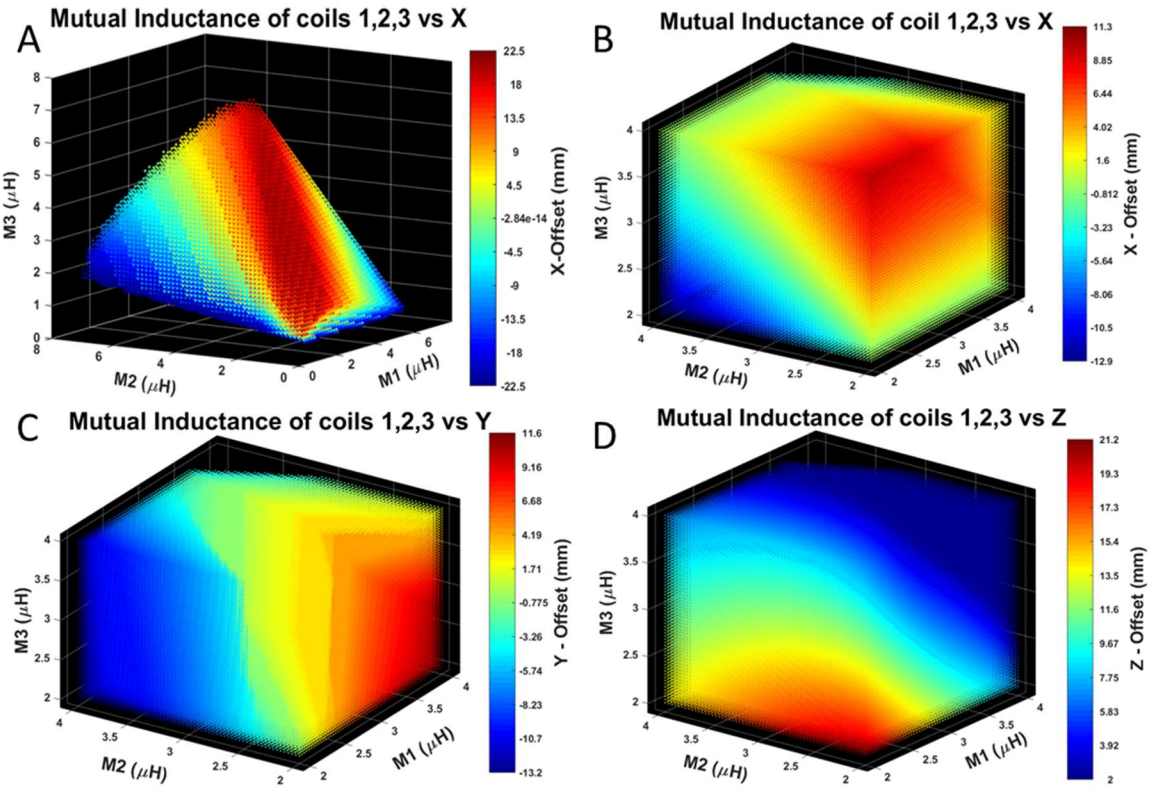


Figure 4.3.3.1.1.: (a) Plotted the interpolated mutual inductance in coil 1, 2, and 3 as a function of the x-coordinate, colormap. Plotted interpolated mutual inductance data, M1, M2, and M3 in coil locations L1, L2, and L3 as a function of the (b) x-coordinate, (c) y-coordinate, and (d) z-coordinate, colormap, representing the assembled libraries for the modified range. Used "Scattered Interpolant" to convert 3D plots into three functions where M1, M2, and M3 are locations in space and x-, y-, and z-displacements are the values at each location. This allowed the n-D lookup tables to search and interpolate the assembled libraries and determine the location of the sensing coil given the measured mutual inductances at the three coil locations.

To implement the n-D lookup tables, we first imported the experimental mutual inductance data m1, m2, and m3 from the three coil locations L1, L2, and L3 into Simulink using the “from workspace” input function. With the experimental data in Simulink, we employed three n-D lookup tables to search the three assembled libraries corresponding to coil locations L1, L2, and L3 to find the x-, y-, and z-coordinates of the sensor coil. To validate the accuracy of the n-D lookup tables, we simulated experimental mutual inductance data for the coil locations L1, L2, and L3 using **interp3** [32] to interpolate the COMSOL simulation data of the mutual inductance**.** We interpolated the simulated experimental data using a step size of 0.1 mm. We then implemented the n-D lookup tables, as seen in **Figure 4.3.3.1.2,** using simulated experimental data as the m1, m2, and m3 input and the assembled libraries for coil locations L1, L2, and L3 to locate the sensor coil.

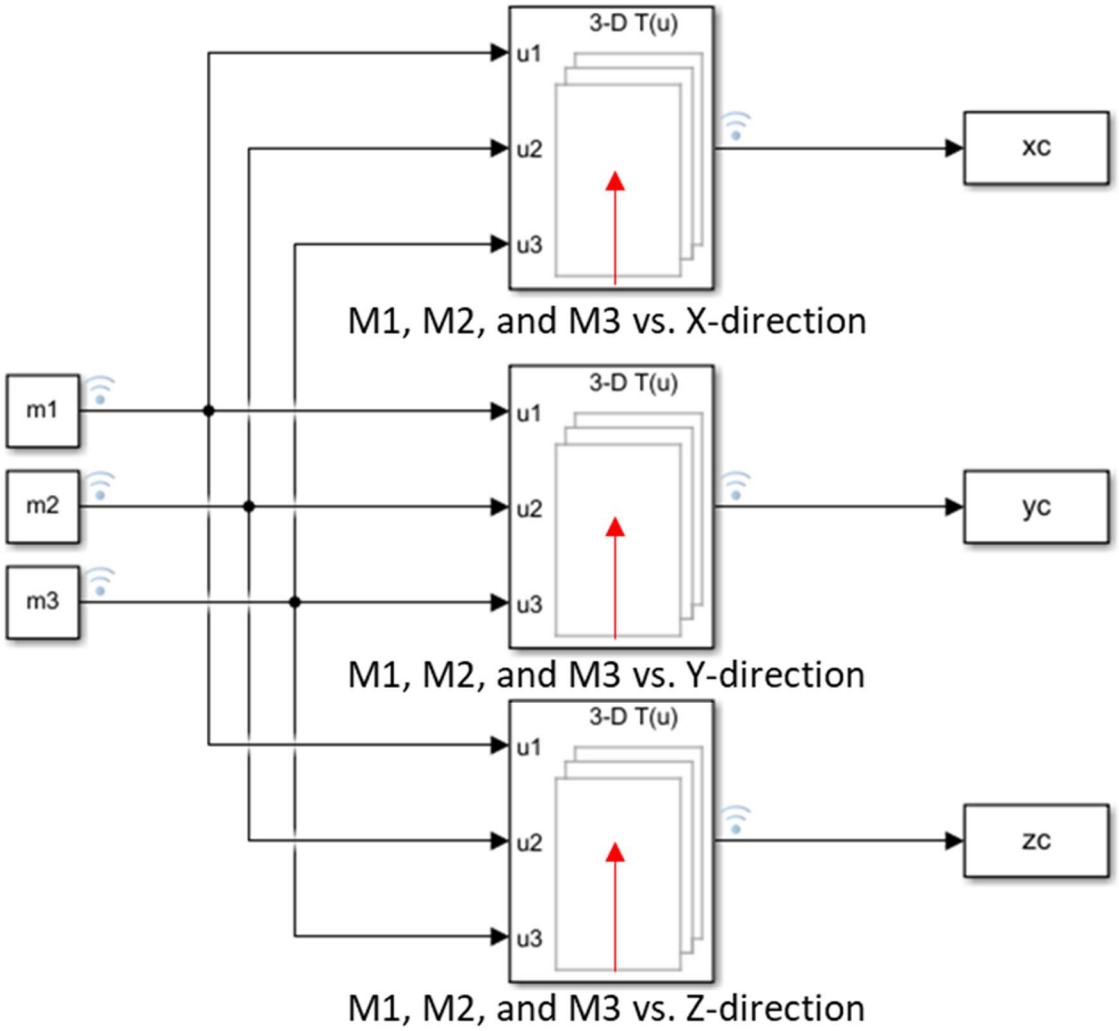


Figure 4.3.3.1.2. n-D lookup tables ran by importing data from the workspace in MATLAB and run in Simulink. Where m1, m2, m3 are the measured mutual inductances; assembled libraries for M1, M2, and M3 versus x-, y-, and z-directions; and xc, yc, and zc are the corresponding locations of the sensor coil.

5. Results and Discussions

The goal of this work was to implement a multicoil reader to locate measure the strain on a piezoelectric sensor. To accomplish this, we first solved the “forwards problem” of finding mutual inductance as a function of position. We validated an experimental procedure and COMSOL multiphysics simulation against published data for a cylindrical coil. We then executed the experimental procedure, matched the results to a high-fidelity multiphysics simulation, and confirmed the two with an analytical approximation for the planar rectangular coil. We built a library of simulated mutual inductance data between two inductively coupled coils for a range of offsets in relative position. However, due to the shielding effect generated by the overlapped coils of the multicoil, we switched the configuration of the reader from multiple coils at fixed locations to a single coil that translates to multiple locations. We implemented the rotary reader to translate a single coil to the four measurement locations while maintaining a constant orientation. This allowed for the measurements to occur on the same plane and eliminated the shielding effect caused by overlapping the coils. To solve the “backward problem”, we assembled libraries of mutual inductance data for three different coil locations as a function of the sensor coil location. The n-D lookup tables searched the assembled libraries, given measured mutual inductance data at the coil locations, to locate the sensor coil. To validate the n-D lookup tables, we interpolated the COMSOL simulation data of the mutual inductance to simulate experimental data for the narrowed range of offsets in relative position. Using the rotary reader, we sampled 13 different offsets in relative position between the rotary reader and sensor coil. For the four coil locations of the rotary reader, we measured the impedance between the reader and sensor coils. From the phase amplitude of the impedance, we

calculated mutual inductance for each of the four coil locations. We implemented three different calculations of the sensor location using the following coil location combinations: L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. Each combination employed different assembled libraries that Simulink's n-D lookup tables searched to locate the sensor coil. We determined the location of the sensing coil for 13 offsets in relative position between the reader and sensor coil.

5.1. Linear Interpolation

To validate the n-D lookup tables, we employed interpolated mutual inductance data to simulate experimental data for the narrowed range of position offsets. The interpolated data included points ranging from -15 mm to 15 mm with steps of 0.1 mm in the x- and y-directions, less than 20 mm in the z-direction with steps of 0.1 mm, and used linear interpolation with the **interp3** function. The step size of the interpolated mutual inductance data used to assemble the libraries was 0.2 mm. With the simulated experimental data, we implemented the n-D lookup tables to locate the sensor coil. Next, we analyzed the results of the test by calculating the residual in the x-, y-, and z-directions that was less than 0.5 mm, as seen in **Figure 5.1. A, B, and C** respectively.

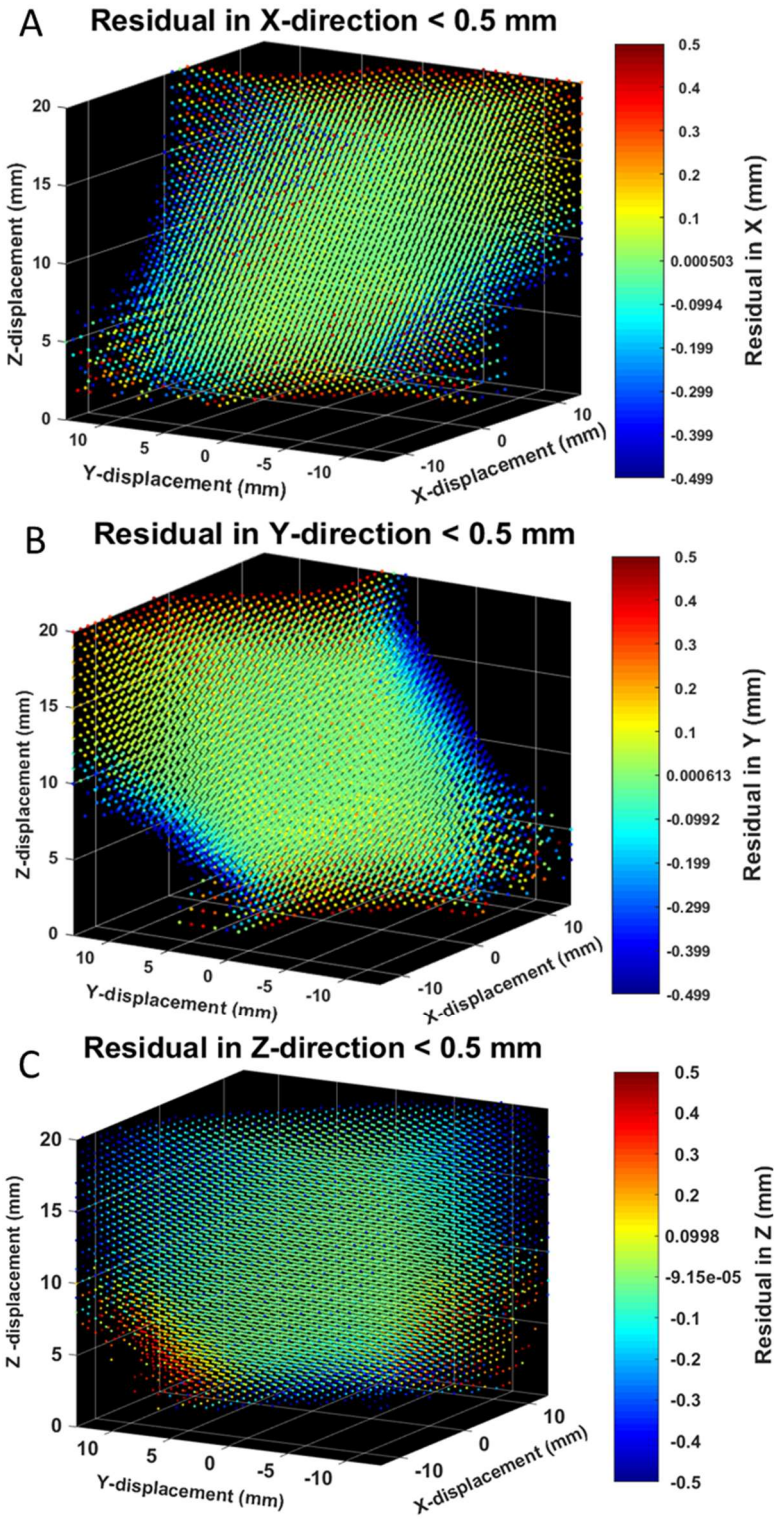


Figure 5.1.: (a) Results from simulated experimental data for the 1st, 2nd, and 3rd coil locations using the n-D lookup tables. Depicted the residual less than 0.5 mm from simulated experimental data in the x-direction (b) y-direction, and (c) z-direction.

5.2. Rotary Reader

We attached the rotary reader to the experimental setup used for the multicoil reader, as seen in **Figure 5.2.1.A**. Using a MATLAB script, the Bode 100 VNA perform the one-port reflection measurement to obtain the phase amplitude at the four coil locations with the sensor coil connected to a crystal oscillator only. With the conversion in (16), we calculated the mutual inductance at each coil location.

5.2.1 Coil Location Combination L1, L2, and L4

We imported three of the four measurements into the Simulink workspace for coil locations L1, L2, and L4 along with the corresponding assembled libraries. Once in Simulink, the three n-D lookup tables searched the assembled libraries to determine the x-, y-, and z-displacements of the sensor coil. We found the sensor coil location for the 13 different offsets in relative position between the rotary reader and sensor coil. **Figure 5.2.1.A, B, and C** depict the results for the 13 different offsets in relative positions between the rotary reader and sensor coil.

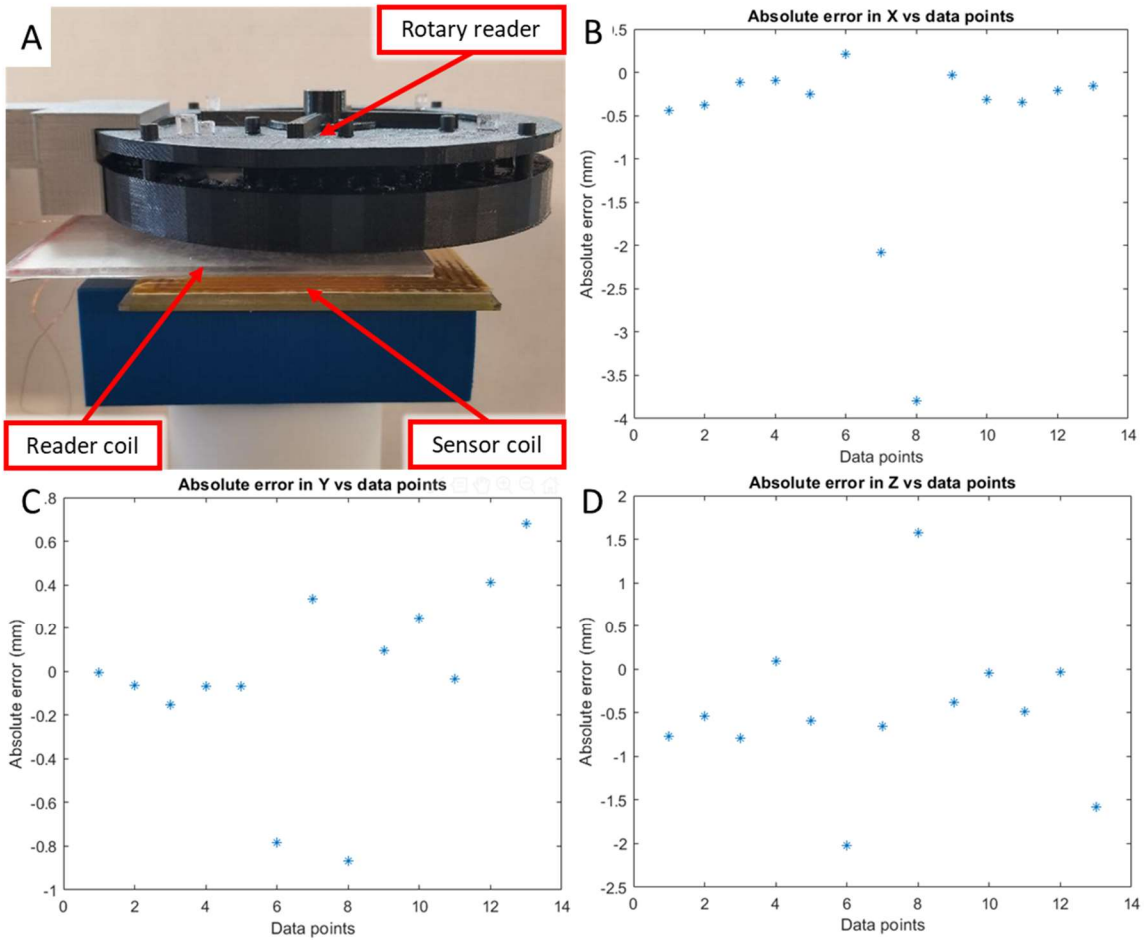


Figure 5.2.1. (a) Experimental setup for the rotary reader taking impedance measurements at the four locations each 47 mm apart in a square configuration. Experimental results from measurements taken at coil locations L1, L2, and L4 for the 13 different offsets in relative position between the rotary reader and sensor coil. Depicting the error between actual location and measured location in the (b) x-direction, (c) y-direction, and (d) z-direction.

Data point	x	y	z	C1-phase	C2-phase	C4-phase	m1	m2	m4	xc	yc	zc	Error -x	Error -y	Error-z
1.00	0	0	2	4.13	4.13	3.90	3.68	3.68	3.58	-0.44	0.00	2.77	0.44	0.00	0.77
2.00	0	0	5	3.41	3.43	3.25	3.34	3.36	3.27	-0.38	0.06	5.54	0.38	0.06	0.54
3.00	0	0	10	2.36	2.40	2.33	2.78	2.81	2.77	-0.12	0.15	10.79	0.12	0.15	0.79
4.00	0	0	15	1.81	1.82	1.79	2.44	2.44	2.42	-0.09	0.07	14.91	0.09	0.07	0.09
5.00	0	-5	5	4.40	2.41	4.26	3.80	2.81	3.74	-0.25	-5.07	5.60	0.25	0.07	0.60
6.00	5	0	5	2.30	2.10	4.10	2.75	2.63	3.67	4.79	-0.78	7.02	0.21	0.78	2.02
7.00	5	-5	5	2.61	1.50	6.03	2.93	2.22	4.45	7.08	-4.66	5.65	2.08	0.34	0.65
8.00	10	-10	5	2.23	0.59	9.93	2.71	1.39	5.71	13.80	-10.87	3.42	3.80	0.87	1.58
9.00	0	2	5	3.00	3.76	3.01	3.14	3.51	3.14	0.03	1.90	5.38	0.03	0.10	0.38
10.00	0	2	10	2.20	2.67	2.28	2.69	2.96	2.74	0.31	1.75	10.04	0.31	0.25	0.04
11.00	0	4	5	2.52	4.07	2.63	2.88	3.65	2.94	0.34	4.04	5.49	0.34	0.04	0.49
12.00	2	4	8	1.99	3.00	2.58	2.56	3.14	2.91	2.21	3.59	8.03	0.21	0.41	0.03
13.00	4	6	5	1.67	3.10	2.75	2.34	3.19	3.00	4.16	5.32	6.58	0.16	0.68	1.58

Table 5.2.1. Experimental results from measurements taken at coil locations L1, L2, and L4 for the 13 different offsets in relative position between the rotary reader and sensor coil. Five absolute errors in the x-, y- and z-directions exceeded 1 mm.

5.2.1.1. Interpolation Techniques for Simulation Data and "Scattered Interpolant"

Using the coil location combination L1, L2, and L4, we investigated different interpolation techniques for the simulation data and "scattered interpolant"; along with their effect on the final measurement of the sensor coil location. To accomplish this, we implemented the MATLAB script and Simulink file to locate the sensor coil using three different interpolation techniques for the simulation data. The three available techniques to interpolate the simulation data using **interp3** were "linear", "spline", and "makima". **Figure 5.2.1.1.A, B,** and **C** depict the absolute error of the sensor coil location for the three interpolation techniques applied to the simulation data in the x-, y-, and z-directions respectively. Once we found the optimal interpolation technique, we employed the MATLAB script again to discover the optimal technique for the "scattered interpolant" function. For the "scattered interpolant" we had two interpolation techniques available

"linear" and "natural". **Figure 5.2.1.1.D, E,** and **F** depict the absolute error of the sensor coil location for the two interpolation techniques applied to the "scattered interpolant" in the x-, y-, and z-directions respectively.

For the interpolation applied to the simulation data, we found the "linear" interpolation to produce the smallest error in sensor coil location. The "linear" method required the second least amount of computational time to complete using a 2.7 GHz i5 processor with 8 GB of ram. The computational times for "spline" was 117 ms, "linear" was 169 ms, and was "makima" taking 291 ms. The longer time taken by the "linear" method is acceptable due to the decrease in error seen in **Figure 5.2.1.1.B** and **C**, which shows a decrease in error of 1 mm and 2 mm for sample 8 in the y-direction and z-directions respectively. For the "scattered interpolant", we observed the "linear" method producing smaller errors, as seen in **Figure 5.2.1.1.D** and **E**, in the x-direction and y-direction. However, we saw the "natural" method procure a smaller error, as seen in **Figure 5.2.1.1.F**, in the z-direction. The "natural" method required a much longer computational time at 406.99 second while the "linear" technique required a computational time of 31.77 seconds. We concluded the "linear" method was the optimal choice for both the simulation data interpolation and the "scattered interpolant".

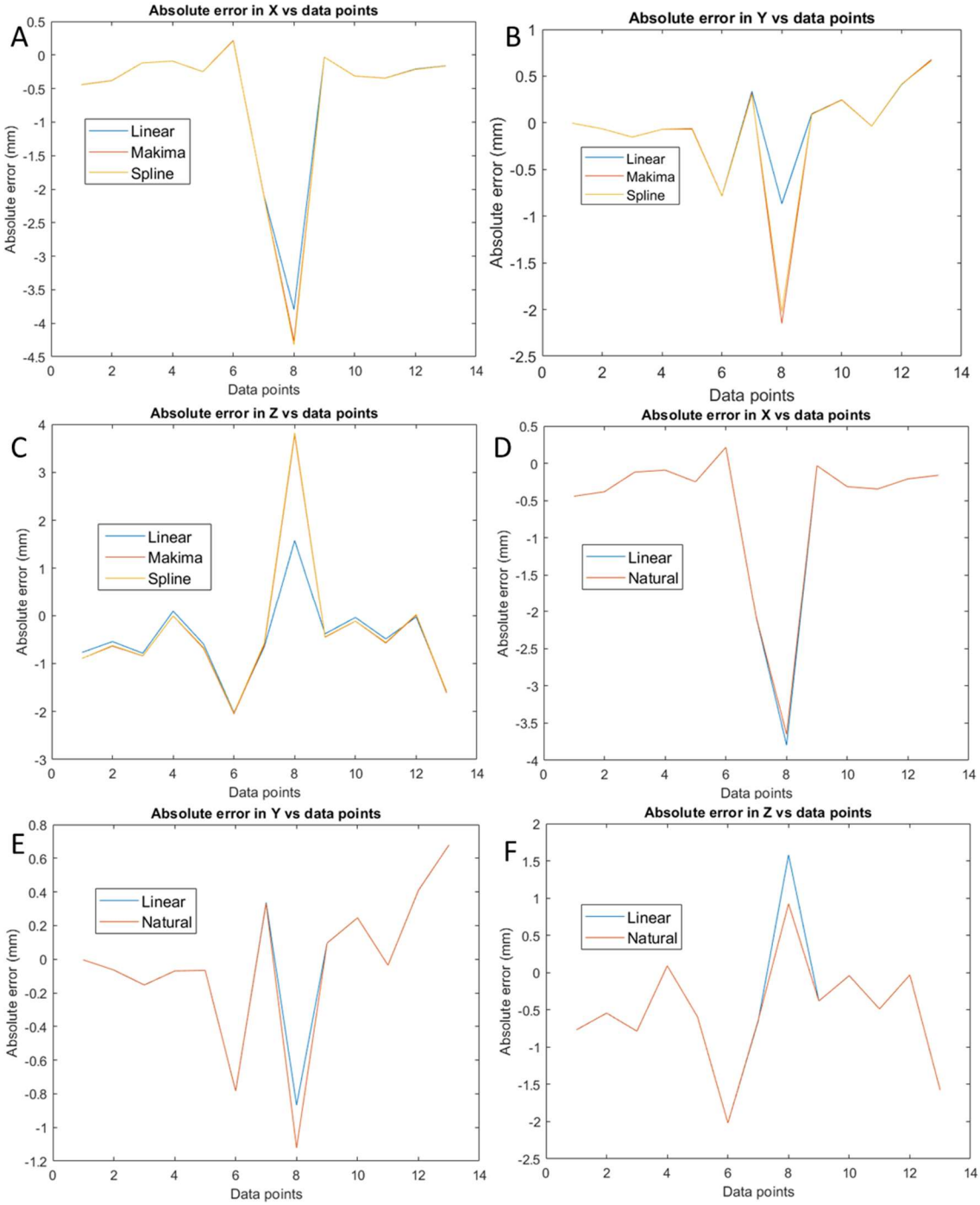


Figure 5.2.1.1.: Using coil locations L1, L2, and L4, implemented MATLAB script and Simulink file to search and interpolate the location of the sensor coil using three different interpolation techniques for the simulation data; “linear”, “spline”, and “makima”. The results for the three different interpolation techniques used for interpolating the simulation data to locate the sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction.

Discovered the optimal interpolation technique for the “scattered interpolant” of the two techniques; “linear” and “natural”. The results for the two different interpolation techniques used by the “scattered interpolant” to locate the sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction.

5.2.2. Coil Location Combination L1, L2, and L3

We repeat the process described in the previous section for the following coil location combination: L1, L2, and L3 as seen in **Figure 5.2.2.A**. **Figure 5.2.2.B, C, and D** depict the results for the 13 different offsets in relative positions between the rotary reader and sensor coil. The n-D lookup tables returned accurate values for the location of the sensing coil. Only four absolute errors in the x-, y- and z-directions exceeded 1 mm for the combination of coil locations L1, L2, and L3.

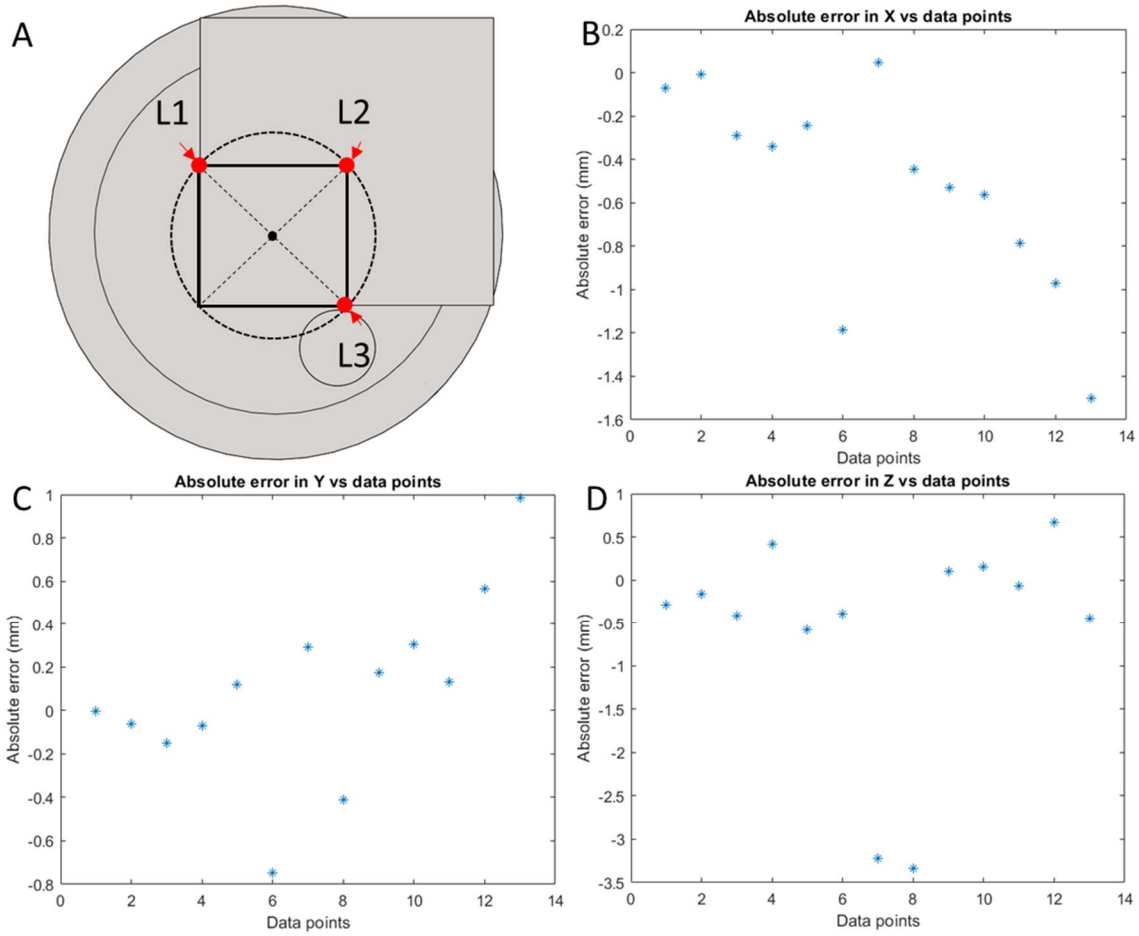


Figure 5.2.2. Error results from measurements taken at the (a) locations of coils L1, L2, and L3 for the 13 different offsets in relative position between the rotary reader and sensor coil. Depicting the error between actual location and measured location in the (b) x-direction, (c) y-direction, and (d) z-direction. The n-D lookup tables returned accurate values for the location of the sensing coil. Only four absolute errors in the x-, y- and z-directions exceeded 1 mm.

Data point	x	y	z	C1-phase	C2-phase	C3-phase	m1	m2	m3	xc	yc	zc	Error -x	Error -y	Error-z
1.00	0	0	2	4.13	4.13	4.16	3.68	3.68	3.70	-0.44	0.00	2.77	0.07	0.00	0.29
2.00	0	0	5	3.41	3.43	3.44	3.34	3.36	3.36	-0.38	0.06	5.54	0.01	0.06	0.16
3.00	0	0	10	2.36	2.40	2.48	2.78	2.81	2.85	-0.12	0.15	10.79	0.29	0.15	0.41
4.00	0	0	15	1.81	1.82	1.89	2.44	2.44	2.49	-0.09	0.07	14.91	0.34	0.07	0.42
5.00	0	-5	5	4.40	2.41	2.34	3.80	2.81	2.77	-0.25	-5.07	5.60	0.25	0.12	0.59
6.00	5	0	5	2.30	2.10	4.40	2.75	2.63	3.80	4.79	-0.78	7.02	1.19	0.75	0.40
7.00	5	-5	5	2.61	1.50	2.64	2.93	2.22	2.94	7.08	-4.66	5.65	0.05	0.29	3.22
8.00	10	-10	5	2.23	0.59	2.17	2.71	1.39	2.67	13.80	-10.87	3.42	0.45	0.41	3.33
9.00	0	2	5	3.00	3.76	4.01	3.14	3.51	3.63	0.03	1.90	5.38	0.53	0.18	0.10
10.00	0	2	10	2.20	2.67	2.84	2.69	2.96	3.06	0.31	1.75	10.04	0.56	0.31	0.16
11.00	0	4	5	2.52	4.07	4.47	2.88	3.65	3.83	0.34	4.04	5.49	0.79	0.13	0.07
12.00	2	4	8	1.99	3.00	4.24	2.56	3.14	3.73	2.21	3.59	8.03	0.97	0.56	0.66
13.00	4	6	5	1.67	3.10	5.80	2.34	3.19	4.37	4.16	5.32	6.58	1.50	0.99	0.44

Table 5.2.2. Experimental results from measurements taken at coil locations L1, L2, and L3 for the 13 different offsets in relative position between the rotary reader and sensor coil. Where x, y, and z are the chosen offsets in relative position; C1-, C2, and C3-Phase are the phase amplitudes from the impedance measurement; m1, m2, and m3 are the measured mutual inductances converted from the phase amplitude data; xc, yc, and zc are the locations found using the n-D lookup tables; and Error-x, -y, and -z are the errors in the corresponding directions. Only four absolute errors in the x-, y- and z-directions exceeded 1 mm for the combination of coil locations L1, L2, and L3.

5.2.3. Coil Location Combination L4, L2, and L3

We repeat the process described in the previous section for the following coil location combination: L4, L2, and L3 as seen in **Figure 5.2.3A**. **Figure 5.2.3.B, C, and D** depict the results for the 13 different offsets in relative positions between the rotary reader and sensor coil. The n-D lookup tables returned less accurate values than the combination of coil locations for L1, L2, and L3. Seven absolute errors in the x-, y- and z-directions exceeded 1 mm for the combination of coil locations L4, L2, and L3.

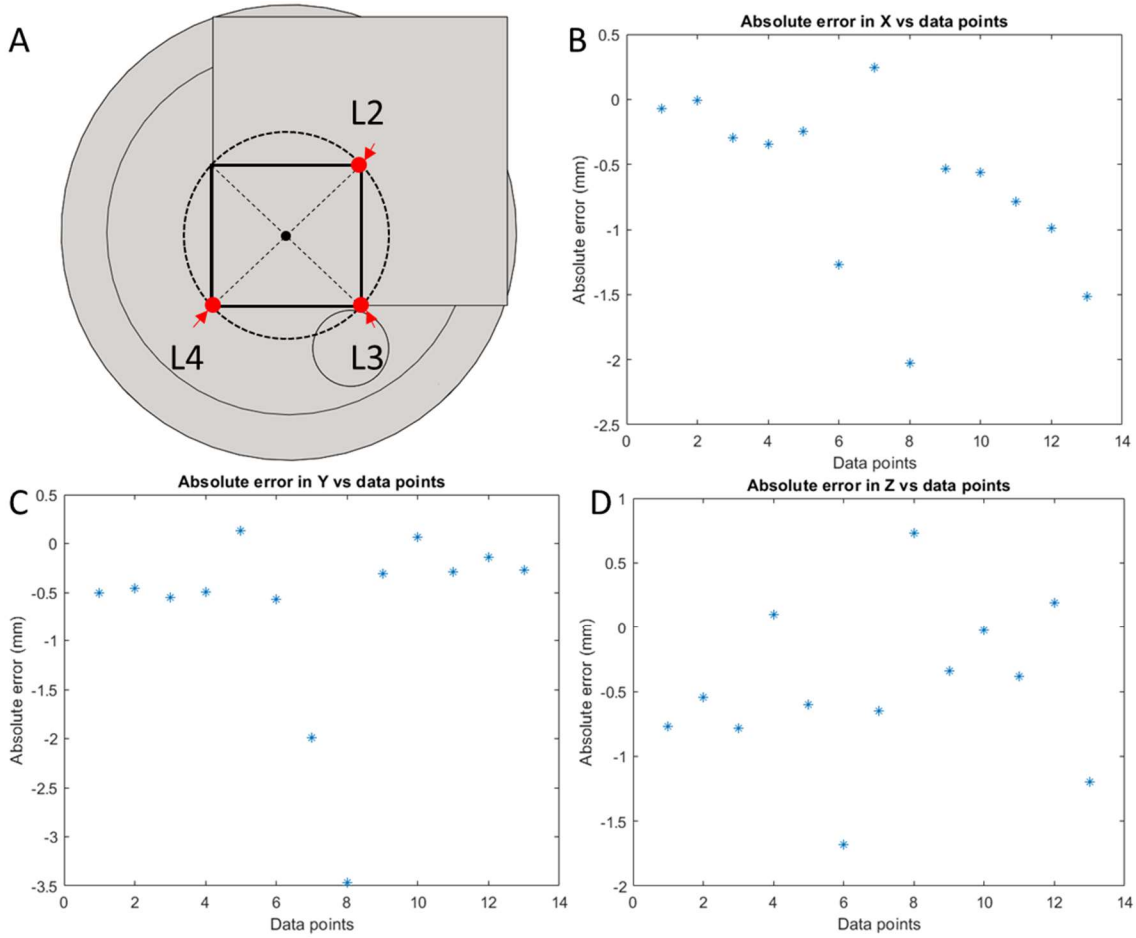


Figure 5.2.3. Error results from measurements taken at coil locations L4, L2, and L3 for the 13 different offsets in relative position between the rotary reader and sensor coil in the (a) x-direction, (b) y-direction, and (c) z-direction.

Data point	x	y	z	C4-phase	C2-phase	C3-phase	m4	m2	m3	xc	yc	zc	Error -x	Error -y	Error-z
1.00	0	0	2	3.90	4.13	4.16	3.58	3.68	3.70	0.07	0.51	2.77	0.07	0.51	0.77
2.00	0	0	5	3.25	3.43	3.44	3.27	3.36	3.36	0.01	0.45	5.54	0.01	0.45	0.54
3.00	0	0	10	2.33	2.40	2.48	2.77	2.81	2.85	0.29	0.55	10.78	0.29	0.55	0.78
4.00	0	0	15	1.79	1.82	1.89	2.42	2.44	2.49	0.34	0.49	14.90	0.34	0.49	0.10
5.00	0	-5	5	4.26	2.41	2.34	3.74	2.81	2.77	-0.25	-4.87	5.60	0.25	0.13	0.60
6.00	5	0	5	4.10	2.10	4.40	3.67	2.63	3.80	6.27	0.57	6.68	1.27	0.57	1.68
7.00	5	-5	5	6.03	1.50	2.64	4.45	2.22	2.94	4.76	-6.99	5.65	0.24	1.99	0.65
8.00	10	-10	5	9.93	0.59	2.17	5.71	1.39	2.67	12.03	-13.47	4.27	2.03	3.47	0.73
9.00	0	2	5	3.01	3.76	4.01	3.14	3.51	3.63	0.53	2.31	5.34	0.53	0.31	0.34
10.00	0	2	10	2.28	2.67	2.84	2.74	2.96	3.06	0.57	1.93	10.03	0.57	0.07	0.03
11.00	0	4	5	2.63	4.07	4.47	2.94	3.65	3.83	0.79	4.29	5.39	0.79	0.29	0.39
12.00	2	4	8	2.58	3.00	4.24	2.91	3.14	3.73	2.99	4.14	7.81	0.99	0.14	0.19
13.00	4	6	5	2.75	3.10	5.80	3.00	3.19	4.37	5.51	6.27	6.20	1.51	0.27	1.20

Table 5.2.3. Experimental results from measurements taken at coil locations L4, L2, and L3 for the 13 different offsets, labelled Data Points, in relative position between the rotary reader and sensor coil. Where x, y, and z are the chosen offsets in relative position; C4-, C2, and C3-Phase are the phase amplitudes from the impedance measurement; m4, m2, and m3 are the measured mutual inductances converted from the phase amplitude data; xc, yc, and zc are the locations found using the n-D lookup tables; and Error-x, -y, and -z are the errors in the corresponding directions. Seven absolute errors in the x-, y- and z-directions exceeded 1 mm for the coil locations L4, L2, and L3.

5.2.4. Average of Three Measurements

Lastly, we averaged the three different measurement combinations to produce a final result. The combinations of coil locations that we took the average of coil locations: L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. The final results had five locations that exceeded an error greater than 1 mm. **Figure 5.2.4.A, B,** and **C** represent the error for the average measurement in the x-, y-, and z-directions, respectively.

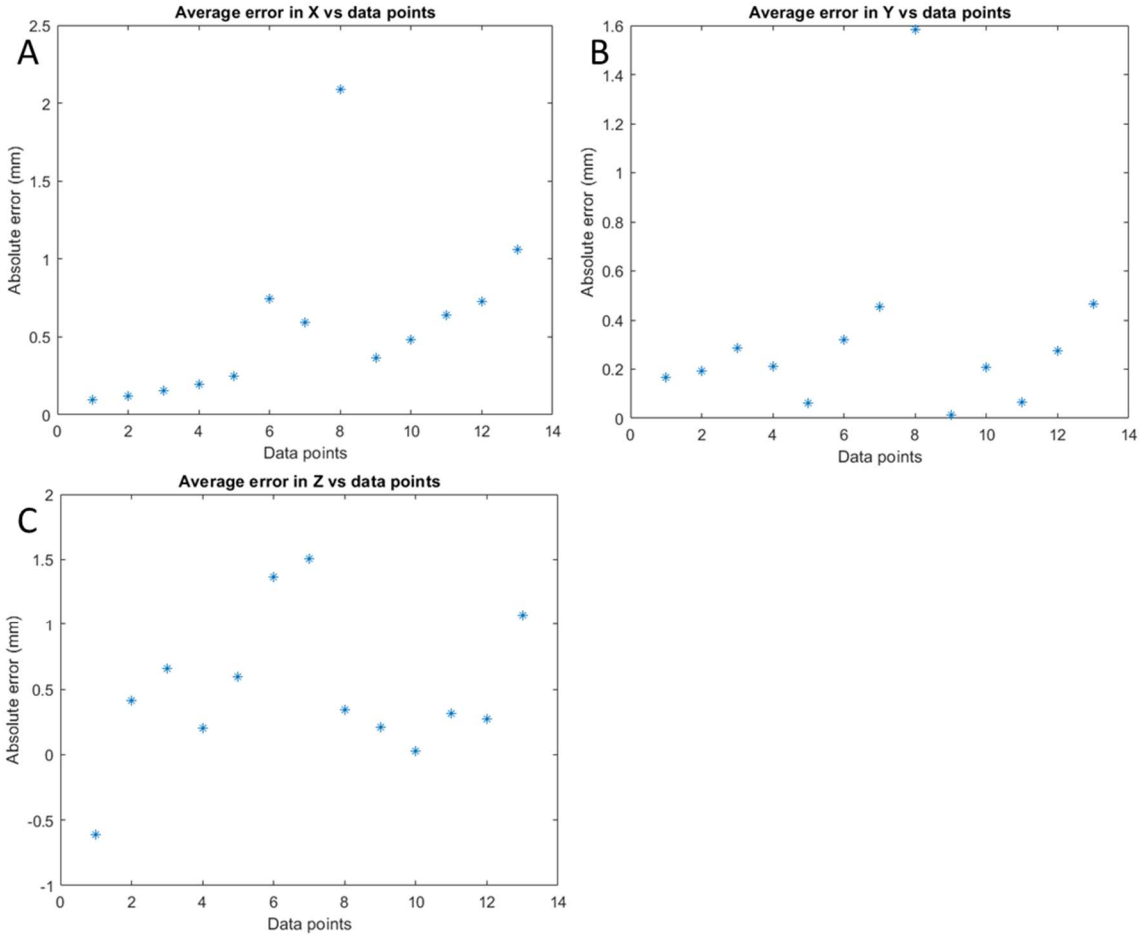


Figure 5.2.4. Experimental results from measurements taken at three different combinations of coil locations: L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. Calculated the average offset in relative position between the rotary reader and sensor coil in the (a) x-direction, (b) y-directions, and (c) z-. Had only five locations with an error exceeding 1 mm for the average location in each direction.

Data point	x	y	z	xc-Mean	yc-Mean	zc-Mean	Error -x	Error -y	Error-z
1.00	0	0	2	-0.10	0.17	2.61	0.10	0.17	0.61
2.00	0	0	5	-0.12	0.19	5.41	0.12	0.19	0.41
3.00	0	0	10	0.16	0.28	10.66	0.16	0.28	0.66
4.00	0	0	15	0.20	0.21	14.80	0.20	0.21	0.20
5.00	0	-5	5	-0.25	-4.94	5.59	0.25	0.06	0.59
6.00	5	0	5	5.75	-0.32	6.37	0.75	0.32	1.37
7.00	5	-5	5	5.60	-5.45	6.51	0.60	0.45	1.51
8.00	10	-10	5	12.09	-11.58	5.34	2.09	1.58	0.34
9.00	0	2	5	0.37	2.01	5.21	0.37	0.01	0.21
10.00	0	2	10	0.48	1.79	9.97	0.48	0.21	0.03
11.00	0	4	5	0.64	4.07	5.31	0.64	0.07	0.31
12.00	2	4	8	2.73	3.72	7.73	0.73	0.28	0.27
13.00	4	6	5	5.06	5.54	6.07	1.06	0.46	1.07

Table 5.2.4. Average of experimental results from measurements combinations of coil locations: L1, L2, and L3; L1, L2, and L4; and L4, L2, and L3. Results found the 13 different offsets, labelled Data Points, in relative position between the rotary reader and sensor coil. Where x, y, and z are the chosen offsets in relative position; xc-, yc-, and zc-Mean are the average locations found using the n-D lookup tables for the three different combinations of coil locations; and Error-x, -y, and -z are the errors in the corresponding directions. Five absolute errors in the x-, y- and z-directions exceeded 1 mm for the coil locations 4, 2, and 3.

6. Future and Ongoing Work

As discussed in the previous section, we have successfully tested the new rotary reader using the backward conversion and n-D lookup tables in Simulink. Next, we would test the rotary reader using the piezoelectric sensor as opposed to the crystal oscillator alone. To expand the rotary reader to measure the location of a sensor coil with a mounted sensor, we would measure the phase amplitude at a range of applied strains and create a library. To implement the reader with a sensor, we would first perform a calibration process where we obtain the resonant frequency of the sensor at a known location, which does not shift due to the relative position between the two coils. With the resonant frequency, we would know the corresponding phase amplitude and mutual inductance at a known location. This would allow for the implementation of the rotary reader.

Another improvement would be to find a transformation for the mutual inductance data that would allow the n-D lookup tables to search it. For the n-D lookup tables to work the data must be cubic, so if we could transform the pyramid-shaped mutual inductance data into a cube, we could use the n-D lookup tables without shortening the range. This would allow us the total range of +/- 40 mm in the x- and y-directions and 40 mm in the z-direction.

This work could also locate an embedded coil for other applications using low frequency ~2 MHz inductive coupling. We can apply this type of reader to any capacitive-based sensor system to measure wirelessly.

7. Appendix

7.1. MATLAB Scripts

7.1.1 Cylindrical Coil Analytical Approximation

MATLAB code of analytical calculation of mutual inductance using approximation (Heinzle)

```
clear, clc
h =[2e-3,3.55e-3,5.1e-3,6.65e-3,8.2e-3,9.75e-3];
N1 = 20 ;
N2 = 20;
r1 = 6.5e-3+.9e-3;
mu = pi*4*10^(-7);
for i = 1:length(h)
fun = @(x)
 besselj(1,r1.*x).*besselj(1,r1.*x).*exp(-1.*x.*(10e-3+h(i)));

M_temp = integral(fun,0,10^(4),'AbsTol',1e-12);

M(i) = mu*pi*r1*r1*N1*N2*M_temp*1e6;
end
disp(M)
plot(h,M,'--b')
xlabel('Separation (m)')
ylabel('Mutual Inductance (H)')
title('Analytical Calculation (Heinzle)')
```

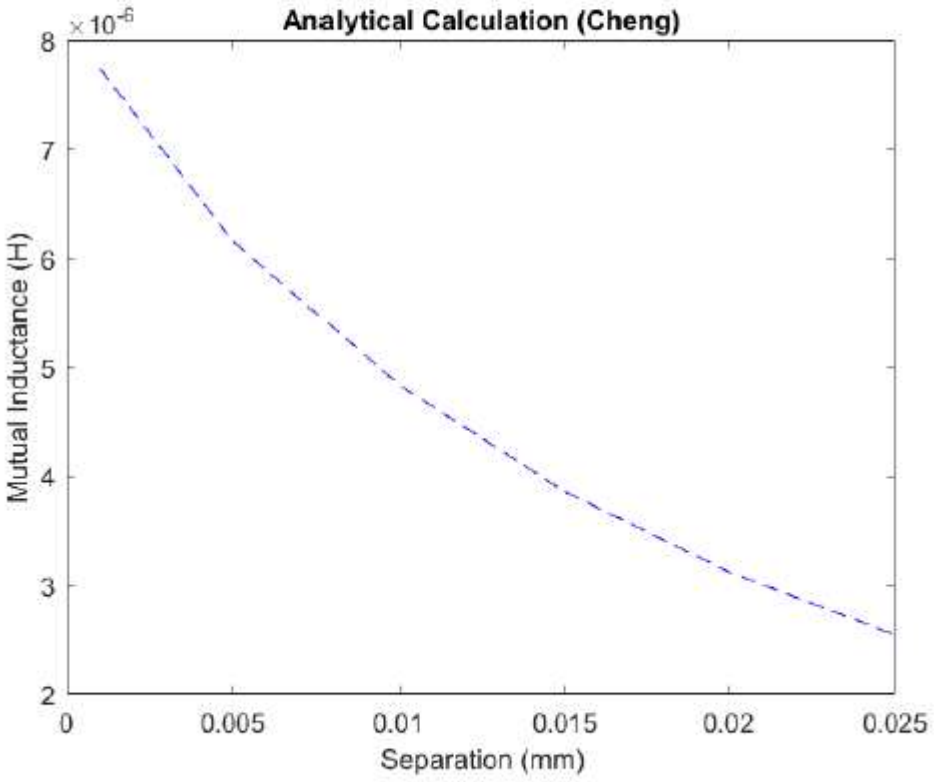
```
    0.6452    0.5058    0.4021    0.3237    0.2636    0.2170
```

7.1.2 Analytical Calculation of Mutual Inductance for Rectangular Coil

```
clear
clc
%width
w = 0.0015;
%Coil turn spacing
s = 0.002;
%Outer most coil width and length
a1 = 0.095/2;
b1 = 0.095/2;
c1 = 0.095/2;
d1 = 0.095/2;
%Number of turns
N = 13;
%Vertical separation
z = [.001; .005; .010; .015; .020; .025];
%Horizontal x-direction separation
x = 0;
%Horzontal y-direction separation
y = 0;
%Permeability of free space
u0 = 4*pi*10^(-7);
M =0;
for k = 1:length(z)
    for i = 1:N
        for j =1:N

            %a(i) is the legnth of coil A's turns
            a(i) = a1 - (i-1)*(w+s);
            %b(i) is the width of coil A's turns
            b(i) = b1 - (i-1)*(w+s);
            %c(j) is the length of coil B's turns
            c(j) = c1 - (j-1)*(w+s);
            %d(j) is the width of coil B's turns
            d(j) = d1 - (j-1)*(w+s);
            %Phicd is the effective magnetic flux of portion cd in
 the z direction
            Phicdz(i,j) = u0/
(2*pi)*(sqrt((b(i)+d(j))^2+z(k)^2+(a(i)+c(j))^2)-...
                (a(i)+c(j))*atanh((a(i)+c(j))/
(sqrt((b(i)+d(j))^2+z(k)^2+(a(i)+c(j))^2)))-...
                sqrt((b(i)+d(j))^2+z(k)^2+(a(i)-c(j))^2)+...
                (a(i)-c(j))*atanh((a(i)-c(j))/
(sqrt((b(i)+d(j))^2+z(k)^2+(a(i)-c(j))^2)))-...
                sqrt((b(i)-d(j))^2 +z(k)^2+(a(i)+c(j))^2)+...
                (a(i)+c(j))*atanh((a(i)+c(j))/(sqrt((b(i)-
d(j))^2+z(k)^2+(a(i)+c(j))^2)))+...
                sqrt((b(i)-d(j))^2+z(k)^2+(a(i)-c(j))^2)-...
                (a(i)-c(j))*atanh((a(i)-c(j))/sqrt((b(i)-
d(j))^2+z(k)^2 +(a(i)-c(j))^2)));
        end
    end
    Mut(k) = sum(sum(Phicdz))*4;
```

```
end
%Mutual Inductance of two rectangular coils
disp(Mut)
plot(z,Mut,'--b')
xlabel('Separation (mm)')
ylabel('Mutual Inductance (H)')
title('Analytical Calculation (Cheng)')

%Multiplied by four for each segment of the coil
```

```
   1.0e-05 *

    0.7744    0.6168    0.4838    0.3860    0.3119    0.2546
```



Published with MATLAB® R2016b

7.1.3. Multicoil Reader Script

Measurement Script for Bode 100 VNA and control of the multicoil reader [36]

```
clear all
close all
format long e
%%
Freq_Low = 1842700;
Freq_High = 1843000;
Num_Pts = 401;
pt = 20;
FitFreqLow = 1842800;
FirFreqHigh= 1843000;

%% Connect Arduino

% a=arduino('COM5','Uno');

%% Connect to Bode 100 and set sweep parameters

automationInterface =
actxserver('OmicronLab.VectorNetworkAnalysis.AutomationInterface');
bode = automationInterface.Connect();
s11 = bode.Impedance.CreateOnePortMeasurement();
s11.ConfigureSweep(Freq_Low,Freq_High, Num_Pts,
'SweepMode_Logarithmic');
s11.ReceiverBandwidth = 'ReceiverBandwidth_Hz30';
s11.SetSourceLevel(13,'LevelUnit_dBm');

%% Coil 1 - closest to the surface, top left - Set relay positions,
execute read, and record/plot data
%
% %Close relays
% writeDigitalPin(a,'D2',0);
% writeDigitalPin(a,'D9',0);
% writeDigitalPin(a,'D3',1);
% writeDigitalPin(a,'D8',1);
% writeDigitalPin(a,'D4',1);
% writeDigitalPin(a,'D7',1);
% writeDigitalPin(a,'D5',1);
% writeDigitalPin(a,'D6',1);
%%
%Execute measurement
state = s11.ExecuteMeasurement();
resultFrequencyC1 = s11.Results.MeasurementFrequencies;
resultMagnitudeC1 = s11.Results.Magnitude('MagnitudeUnit_dB');
resultPhaseC1 = s11.Results.Phase('AngleUnit_Degree');
%%
% Fitting
F_C1 = resultFrequencyC1;
Y_C1 = resultPhaseC1;
 index_countm=size(F_C1);
    index_count=index_countm(2);
```

```
    x_fit_C1 = F_C1';
    y_fit_C1 = Y_C1';
         f_fit_C1 = fit(x_fit_C1,y_fit_C1,'gauss2',...          % SENSOR
1
    'Lower',      [60,      1842000,    0,          -500,
FitFreqLow,       -Inf],...
    'Upper',      [90,      1843000,    Inf,        0,
FirFreqHigh,        Inf],...
    'StartPoint',[70,   FitFreqLow,    1842800,    -52,    FirFreqHigh,
400]);
%      f_fit_C1 = fit(x_fit_C1,y_fit_C1,'gauss2',...          % SENSOR 1
%     'Lower',      [84,      1830000,    0,          -2,
FitFreqLow,       -Inf],...
%     'Upper',      [91,      1870000,    Inf,        2,
FirFreqHigh,        Inf],...
%     'StartPoint',[86,   FitFreqLow,    1776000,    -0.05,
FirFreqHigh,   400]);
    phase_fit_C1=zeros(index_count,1);
    for n=1:index_count
    phase_fit_C1(n) = f_fit_C1.a1*exp(-((F_C1(1,n)-
f_fit_C1.b1)/f_fit_C1.c1)^2) +...
                 f_fit_C1.a2*exp(-((F_C1(1,n)-
f_fit_C1.b2)/f_fit_C1.c2)^2);
    end

 % find minimum/maximum point of phase
        [Mini_Y_C1,Mini_index_C1]=min(phase_fit_C1);
        [Mini_Y2_C1,Mini_index2_C1]=min(Y_C1);
        [Max_Y_C1,Max_index_C1]=max(phase_fit_C1);
        [Max_Y2_C1,Max_index2_C1]=max(Y_C1);
        Mini_freq_C1=F_C1(Mini_index_C1);
        Mini_freq2_C1=F_C1(Mini_index2_C1);
        check_size=size(Mini_freq_C1);
        if check_size(1)==0
           Mini_freq_C1=0;
        end
 figure
 subplot(2,1,1)
 plot(resultFrequencyC1,resultMagnitudeC1);
 title('Coil 1 Impedance')
 xlabel('Frequency')
 ylabel('Magnitude in dB')
 subplot(2,1,2)
 plot(resultFrequencyC1, resultPhaseC1,resultFrequencyC1,phase_fit_C1);
 xlabel('Frequency')
 ylabel('Phase in degree')
 pause (pt)

disp('Coil 1')
%% Coil 2 - 2nd to the surface, top right - Set relay positions,
execute read, and record/plot data
% writeDigitalPin(a,'D2',1);
% writeDigitalPin(a,'D9',1);
% writeDigitalPin(a,'D3',0);
% writeDigitalPin(a,'D8',0);
% writeDigitalPin(a,'D4',1);
% writeDigitalPin(a,'D7',1);
```

```
% writeDigitalPin(a,'D5',1);
% writeDigitalPin(a,'D6',1);

state = s11.ExecuteMeasurement();
resultFrequencyC2 = s11.Results.MeasurementFrequencies;
resultMagnitudeC2 = s11.Results.Magnitude('MagnitudeUnit_dB');
resultPhaseC2 = s11.Results.Phase('AngleUnit_Degree');
%%
% Fitting
F_C2 = resultFrequencyC2;
Y_C2 = resultPhaseC2;
 index_countm=size(F_C2);
    index_count=index_countm(2);
    x_fit_C2 = F_C2';
    y_fit_C2 = Y_C2';
     f_fit_C2 = fit(x_fit_C2,y_fit_C2,'gauss2',...          % SENSOR 2
    'Lower',     [60,      1800000,    0,          -22,
FitFreqLow,       -Inf],...
    'Upper',     [91,      1900000,    Inf,        0,
FirFreqHigh,        Inf],...
    'StartPoint',[82,  FitFreqLow,    1776000,    -0.5,    FirFreqHigh,
400]);
    phase_fit_C2=zeros(index_count,1);
    for n=1:index_count
    phase_fit_C2(n) = f_fit_C2.a1*exp(-((F_C2(1,n)-
f_fit_C2.b1)/f_fit_C2.c1)^2) +...
                f_fit_C2.a2*exp(-((F_C2(1,n)-
f_fit_C2.b2)/f_fit_C2.c2)^2);
    end
 % find minimum/maximum point of phase
        [Mini_Y_C2,Mini_index_C2]=min(phase_fit_C2);
        [Mini_Y2_C2,Mini_index2_C2]=min(Y_C2);
        [Max_Y_C2,Max_index_C2]=max(phase_fit_C2);
        [Max_Y2_C2,Max_index2_C2]=max(Y_C2);
        Mini_freq_C2=F_C2(Mini_index_C2);
        Mini_freq2_C2=F_C2(Mini_index2_C2);
        check_size=size(Mini_freq_C2);
        if check_size(1)==0
           Mini_freq_C2=0;
        end
 figure
 subplot(2,1,1)
 plot(resultFrequencyC2,resultMagnitudeC2);
 title('Coil 2 Impedance')
 xlabel('Frequency')
 ylabel('Magnitude in dB')
 subplot(2,1,2)
 plot(resultFrequencyC2, resultPhaseC2,resultFrequencyC2,phase_fit_C2);
 xlabel('Frequency')
 ylabel('Phase in degree')
pause (pt)
disp('Coil 2')
%% Coil 3 - 3rd to the surface, bottom right - Set relay positions,
execute read, and record/plot data
% writeDigitalPin(a,'D2',1);
% writeDigitalPin(a,'D9',1);
% writeDigitalPin(a,'D3',1);
```

```
% writeDigitalPin(a,'D8',1);
% writeDigitalPin(a,'D4',0);
% writeDigitalPin(a,'D7',0);
% writeDigitalPin(a,'D5',1);
% writeDigitalPin(a,'D6',1);

state = s11.ExecuteMeasurement();
resultFrequencyC3 = s11.Results.MeasurementFrequencies;
resultMagnitudeC3 = s11.Results.Magnitude('MagnitudeUnit_dB');
resultPhaseC3 = s11.Results.Phase('AngleUnit_Degree');
%%
% Fitting
F_C3 = resultFrequencyC3;
Y_C3 = resultPhaseC3;
 index_countm=size(F_C3);
    index_count=index_countm(2);
    x_fit_C3 = F_C3';
    y_fit_C3 = Y_C3';
     f_fit_C3 = fit(x_fit_C3,y_fit_C3,'gauss2',...          % SENSOR 3
    'Lower',     [60,      1800000,    0,          -22,
FitFreqLow,       -Inf],...
    'Upper',     [91,      1900000,    Inf,        0,
FirFreqHigh,        Inf],...
    'StartPoint',[75,   FitFreqLow,    1776000,    -0.5,
FirFreqHigh,   400]);
    phase_fit_C3=zeros(index_count,1);
    for n=1:index_count
    phase_fit_C3(n) = f_fit_C3.a1*exp(-((F_C3(1,n)-
f_fit_C3.b1)/f_fit_C3.c1)^2) +...
                f_fit_C3.a2*exp(-((F_C3(1,n)-
f_fit_C3.b2)/f_fit_C3.c2)^2);
    end
 % find minimum/maximum point of phase
        [Mini_Y_C3,Mini_index_C3]=min(phase_fit_C3);
        [Mini_Y2_C3,Mini_index2_C3]=min(Y_C3);
        [Max_Y_C3,Max_index_C3]=max(phase_fit_C3);
        [Max_Y2_C3,Max_index2_C3]=max(Y_C3);
        Mini_freq_C3=F_C3(Mini_index_C3);
        Mini_freq2_C3=F_C3(Mini_index2_C3);
        check_size=size(Mini_freq_C3);
        if check_size(1)==0
           Mini_freq_C3=0;
        end
 figure
 subplot(2,1,1)
 plot(resultFrequencyC3,resultMagnitudeC3);
 title('Coil 3 Impedance')
 xlabel('Frequency')
 ylabel('Magnitude in dB')
 subplot(2,1,2)
 plot(resultFrequencyC3, resultPhaseC3,resultFrequencyC3,phase_fit_C3);
 xlabel('Frequency')
 ylabel('Phase in degree')
pause (pt)
disp('Coil 3')
%% Coil 4 - 4rd to the surface, bottom left - Set relay positions,
execute read, and record/plot data
```

```
% writeDigitalPin(a,'D2',1);
% writeDigitalPin(a,'D9',1);
% writeDigitalPin(a,'D3',1);
% writeDigitalPin(a,'D8',1);
% writeDigitalPin(a,'D4',1);
% writeDigitalPin(a,'D7',1);
% writeDigitalPin(a,'D5',0);
% writeDigitalPin(a,'D6',0);


state = s11.ExecuteMeasurement();
resultFrequencyC4 = s11.Results.MeasurementFrequencies;
resultMagnitudeC4 = s11.Results.Magnitude('MagnitudeUnit_dB');
resultPhaseC4 = s11.Results.Phase('AngleUnit_Degree');
%%
% Fitting
F_C4 = resultFrequencyC4;
Y_C4 = resultPhaseC4;
 index_countm=size(F_C4);
    index_count=index_countm(2);
    x_fit_C4 = F_C4';
    y_fit_C4 = Y_C4';
     f_fit_C4 = fit(x_fit_C4,y_fit_C4,'gauss2',...          % SENSOR 4
    'Lower',     [60,      1800000,    0,          -22,
FitFreqLow,       -Inf],...
    'Upper',     [91,      1900000,    Inf,        0,
FirFreqHigh,        Inf],...
    'StartPoint',[87.9,   FitFreqLow,    1776000,    -0.5,
FirFreqHigh,   400]);
    phase_fit_C4=zeros(index_count,1);
    for n=1:index_count
    phase_fit_C4(n) = f_fit_C4.a1*exp(-((F_C4(1,n)-
f_fit_C4.b1)/f_fit_C4.c1)^2) +...
                f_fit_C4.a2*exp(-((F_C4(1,n)-
f_fit_C4.b2)/f_fit_C4.c2)^2);
    end
 % find minimum/maximum point of phase
        [Mini_Y_C4,Mini_index_C4]=min(phase_fit_C4);
        [Mini_Y2_C4,Mini_index2_C4]=min(Y_C4);
        [Max_Y_C4,Max_index_C4]=max(phase_fit_C4);
        [Max_Y2_C4,Max_index2_C4]=max(Y_C4);
        Mini_freq_C4=F_C4(Mini_index_C4);
        Mini_freq2_C4=F_C4(Mini_index2_C4);
        check_size=size(Mini_freq_C4);
        if check_size(1)==0
           Mini_freq_C4=0;
        end
 figure
 subplot(2,1,1)
 plot(resultFrequencyC4,resultMagnitudeC4);
 title('Coil 4 Impedance')
 xlabel('Frequency')
 ylabel('Magnitude in dB')
 subplot(2,1,2)
 plot(resultFrequencyC4, resultPhaseC4,resultFrequencyC4,phase_fit_C4);
 xlabel('Frequency')
 ylabel('Phase in degree')
```

```
 disp('Coil 4')
 bode.ShutDown();
%  release(automationInterface);

%% Log Data

TableC1 =
table(resultFrequencyC1',resultPhaseC1',resultMagnitudeC1',phase_fit_C1
);
filename = sprintf('testC1_%s.xlsx', datestr(now,'mm-dd-yyyy HH-MM'));
writetable(TableC1,filename,'Sheet',1,'Range','A1')
TableC2 =
table(resultFrequencyC2',resultPhaseC2',resultMagnitudeC2',phase_fit_C2
);
filename = sprintf('testC2_%s.xlsx', datestr(now,'mm-dd-yyyy HH-MM'));
writetable(TableC2,filename,'Sheet',1,'Range','A1')
TableC3 =
table(resultFrequencyC3',resultPhaseC3',resultMagnitudeC3',phase_fit_C3
);
filename = sprintf('testC3_%s.xlsx', datestr(now,'mm-dd-yyyy HH-MM'));
writetable(TableC3,filename,'Sheet',1,'Range','A1')
TableC4 =
table(resultFrequencyC4',resultPhaseC4',resultMagnitudeC4',phase_fit_C4
);
filename = sprintf('testC4_%s.xlsx', datestr(now,'mm-dd-yyyy HH-MM'));
writetable(TableC4,filename,'Sheet',1,'Range','A1')
 %% Kinematics

 %Fit Phase
 Del_Phase_C1 = Max_Y_C1-Mini_Y_C1;
 Del_Phase_C2 = Max_Y_C2-Mini_Y_C2;
 Del_Phase_C3 = Max_Y_C3-Mini_Y_C3;
 Del_Phase_C4 = Max_Y_C4-Mini_Y_C4;

 %Original Data
 Del_Phase2_C1 = Max_Y2_C1-Mini_Y2_C1;
 Del_Phase2_C2 = Max_Y2_C2-Mini_Y2_C2;
 Del_Phase2_C3 = Max_Y2_C3-Mini_Y2_C3;
 Del_Phase2_C4 = Max_Y2_C4-Mini_Y2_C4;

x = 23.5;
y = 23.5;
z = 2;
 TableDelPhase =
table(Del_Phase_C1,Del_Phase_C2,Del_Phase_C3,Del_Phase_C4,Del_Phase2_C1
,Del_Phase2_C2,Del_Phase2_C3,Del_Phase2_C4,Mini_freq_C1,Mini_freq_C2,Mi
ni_freq_C3,Mini_freq_C4,x,y,z);
filename = sprintf('testDelPhase_%s.xlsx', datestr(now,'mm-dd-yyyy HH-
MM'));
writetable(TableDelPhase,filename,'Sheet',1,'Range','A1')
```

7.1.3.1 Mutual Inductance Reverse Lookup Algorithm

MATLAB script that implements the n-D lookup tables, kinematics of reader, and library of mutual inductance data

```
clear
clc
%% Mutual Inductance Data
%Z=2mm
M(:,:,1)=[0.09349   0.25583 0.42607 0.60115 0.78516 0.97458
1.16288 1.33688 1.47421 1.55475 1.47421 1.33688 1.16288 0.97458
0.78516 0.60115 0.42607 0.25583 0.09349
          0.25583   0.55627 0.8176  1.07671 1.33219 1.58417
1.8262  2.04238 2.20927 2.37994 2.20927 2.04238 1.8262  1.58417
1.33219 1.07671 0.8176  0.55627 0.25583
          0.42607   0.85611 1.22762 1.58883 1.93325 2.26033
2.56119 2.82318 3.0218  3.20879 3.0218  2.82318 2.56119 2.26033
1.93325 1.58883 1.22762 0.85611 0.42607
          0.60115   1.14099 1.62585 2.10564 2.55739 2.97194
3.34099 3.6553  3.88676 4.07795 3.88676 3.6553  3.34099 2.97194
2.55739 2.10564 1.62585 1.14099 0.60115
          0.78516   1.41002 1.99473 2.58977 3.1611  3.68148
4.13347 4.5078  4.77585 4.96514 4.77585 4.5078  4.13347 3.68148
3.1611  2.58977 1.99473 1.41002 0.78516
          0.97458   1.66707 2.33528 3.02622 3.70911 4.34837
4.89992 5.34607 5.65595 5.84093 5.65595 5.34607 4.89992 4.34837
3.70911 3.02622 2.33528 1.66707 0.97458
          1.16288   1.90612 2.63881 3.40458 4.17637 4.91848
5.57892 6.11311 6.47472 6.65741 6.47472 6.11311 5.57892 4.91848
4.17637 3.40458 2.63881 1.90612 1.16288
          1.33688   2.12004 2.90209 3.72383 4.55968 5.3774
6.12061 6.75392 7.17757 7.36597 7.17757 6.75392 6.12061 5.3774
4.55968 3.72383 2.90209 2.12004 1.33688
          1.47421   2.28594 3.10085 3.95692 4.83114 5.69336
6.49307 7.18563 7.67508 7.88157 7.67508 7.18563 6.49307 5.69336
4.83114 3.95692 3.10085 2.28594 1.47421
          1.55475   2.37994 3.20879 4.07795 4.96514 5.84093
6.65741 7.36597 7.88157 8.10734 7.88157 7.36597 6.65741 5.84093
4.96514 4.07795 3.20879 2.37994 1.55475
          1.47421   2.28594 3.10085 3.95692 4.83114 5.69336
6.49307 7.18563 7.67508 7.88157 7.67508 7.18563 6.49307 5.69336
4.83114 3.95692 3.10085 2.28594 1.47421
          1.33688   2.12004 2.90209 3.72383 4.55968 5.3774
6.12061 6.75392 7.17757 7.36597 7.17757 6.75392 6.12061 5.3774
4.55968 3.72383 2.90209 2.12004 1.33688
          1.16288   1.90612 2.63881 3.40458 4.17637 4.91848
5.57892 6.11311 6.47472 6.65741 6.47472 6.11311 5.57892 4.91848
4.17637 3.40458 2.63881 1.90612 1.16288
          0.97458   1.66707 2.33528 3.02622 3.70911 4.34837
4.89992 5.34607 5.65595 5.84093 5.65595 5.34607 4.89992 4.34837
3.70911 3.02622 2.33528 1.66707 0.97458
```

```
          0.78516   1.41002 1.99473 2.58977 3.1611  3.68148
4.13347 4.5078  4.77585 4.96514 4.77585 4.5078  4.13347 3.68148
3.1611  2.58977 1.99473 1.41002 0.78516
          0.60115   1.14099 1.62585 2.10564 2.55739 2.97194
3.34099 3.6553  3.88676 4.07795 3.88676 3.6553  3.34099 2.97194
2.55739 2.10564 1.62585 1.14099 0.60115
          0.42607   0.85611 1.22762 1.58883 1.93325 2.26033
2.56119 2.82318 3.0218  3.20879 3.0218  2.82318 2.56119 2.26033
1.93325 1.58883 1.22762 0.85611 0.42607
          0.25583   0.55627 0.8176  1.07671 1.33219 1.58417
1.8262  2.04238 2.20927 2.37994 2.20927 2.04238 1.8262  1.58417
1.33219 1.07671 0.8176  0.55627 0.25583
          0.09349   0.25583 0.42607 0.60115 0.78516 0.97458
1.16288 1.33688 1.47421 1.55475 1.47421 1.33688 1.16288 0.97458
0.78516 0.60115 0.42607 0.25583 0.09349];
%Z=3mm
 M(:,:,2)=  [0.10894    0.27185 0.44225 0.61651 0.7987  0.98318
1.16487 1.32932 1.45869 1.53003 1.45869 1.32932 1.16487 0.98318
0.7987  0.61651 0.44225 0.27185 0.10894
            0.27185 0.56122 0.81767 1.07223 1.32369 1.56935
1.80242 2.0074  2.23661 2.25046 2.23661 2.0074  1.80242 1.56935
1.32369 1.07223 0.81767 0.56122 0.27185
            0.44225 0.85141 1.20949 1.56015 1.89615 2.21338
2.50379 2.75283 2.94022 3.03919 2.94022 2.75283 2.50379 2.21338
1.89615 1.56015 1.20949 0.85141 0.44225
            0.61651 1.1299  1.59246 2.0521  2.4883  2.88915
3.24548 3.54415 3.76364 3.87487 3.76364 3.54415 3.24548 2.88915
2.4883  2.0521  1.59246 1.1299  0.61651
            0.7987  1.39445 1.95101 2.51757 3.06319 3.56191
3.99695 4.35212 4.60693 4.73001 4.60693 4.35212 3.99695 3.56191
3.06319 2.51757 1.95101 1.39445 0.7987
            0.98318 1.64502 2.28067 2.93758 3.58599 4.19141
4.71822 5.14095 5.43573 5.57106 5.43573 5.14095 4.71822 4.19141
3.58599 2.93758 2.28067 1.64502 0.98318
            1.16487 1.87664 2.57516 3.30394 4.03599 4.73584
5.35987 5.86143 6.2037  6.35263 6.2037  5.86143 5.35987 4.73584
4.03599 3.30394 2.57516 1.87664 1.16487
            1.32932 2.08013 2.82556 3.60749 4.40004 5.17023
5.87348 6.45555 6.85397 7.01834 6.85397 6.45555 5.87348 5.17023
4.40004 3.60749 2.82556 2.08013 1.32932
            1.45869 2.23661 3.01363 3.82879 4.65813 5.47031
6.22199 6.85953 7.31046 7.49252 7.31046 6.85953 6.22199 5.47031
4.65813 3.82879 3.01363 2.23661 1.45869
            1.53003 2.32097 3.11045 3.9374  4.77847 5.60241
6.36785 7.02064 7.49003 7.67905 7.49003 7.02064 6.36785 5.60241
4.77847 3.9374  3.11045 2.32097 1.53003
            1.45869 2.23661 3.01363 3.82879 4.65813 5.47031
6.22199 6.85953 7.31046 7.49252 7.31046 6.85953 6.22199 5.47031
4.65813 3.82879 3.01363 2.23661 1.45869
            1.32932 2.08013 2.82556 3.60749 4.40004 5.17023
5.87348 6.45555 6.85397 7.01834 6.85397 6.45555 5.87348 5.17023
4.40004 3.60749 2.82556 2.08013 1.32932
```

```
            1.16487 1.87664 2.57516 3.30394 4.03599 4.73584
5.35987 5.86143 6.2037  6.35263 6.2037  5.86143 5.35987 4.73584
4.03599 3.30394 2.57516 1.87664 1.16487
            0.98318 1.64502 2.28067 2.93758 3.58599 4.19141
4.71822 5.14095 5.43573 5.57106 5.43573 5.14095 4.71822 4.19141
3.58599 2.93758 2.28067 1.64502 0.98318
            0.7987  1.39445 1.95101 2.51757 3.06319 3.56191
3.99695 4.35212 4.60693 4.73001 4.60693 4.35212 3.99695 3.56191
3.06319 2.51757 1.95101 1.39445 0.7987
            0.61651 1.1299  1.59246 2.0521  2.4883  2.88915
3.24548 3.54415 3.76364 3.87487 3.76364 3.54415 3.24548 2.88915
2.4883  2.0521  1.59246 1.1299  0.61651
            0.44225 0.85141 1.20949 1.56015 1.89615 2.21338
2.50379 2.75283 2.94022 3.03919 2.94022 2.75283 2.50379 2.21338
1.89615 1.56015 1.20949 0.85141 0.44225
            0.27185 0.56122 0.81767 1.07223 1.32369 1.56935
1.80242 2.0074  2.23661 2.25046 2.23661 2.0074  1.80242 1.56935
1.32369 1.07223 0.81767 0.56122 0.27185
            0.10894 0.27185 0.44225 0.61651 0.7987  0.98318
1.16487 1.32932 1.45869 1.53003 1.45869 1.32932 1.16487 0.98318
0.7987  0.61651 0.44225 0.27185 0.10894];
%Z=4mm

 M(:,:,3)= [0.12453 0.28697 0.45672 0.63069 0.8097  0.98977
1.16468 1.321   1.44212 1.50714 1.44212 1.321   1.16468 0.98977
0.8097  0.63069 0.45672 0.28697 0.12453
            0.28697 0.5657  0.81658 1.06657 1.31285 1.55174
1.77605 1.97111 2.18674 2.26363 2.18674 1.97111 1.77605 1.55174
1.31285 1.06657 0.81658 0.5657  0.28697
            0.45672 0.84645 1.19176 1.53192 1.85864 2.16589
2.44565 2.68316 2.92777 3.01636 2.92777 2.68316 2.44565 2.16589
1.85864 1.53192 1.19176 0.84645 0.45672
            0.63069 1.11836 1.56046 2.00123 2.42151 2.80809
3.15101 3.43608 3.70342 3.80344 3.70342 3.43608 3.15101 2.80809
2.42151 2.00123 1.56046 1.11836 0.63069
            0.8097  1.37747 1.90794 2.44749 2.96826 3.4458
3.86291 4.20182 4.48984 4.60037 4.48984 4.20182 3.86291 3.4458
2.96826 2.44749 1.90794 1.37747 0.8097
            0.98977 1.62137 2.22727 2.85201 3.46778 4.04241
4.54369 4.94547 5.25533 5.37736 5.25533 4.94547 4.54369 4.04241
3.46778 2.85201 2.22727 1.62137 0.98977
            1.16468 1.84577 2.51149 3.20486 3.89875 4.5599
5.14842 5.62096 5.95973 6.09418 5.95973 5.62096 5.14842 4.5599
3.89875 3.20486 2.51149 1.84577 1.16468
            1.321   2.03952 2.75088 3.49471 4.24606 4.97244
5.63246 6.17513 6.5509  6.69854 6.5509  6.17513 5.63246 4.97244
4.24606 3.49471 2.75088 2.03952 1.321
            1.44212 2.18674 2.92777 3.70342 4.48984 5.25533
5.95973 6.5509  6.96378 7.12635 6.96378 6.5509  5.95973 5.25533
4.48984 3.70342 2.92777 2.18674 1.44212
```

```
            1.50714 2.26379 3.01656 3.80342 4.60035 5.37727
6.0935  6.698   7.29122 7.29122 7.29122 6.698   6.0935  5.37727
4.60035 3.80342 3.01656 2.26379 1.50714
            1.44212 2.18674 2.92777 3.70342 4.48984 5.25533
5.95973 6.5509  6.96378 7.12635 6.96378 6.5509  5.95973 5.25533
4.48984 3.70342 2.92777 2.18674 1.44212
            1.321   2.03952 2.75088 3.49471 4.24606 4.97244
5.63246 6.17513 6.5509  6.69854 6.5509  6.17513 5.63246 4.97244
4.24606 3.49471 2.75088 2.03952 1.321
            1.16468 1.84577 2.51149 3.20486 3.89875 4.5599
5.14842 5.62096 5.95973 6.09418 5.95973 5.62096 5.14842 4.5599
3.89875 3.20486 2.51149 1.84577 1.16468
            0.98977 1.62137 2.22727 2.85201 3.46778 4.04241
4.54369 4.94547 5.25533 5.37736 5.25533 4.94547 4.54369 4.04241
3.46778 2.85201 2.22727 1.62137 0.98977
            0.8097  1.37747 1.90794 2.44749 2.96826 3.4458
3.86291 4.20182 4.48984 4.60037 4.48984 4.20182 3.86291 3.4458
2.96826 2.44749 1.90794 1.37747 0.8097
            0.63069 1.11836 1.56046 2.00123 2.42151 2.80809
3.15101 3.43608 3.70342 3.80344 3.70342 3.43608 3.15101 2.80809
2.42151 2.00123 1.56046 1.11836 0.63069
            0.45672 0.84645 1.19176 1.53192 1.85864 2.16589
2.44565 2.68316 2.92777 3.01636 2.92777 2.68316 2.44565 2.16589
1.85864 1.53192 1.19176 0.84645 0.45672
            0.28697 0.5657  0.81658 1.06657 1.31285 1.55174
1.77605 1.97111 2.18674 2.26363 2.18674 1.97111 1.77605 1.55174
1.31285 1.06657 0.81658 0.5657  0.28697
            0.12453 0.28697 0.45672 0.63069 0.8097  0.98977
1.16468 1.321   1.44212 1.50714 1.44212 1.321   1.16468 0.98977
0.8097  0.63069 0.45672 0.28697 0.12453];
 %Z=5mm
M(:,:,4)=  [0.13998 0.30165 0.46943 0.64224 0.81867 0.99408
1.16249 1.31129 1.42516 1.48483 1.42516 1.31129 1.16249 0.99408
0.81867 0.64224 0.46943 0.30165 0.13998
            0.30165 0.57113 0.81639 1.06098 1.30172 1.53387
1.74975 1.93551 2.0749  2.1462  2.0749  1.93551 1.74975 1.53387
1.30172 1.06098 0.81639 0.57113 0.30165
            0.46943 0.84282 1.17587 1.50517 1.82227 2.11975
2.38882 2.61518 2.78135 2.8643  2.78135 2.61518 2.38882 2.11975
1.82227 1.50517 1.17587 0.84282 0.46943
            0.64224 1.10784 1.53085 1.95338 2.35766 2.73041
3.05984 3.33162 3.52676 3.6202  3.52676 3.33162 3.05984 2.73041
2.35766 1.95338 1.53085 1.10784 0.64224
            0.81867 1.36085 1.86694 2.38108 2.87829 3.33533
3.73431 4.05764 4.28427 4.388   4.28427 4.05764 3.73431 3.33533
2.87829 2.38108 1.86694 1.36085 0.81867
            0.99408 1.59833 2.17586 2.77023 3.35518 3.90082
4.3774  4.75861 5.02058 5.1347  5.02058 4.75861 4.3774  3.90082
3.35518 2.77023 2.17586 1.59833 0.99408
            1.16249 1.81475 2.44987 3.10932 3.76751 4.39262
4.9473  5.39272 5.69453 5.8194  5.69453 5.39272 4.9473  4.39262
3.76751 3.10932 2.44987 1.81475 1.16249
```

```
            1.31129 1.99984 2.67846 3.38608 4.09846 4.78385
5.40332 5.90993 6.25433 6.3904  6.25433 5.90993 5.40332 4.78385
4.09846 3.38608 2.67846 1.99984 1.31129
            1.42516 2.13863 2.84507 3.58308 4.32835 5.05032
5.71023 6.25923 6.63921 6.7852  6.63921 6.25923 5.71023 5.05032
4.32835 3.58308 2.84507 2.13863 1.42516
            1.48483 2.20951 2.92704 3.67551 4.43106 5.16336
5.83429 6.39462 6.78469 6.9311  6.78469 6.39462 5.83429 5.16336
4.43106 3.67551 2.92704 2.20951 1.48483
            1.42516 2.13863 2.84507 3.58308 4.32835 5.05032
5.71023 6.25923 6.63921 6.7852  6.63921 6.25923 5.71023 5.05032
4.32835 3.58308 2.84507 2.13863 1.42516
            1.31129 1.99984 2.67846 3.38608 4.09846 4.78385
5.40332 5.90993 6.25433 6.3904  6.25433 5.90993 5.40332 4.78385
4.09846 3.38608 2.67846 1.99984 1.31129
            1.16249 1.81475 2.44987 3.10932 3.76751 4.39262
4.9473  5.39272 5.69453 5.8194  5.69453 5.39272 4.9473  4.39262
3.76751 3.10932 2.44987 1.81475 1.16249
            0.99408 1.59833 2.17586 2.77023 3.35518 3.90082
4.3774  4.75861 5.02058 5.1347  5.02058 4.75861 4.3774  3.90082
3.35518 2.77023 2.17586 1.59833 0.99408
            0.81867 1.36085 1.86694 2.38108 2.87829 3.33533
3.73431 4.05764 4.28427 4.388   4.28427 4.05764 3.73431 3.33533
2.87829 2.38108 1.86694 1.36085 0.81867
            0.64224 1.10784 1.53085 1.95338 2.35766 2.73041
3.05984 3.33162 3.52676 3.6202  3.52676 3.33162 3.05984 2.73041
2.35766 1.95338 1.53085 1.10784 0.64224
            0.46943 0.84282 1.17587 1.50517 1.82227 2.11975
2.38882 2.61518 2.78135 2.8643  2.78135 2.61518 2.38882 2.11975
1.82227 1.50517 1.17587 0.84282 0.46943
            0.30165 0.57113 0.81639 1.06098 1.30172 1.53387
1.74975 1.93551 2.0749  2.1462  2.0749  1.93551 1.74975 1.53387
1.30172 1.06098 0.81639 0.57113 0.30165
            0.13998 0.30165 0.46943 0.64224 0.81867 0.99408
1.16249 1.31129 1.42516 1.48483 1.42516 1.31129 1.16249 0.99408
0.81867 0.64224 0.46943 0.30165 0.13998];

 %Z=10mm
M(:,:,5)=[0.20963   0.36042 0.51732 0.6768  0.83541 0.98795
1.1277  1.24557 1.33108 1.37389 1.33108 1.24557 1.1277  0.98795
0.83541 0.6768  0.51732 0.36042 0.20963
            0.36042 0.58426 0.79875 1.01396 1.22396 1.42211
1.60143 1.74872 1.85368 1.90459 1.85368 1.74872 1.60143 1.42211
1.22396 1.01396 0.79875 0.58426 0.36042
            0.51732 0.81488 1.09228 1.36971 1.63763 1.8868
2.10816 2.28757 2.41315 2.47229 2.41315 2.28757 2.10816 1.8868
1.63763 1.36971 1.09228 0.81488 0.51732
            0.6768  1.0435  1.38514 1.72835 2.05887 2.36398
2.63212 2.84649 2.99411 3.06086 2.99411 2.84649 2.63212 2.36398
2.05887 1.72835 1.38514 1.0435  0.6768
```

```
            0.83541 1.2629  1.66573 2.07331 2.46794 2.83229
3.15018 3.40274 3.57371 3.64782 3.57371 3.40274 3.15018 2.83229
2.46794 2.07331 1.66573 1.2629  0.83541
            0.98795 1.46668 1.92371 2.38968 2.84495 3.26765
3.63727 3.92947 4.12485 4.20592 4.12485 3.92947 3.63727 3.26765
2.84495 2.38968 1.92371 1.46668 0.98795
            1.1277  1.64725 2.14903 2.66398 3.17106 3.64623
4.06449 4.3956  4.61564 4.70313 4.61564 4.3956  4.06449 3.64623
3.17106 2.66398 2.14903 1.64725 1.1277
            1.24557 1.79563 2.33124 2.88313 3.4295  3.94548
4.40293 4.76819 5.01021 5.10253 5.01021 4.76819 4.40293 3.94548
3.4295  2.88313 2.33124 1.79563 1.24557
            1.33108 1.90057 2.45791 3.03257 3.60344 4.14464
4.62691 5.01399 5.27085 5.36567 5.27085 5.01399 4.62691 4.14464
3.60344 3.03257 2.45791 1.90057 1.33108
            1.37389 1.95108 2.51652 3.09936 3.6777  4.22616
4.71483 5.10697 5.36613 5.45827 5.36613 5.10697 4.71483 4.22616
3.6777  3.09936 2.51652 1.95108 1.37389
            1.33108 1.90057 2.45791 3.03257 3.60344 4.14464
4.62691 5.01399 5.27085 5.36567 5.27085 5.01399 4.62691 4.14464
3.60344 3.03257 2.45791 1.90057 1.33108
            1.24557 1.79563 2.33124 2.88313 3.4295  3.94548
4.40293 4.76819 5.01021 5.10253 5.01021 4.76819 4.40293 3.94548
3.4295  2.88313 2.33124 1.79563 1.24557
            1.1277  1.64725 2.14903 2.66398 3.17106 3.64623
4.06449 4.3956  4.61564 4.70313 4.61564 4.3956  4.06449 3.64623
3.17106 2.66398 2.14903 1.64725 1.1277
            0.98795 1.46668 1.92371 2.38968 2.84495 3.26765
3.63727 3.92947 4.12485 4.20592 4.12485 3.92947 3.63727 3.26765
2.84495 2.38968 1.92371 1.46668 0.98795
            0.83541 1.2629  1.66573 2.07331 2.46794 2.83229
3.15018 3.40274 3.57371 3.64782 3.57371 3.40274 3.15018 2.83229
2.46794 2.07331 1.66573 1.2629  0.83541
            0.6768  1.0435  1.38514 1.72835 2.05887 2.36398
2.63212 2.84649 2.99411 3.06086 2.99411 2.84649 2.63212 2.36398
2.05887 1.72835 1.38514 1.0435  0.6768
            0.51732 0.81488 1.09228 1.36971 1.63763 1.8868
2.10816 2.28757 2.41315 2.47229 2.41315 2.28757 2.10816 1.8868
1.63763 1.36971 1.09228 0.81488 0.51732
            0.36042 0.58426 0.79875 1.01396 1.22396 1.42211
1.60143 1.74872 1.85368 1.90459 1.85368 1.74872 1.60143 1.42211
1.22396 1.01396 0.79875 0.58426 0.36042
            0.20963 0.36042 0.51732 0.6768  0.83541 0.98795
1.1277  1.24557 1.33108 1.37389 1.33108 1.24557 1.1277  0.98795
0.83541 0.6768  0.51732 0.36042 0.20963];

 %Z=15mm
M(:,:,6)=[0.26264   0.39828 0.53899 0.68108 0.81999 0.95057
1.06689 1.16195 1.22882 1.26133 1.22882 1.16195 1.06689 0.95057
0.81999 0.68108 0.53899 0.39828 0.26264
```

```
            0.39828 0.58733 0.77153 0.95619 1.13493 1.30101
1.44749 1.56547 1.64639 1.68495 1.64639 1.56547 1.44749 1.30101
1.13493 0.95619 0.77153 0.58733 0.39828
            0.53899 0.78215 1.01287 1.2436  1.46586 1.67091
1.84992 1.99254 2.08918 2.13369 2.08918 1.99254 1.84992 1.67091
1.46586 1.2436  1.01287 0.78215 0.53899
            0.68108 0.9759  1.25372 1.53212 1.80018 2.04657
2.26018 2.4292  2.54215 2.59235 2.54215 2.4292  2.26018 2.04657
1.80018 1.53212 1.25372 0.9759  0.68108
            0.81999 1.16164 1.48464 1.80968 2.12343 2.41188
2.66131 2.85753 2.98731 3.04261 2.98731 2.85753 2.66131 2.41188
2.12343 1.80968 1.48464 1.16164 0.81999
            0.95057 1.33259 1.69619 2.06386 2.42023 2.749
3.03369 3.25683 3.40315 3.46328 3.40315 3.25683 3.03369 2.749
2.42023 2.06386 1.69619 1.33259 0.95057
            1.06689 1.48167 1.87928 2.28297 2.67593 3.04032
3.35648 3.60469 3.76652 3.83005 3.76652 3.60469 3.35648 3.04032
2.67593 2.28297 1.87928 1.48167 1.06689
            1.16195 1.60109 2.02446 2.45544 2.87665 3.26835
3.60981 3.8779  4.05217 4.11811 4.05217 3.8779  3.60981 3.26835
2.87665 2.45544 2.02446 1.60109 1.16195
            1.22882 1.68304 2.1229  2.57101 3.00941 3.41829
3.77503 4.05572 4.23723 4.30347 4.23723 4.05572 3.77503 3.41829
3.00941 2.57101 2.1229  1.68304 1.22882
            1.26133 1.72087 2.1668  2.62086 3.06495 3.47844
3.83909 4.12183 4.30381 4.36758 4.30381 4.12183 3.83909 3.47844
3.06495 2.62086 2.1668  1.72087 1.26133
            1.22882 1.68304 2.1229  2.57101 3.00941 3.41829
3.77503 4.05572 4.23723 4.30347 4.23723 4.05572 3.77503 3.41829
3.00941 2.57101 2.1229  1.68304 1.22882
            1.16195 1.60109 2.02446 2.45544 2.87665 3.26835
3.60981 3.8779  4.05217 4.11811 4.05217 3.8779  3.60981 3.26835
2.87665 2.45544 2.02446 1.60109 1.16195
            1.06689 1.48167 1.87928 2.28297 2.67593 3.04032
3.35648 3.60469 3.76652 3.83005 3.76652 3.60469 3.35648 3.04032
2.67593 2.28297 1.87928 1.48167 1.06689
            0.95057 1.33259 1.69619 2.06386 2.42023 2.749
3.03369 3.25683 3.40315 3.46328 3.40315 3.25683 3.03369 2.749
2.42023 2.06386 1.69619 1.33259 0.95057
            0.81999 1.16164 1.48464 1.80968 2.12343 2.41188
2.66131 2.85753 2.98731 3.04261 2.98731 2.85753 2.66131 2.41188
2.12343 1.80968 1.48464 1.16164 0.81999
            0.68108 0.9759  1.25372 1.53212 1.80018 2.04657
2.26018 2.4292  2.54215 2.59235 2.54215 2.4292  2.26018 2.04657
1.80018 1.53212 1.25372 0.9759  0.68108
            0.53899 0.78215 1.01287 1.2436  1.46586 1.67091
1.84992 1.99254 2.08918 2.13369 2.08918 1.99254 1.84992 1.67091
1.46586 1.2436  1.01287 0.78215 0.53899
            0.39828 0.58733 0.77153 0.95619 1.13493 1.30101
1.44749 1.56547 1.64639 1.68495 1.64639 1.56547 1.44749 1.30101
1.13493 0.95619 0.77153 0.58733 0.39828
```

```
            0.26264 0.39828 0.53899 0.68108 0.81999 0.95057
1.06689 1.16195 1.22882 1.26133 1.22882 1.16195 1.06689 0.95057
0.81999 0.68108 0.53899 0.39828 0.26264];
 %Z=20mm
 M(:,:,7)=[0.29854  0.41837 0.54202 0.66578 0.78534 0.89605
0.99281 1.07031 1.12373 1.14911 1.12373 1.07031 0.99281 0.89605
0.78534 0.66578 0.54202 0.41837 0.29854
            0.41837 0.57774 0.73457 0.89075 1.04097 1.179
1.29881 1.39391 1.45866 1.48842 1.45866 1.39391 1.29881 1.179
1.04097 0.89075 0.73457 0.57774 0.41837
            0.54202 0.74176 0.93334 1.12423 1.30716 1.47462 1.619
1.73282 1.80928 1.84341 1.80928 1.73282 1.619   1.47462 1.30716
1.12423 0.93334 0.74176 0.54202
            0.66578 0.90444 1.1312  1.3573  1.57391 1.77178
1.94188 2.07503 2.16377 2.20194 2.16377 2.07503 1.94188 1.77178
1.57391 1.3573  1.1312  0.90444 0.66578
            0.78534 1.05986 1.32025 1.58043 1.83013 2.05836
2.25407 2.40684 2.5073  2.54907 2.5073  2.40684 2.25407 2.05836
1.83013 1.58043 1.32025 1.05986 0.78534
            0.89605 1.20191 1.49252 1.78398 2.06428 2.32079
2.5408  2.71215 2.82404 2.86876 2.82404 2.71215 2.5408  2.32079
2.06428 1.78398 1.49252 1.20191 0.89605
            0.99281 1.32436 1.64044 1.9582  2.26459 2.5458
2.78719 2.97485 3.09642 3.14362 3.09642 2.97485 2.78719 2.5458
2.26459 1.9582  1.64044 1.32436 0.99281
            1.07031 1.42074 1.75634 2.09421 2.42055 2.72061
2.97844 3.17887 3.3079  3.356   3.3079  3.17887 2.97844 2.72061
2.42055 2.09421 1.75634 1.42074 1.07031
            1.12373 1.48577 1.83375 2.18399 2.52284 2.8344
3.10227 3.30998 3.44303 3.49077 3.44303 3.30998 3.10227 2.8344
2.52284 2.18399 1.83375 1.48577 1.12373
            1.14911 1.51494 1.86735 2.22211 2.56465 2.8794
3.14948 3.3584  3.49122 3.53714 3.49122 3.3584  3.14948 2.8794
2.56465 2.22211 1.86735 1.51494 1.14911
            1.12373 1.48577 1.83375 2.18399 2.52284 2.8344
3.10227 3.30998 3.44303 3.49077 3.44303 3.30998 3.10227 2.8344
2.52284 2.18399 1.83375 1.48577 1.12373
            1.07031 1.42074 1.75634 2.09421 2.42055 2.72061
2.97844 3.17887 3.3079  3.356   3.3079  3.17887 2.97844 2.72061
2.42055 2.09421 1.75634 1.42074 1.07031
            0.99281 1.32436 1.64044 1.9582  2.26459 2.5458
2.78719 2.97485 3.09642 3.14362 3.09642 2.97485 2.78719 2.5458
2.26459 1.9582  1.64044 1.32436 0.99281
            0.89605 1.20191 1.49252 1.78398 2.06428 2.32079
2.5408  2.71215 2.82404 2.86876 2.82404 2.71215 2.5408  2.32079
2.06428 1.78398 1.49252 1.20191 0.89605
            0.78534 1.05986 1.32025 1.58043 1.83013 2.05836
2.25407 2.40684 2.5073  2.54907 2.5073  2.40684 2.25407 2.05836
1.83013 1.58043 1.32025 1.05986 0.78534
            0.66578 0.90444 1.1312  1.3573  1.57391 1.77178
1.94188 2.07503 2.16377 2.20194 2.16377 2.07503 1.94188 1.77178
1.57391 1.3573  1.1312  0.90444 0.66578
```

```
            0.54202 0.74176 0.93334 1.12423 1.30716 1.47462 1.619
1.73282 1.80928 1.84341 1.80928 1.73282 1.619   1.47462 1.30716
1.12423 0.93334 0.74176 0.54202
            0.41837 0.57774 0.73457 0.89075 1.04097 1.179
1.29881 1.39391 1.45866 1.48842 1.45866 1.39391 1.29881 1.179
1.04097 0.89075 0.73457 0.57774 0.41837
            0.29854 0.41837 0.54202 0.66578 0.78534 0.89605
0.99281 1.07031 1.12373 1.14911 1.12373 1.07031 0.99281 0.89605
0.78534 0.66578 0.54202 0.41837 0.29854];
  %Z=25mm
 M(:,:,8)=[0.31991  0.42456 0.53174 0.63822 0.74008 0.83337
0.91386 0.97739 1.02058 1.04064 1.02058 0.97739 0.91386 0.83337
0.74008 0.63822 0.53174 0.42456 0.31991
            0.42456 0.55856 0.69122 0.82262 0.94808 1.06239
1.16045 1.23752 1.28927 1.31269 1.28927 1.23752 1.16045 1.06239
0.94808 0.82262 0.69122 0.55856 0.42456
            0.53174 0.6962  0.85524 1.01284 1.16297 1.29938
1.4161  1.5072  1.56782 1.59426 1.56782 1.5072  1.4161  1.29938
1.16297 1.01284 0.85524 0.6962  0.53174
            0.63822 0.83221 1.01765 1.20167 1.3767  1.53571
1.67125 1.77664 1.84618 1.87549 1.84618 1.77664 1.67125 1.53571
1.3767  1.20167 1.01765 0.83221 0.63822
            0.74008 0.96145 1.17206 1.38136 1.58055 1.76152
1.91572 2.03506 2.11314 2.14506 2.11314 2.03506 1.91572 1.76152
1.58055 1.38136 1.17206 0.96145 0.74008
            0.83337 1.07877 1.31224 1.54421 1.76572 1.9669
2.13819 2.27063 2.35646 2.39041 2.35646 2.27063 2.13819 1.9669
1.76572 1.54421 1.31224 1.07877 0.83337
            0.91386 1.17905 1.43156 1.68287 1.92318 2.14164
2.32776 2.47127 2.56354 2.59896 2.56354 2.47127 2.32776 2.14164
1.92318 1.68287 1.43156 1.17905 0.91386
            0.97739 1.25731 1.52431 1.79027 2.04483 2.27653
2.47372 2.62545 2.72259 2.75864 2.72259 2.62545 2.47372 2.27653
2.04483 1.79027 1.52431 1.25731 0.97739
            1.02058 1.30943 1.58536 1.86064 2.12403 2.36362
2.5675  2.72399 2.82347 2.85897 2.82347 2.72399 2.5675  2.36362
2.12403 1.86064 1.58536 1.30943 1.02058
            1.04064 1.33219 1.61151 1.88982 2.15584 2.3977
2.60296 2.76012 2.85917 2.89324 2.85917 2.76012 2.60296 2.3977
2.15584 1.88982 1.61151 1.33219 1.04064
            1.02058 1.30943 1.58536 1.86064 2.12403 2.36362
2.5675  2.72399 2.82347 2.85897 2.82347 2.72399 2.5675  2.36362
2.12403 1.86064 1.58536 1.30943 1.02058
            0.97739 1.25731 1.52431 1.79027 2.04483 2.27653
2.47372 2.62545 2.72259 2.75864 2.72259 2.62545 2.47372 2.27653
2.04483 1.79027 1.52431 1.25731 0.97739
            0.91386 1.17905 1.43156 1.68287 1.92318 2.14164
2.32776 2.47127 2.56354 2.59896 2.56354 2.47127 2.32776 2.14164
1.92318 1.68287 1.43156 1.17905 0.91386
            0.83337 1.07877 1.31224 1.54421 1.76572 1.9669
2.13819 2.27063 2.35646 2.39041 2.35646 2.27063 2.13819 1.9669
1.76572 1.54421 1.31224 1.07877 0.83337
```

```
            0.74008 0.96145 1.17206 1.38136 1.58055 1.76152
1.91572 2.03506 2.11314 2.14506 2.11314 2.03506 1.91572 1.76152
1.58055 1.38136 1.17206 0.96145 0.74008
            0.63822 0.83221 1.01765 1.20167 1.3767  1.53571
1.67125 1.77664 1.84618 1.87549 1.84618 1.77664 1.67125 1.53571
1.3767  1.20167 1.01765 0.83221 0.63822
            0.53174 0.6962  0.85524 1.01284 1.16297 1.29938
1.4161  1.5072  1.56782 1.59426 1.56782 1.5072  1.4161  1.29938
1.16297 1.01284 0.85524 0.6962  0.53174
            0.42456 0.55856 0.69122 0.82262 0.94808 1.06239
1.16045 1.23752 1.28927 1.31269 1.28927 1.23752 1.16045 1.06239
0.94808 0.82262 0.69122 0.55856 0.42456
            0.31991 0.42456 0.53174 0.63822 0.74008 0.83337
0.91386 0.97739 1.02058 1.04064 1.02058 0.97739 0.91386 0.83337
0.74008 0.63822 0.53174 0.42456 0.31991];
 %z=30mm
 M(:,:,9)=[0.32961  0.42031 0.51259 0.60339 0.68969 0.76794
0.83487 0.88724 0.92232 0.93828 0.92232 0.88724 0.83487 0.76794
0.68969 0.60339 0.51259 0.42031 0.32961
           0.42031  0.53322 0.64511 0.75516 0.85944 0.9539
1.03433 1.09692 1.13855 1.15703 1.13855 1.09692 1.03433 0.9539
0.85944 0.75516 0.64511 0.53322 0.42031
            0.51259 0.64863 0.78061 0.91053 1.03358 1.14478
1.23907 1.31226 1.36046 1.38122 1.36046 1.31226 1.23907 1.14478
1.03358 0.91053 0.78061 0.64863 0.51259
            0.60339 0.76201 0.91403 1.06373 1.20553 1.33344
1.44178 1.52542 1.58022 1.60308 1.58022 1.52542 1.44178 1.33344
1.20553 1.06373 0.91403 0.76201 0.60339
            0.68969 0.86922 1.04018 1.2087  1.36842 1.51241
1.63428 1.72809 1.78895 1.81362 1.78895 1.72809 1.63428 1.51241
1.36842 1.2087  1.04018 0.86922 0.68969
            0.76794 0.96593 1.15396 1.33953 1.51534 1.67398
1.80814 1.91113 1.97754 2.00366 1.97754 1.91113 1.80814 1.67398
1.51534 1.33953 1.15396 0.96593 0.76794
            0.83487 1.04798 1.25026 1.45025 1.63961 1.81067
1.95515 2.06589 2.13681 2.16374 2.13681 2.06589 1.95515 1.81067
1.63961 1.45025 1.25026 1.04798 0.83487
            0.88724 1.1115  1.32476 1.5354  1.73517 1.91541
2.06772 2.18409 2.25801 2.28536 2.25801 2.18409 2.06772 1.91541
1.73517 1.5354  1.32476 1.1115  0.88724
            0.92232 1.15343 1.37355 1.5908  1.79688 1.98278
2.13958 2.25909 2.33456 2.36146 2.33456 2.25909 2.13958 1.98278
1.79688 1.5908  1.37355 1.15343 0.92232
            0.93828 1.17147 1.39389 1.61345 1.82143 2.0088
2.16661 2.28636 2.36149 2.38738 2.36149 2.28636 2.16661 2.0088
1.82143 1.61345 1.39389 1.17147 0.93828
            0.92232 1.15343 1.37355 1.5908  1.79688 1.98278
2.13958 2.25909 2.33456 2.36146 2.33456 2.25909 2.13958 1.98278
1.79688 1.5908  1.37355 1.15343 0.92232
            0.88724 1.1115  1.32476 1.5354  1.73517 1.91541
2.06772 2.18409 2.25801 2.28536 2.25801 2.18409 2.06772 1.91541
1.73517 1.5354  1.32476 1.1115  0.88724
```

```
            0.83487 1.04798 1.25026 1.45025 1.63961 1.81067
1.95515 2.06589 2.13681 2.16374 2.13681 2.06589 1.95515 1.81067
1.63961 1.45025 1.25026 1.04798 0.83487
            0.76794 0.96593 1.15396 1.33953 1.51534 1.67398
1.80814 1.91113 1.97754 2.00366 1.97754 1.91113 1.80814 1.67398
1.51534 1.33953 1.15396 0.96593 0.76794
            0.68969 0.86922 1.04018 1.2087  1.36842 1.51241
1.63428 1.72809 1.78895 1.81362 1.78895 1.72809 1.63428 1.51241
1.36842 1.2087  1.04018 0.86922 0.68969
            0.60339 0.76201 0.91403 1.06373 1.20553 1.33344
1.44178 1.52542 1.58022 1.60308 1.58022 1.52542 1.44178 1.33344
1.20553 1.06373 0.91403 0.76201 0.60339
            0.51259 0.64863 0.78061 0.91053 1.03358 1.14478
1.23907 1.31226 1.36046 1.38122 1.36046 1.31226 1.23907 1.14478
1.03358 0.91053 0.78061 0.64863 0.51259
            0.42031 0.53322 0.64511 0.75516 0.85944 0.9539
1.03433 1.09692 1.13855 1.15703 1.13855 1.09692 1.03433 0.9539
0.85944 0.75516 0.64511 0.53322 0.42031
            0.32961 0.42031 0.51259 0.60339 0.68969 0.76794
0.83487 0.88724 0.92232 0.93828 0.92232 0.88724 0.83487 0.76794
0.68969 0.60339 0.51259 0.42031 0.32961];
 %z=35mm
 M(:,:,10)=[0.3305  0.40864 0.48764 0.56488 0.63768 0.70327
0.75887 0.80202 0.8308  0.84356 0.8308  0.80202 0.75887 0.70327
0.63768 0.56488 0.48764 0.40864 0.3305
            0.40864 0.50375 0.59786 0.68986 0.77646 0.85439
0.92025 0.97123 1.00497 1.01999 1.00497 0.97123 0.92025 0.85439
0.77646 0.68986 0.59786 0.50375 0.40864
            0.48764 0.60048 0.71005 0.81717 0.91799 1.0085
1.08498 1.14397 1.18259 1.19936 1.18259 1.14397 1.08498 1.0085
0.91799 0.81717 0.71005 0.60048 0.48764
            0.56488 0.6949  0.81968 0.94186 1.05674 1.1598
1.24676 1.31358 1.35716 1.37523 1.35716 1.31358 1.24676 1.1598
1.05674 0.94186 0.81968 0.6949  0.56488
            0.63768 0.78376 0.92291 1.05914 1.18734 1.30239
1.39932 1.47363 1.5216  1.54088 1.5216  1.47363 1.39932 1.30239
1.18734 1.05914 0.92291 0.78376 0.63768
            0.70327 0.86344 1.01543 1.16437 1.30462 1.4304
1.53632 1.61721 1.66898 1.68921 1.66898 1.61721 1.53632 1.4304
1.30462 1.16437 1.01543 0.86344 0.70327
            0.75887 0.93067 1.09346 1.25301 1.40326 1.53817
1.65143 1.73763 1.79259 1.81345 1.79259 1.73763 1.65143 1.53817
1.40326 1.25301 1.09346 0.93067 0.75887
            0.80202 0.98242 1.15337 1.321   1.47885 1.62033
1.73901 1.8292  1.88614 1.90716 1.88614 1.8292  1.73901 1.62033
1.47885 1.321   1.15337 0.98242 0.80202
            0.8308  1.01632 1.19246 1.36507 1.52745 1.67278
1.79458 1.88695 1.94496 1.96565 1.94496 1.88695 1.79458 1.67278
1.52745 1.36507 1.19246 1.01632 0.8308
            0.84356 1.03067 1.20863 1.3828  1.5465  1.69293
1.8154  1.90789 1.96566 1.98549 1.96566 1.90789 1.8154  1.69293
1.5465  1.3828  1.20863 1.03067 0.84356
```

```
            0.8308  1.01632 1.19246 1.36507 1.52745 1.67278
1.79458 1.88695 1.94496 1.96565 1.94496 1.88695 1.79458 1.67278
1.52745 1.36507 1.19246 1.01632 0.8308
            0.80202 0.98242 1.15337 1.321   1.47885 1.62033
1.73901 1.8292  1.88614 1.90716 1.88614 1.8292  1.73901 1.62033
1.47885 1.321   1.15337 0.98242 0.80202
            0.75887 0.93067 1.09346 1.25301 1.40326 1.53817
1.65143 1.73763 1.79259 1.81345 1.79259 1.73763 1.65143 1.53817
1.40326 1.25301 1.09346 0.93067 0.75887
            0.70327 0.86344 1.01543 1.16437 1.30462 1.4304
1.53632 1.61721 1.66898 1.68921 1.66898 1.61721 1.53632 1.4304
1.30462 1.16437 1.01543 0.86344 0.70327
            0.63768 0.78376 0.92291 1.05914 1.18734 1.30239
1.39932 1.47363 1.5216  1.54088 1.5216  1.47363 1.39932 1.30239
1.18734 1.05914 0.92291 0.78376 0.63768
            0.56488 0.6949  0.81968 0.94186 1.05674 1.1598
1.24676 1.31358 1.35716 1.37523 1.35716 1.31358 1.24676 1.1598
1.05674 0.94186 0.81968 0.6949  0.56488
            0.48764 0.60048 0.71005 0.81717 0.91799 1.0085
1.08498 1.14397 1.18259 1.19936 1.18259 1.14397 1.08498 1.0085
0.91799 0.81717 0.71005 0.60048 0.48764
            0.40864 0.50375 0.59786 0.68986 0.77646 0.85439
0.92025 0.97123 1.00497 1.01999 1.00497 0.97123 0.92025 0.85439
0.77646 0.68986 0.59786 0.50375 0.40864
            0.3305  0.40864 0.48764 0.56488 0.63768 0.70327
0.75887 0.80202 0.8308  0.84356 0.8308  0.80202 0.75887 0.70327
0.63768 0.56488 0.48764 0.40864 0.3305];
  %z=40mm
 M(:,:,11)=[0.32494 0.39212 0.45957 0.52505 0.58639 0.64128
0.68755 0.72327 0.74682 0.7571  0.74682 0.72327 0.68755 0.64128
0.58639 0.52505 0.45957 0.39212 0.32494
            0.39212 0.47235 0.5515  0.62839 0.70037 0.76476
0.81891 0.86058 0.8879  0.89956 0.8879  0.86058 0.81891 0.76476
0.70037 0.62839 0.5515  0.47235 0.39212
            0.45957 0.55344 0.6445  0.73306 0.8159  0.88984
0.95197 0.99968 1.03071 1.04377 1.03071 0.99968 0.95197 0.88984
0.8159  0.73306 0.6445  0.55344 0.45957
            0.52505 0.6322  0.73491 0.83484 0.92833 1.01174
1.08178 1.13533 1.1699  1.18417 1.1699  1.13533 1.08178 1.01174
0.92833 0.83484 0.73491 0.6322  0.52505
            0.58639 0.7058  0.81952 0.93011 1.03361 1.12591
1.20328 1.26233 1.30034 1.31536 1.30034 1.26233 1.20328 1.12591
1.03361 0.93011 0.81952 0.7058  0.58639
            0.64128 0.77151 0.89504 1.01514 1.12752 1.22776
1.31173 1.37564 1.41648 1.43212 1.41648 1.37564 1.31173 1.22776
1.12752 1.01514 0.89504 0.77151 0.64128
            0.68755 0.8267  0.9583  1.08639 1.2062  1.31307
1.40249 1.47029 1.51321 1.52928 1.51321 1.47029 1.40249 1.31307
1.2062  1.08639 0.9583  0.8267  0.68755
            0.72327 0.86891 1.00669 1.14076 1.26613 1.37797
1.47122 1.54174 1.58621 1.60239 1.58621 1.54174 1.47122 1.37797
1.26613 1.14076 1.00669 0.86891 0.72327
```

```
            0.74682 0.89646 1.03802 1.17576 1.3046  1.41926
1.51471 1.58673 1.63187 1.64779 1.63187 1.58673 1.51471 1.41926
1.3046  1.17576 1.03802 0.89646 0.74682
            0.7571  0.9079  1.05084 1.18988 1.31963 1.43491
1.53083 1.60295 1.64785 1.66325 1.64785 1.60295 1.53083 1.43491
1.31963 1.18988 1.05084 0.9079  0.7571
            0.74682 0.89646 1.03802 1.17576 1.3046  1.41926
1.51471 1.58673 1.63187 1.64779 1.63187 1.58673 1.51471 1.41926
1.3046  1.17576 1.03802 0.89646 0.74682
            0.72327 0.86891 1.00669 1.14076 1.26613 1.37797
1.47122 1.54174 1.58621 1.60239 1.58621 1.54174 1.47122 1.37797
1.26613 1.14076 1.00669 0.86891 0.72327
            0.68755 0.8267  0.9583  1.08639 1.2062  1.31307
1.40249 1.47029 1.51321 1.52928 1.51321 1.47029 1.40249 1.31307
1.2062  1.08639 0.9583  0.8267  0.68755
            0.64128 0.77151 0.89504 1.01514 1.12752 1.22776
1.31173 1.37564 1.41648 1.43212 1.41648 1.37564 1.31173 1.22776
1.12752 1.01514 0.89504 0.77151 0.64128
            0.58639 0.7058  0.81952 0.93011 1.03361 1.12591
1.20328 1.26233 1.30034 1.31536 1.30034 1.26233 1.20328 1.12591
1.03361 0.93011 0.81952 0.7058  0.58639
            0.52505 0.6322  0.73491 0.83484 0.92833 1.01174
1.08178 1.13533 1.1699  1.18417 1.1699  1.13533 1.08178 1.01174
0.92833 0.83484 0.73491 0.6322  0.52505
            0.45957 0.55344 0.6445  0.73306 0.8159  0.88984
0.95197 0.99968 1.03071 1.04377 1.03071 0.99968 0.95197 0.88984
0.8159  0.73306 0.6445  0.55344 0.45957
            0.39212 0.47235 0.5515  0.62839 0.70037 0.76476
0.81891 0.86058 0.8879  0.89956 0.8879  0.86058 0.81891 0.76476
0.70037 0.62839 0.5515  0.47235 0.39212
            0.32494 0.39212 0.45957 0.52505 0.58639 0.64128
0.68755 0.72327 0.74682 0.7571  0.74682 0.72327 0.68755 0.64128
0.58639 0.52505 0.45957 0.39212 0.32494];

%% Setting up the mesh

x = -45:5:45;
y = -45:5:45;
z = [2 3 4 5 10 15 20 25 30 35 40];

[x1,y1,z1] = ndgrid(x,y,z);

%% Interpolation
xp = -45:0.5:45;
yp = -45:0.5:45;
zp = 2:0.5:40;
[x1p,y1p,z1p] = ndgrid(xp,yp,zp);

Mp = interp3(x1,y1,z1,M,x1p,y1p,z1p,'linear');

hxMp = fix(size(Mp,1)/2) + 1;
```

```
M1p = Mp(hxMp:end,hxMp:end,1:end);
M2p = Mp(hxMp:end,1:hxMp,1:end);
M3p = Mp(1:hxMp,1:hxMp,1:end);
M4p = Mp(1:hxMp,hxMp:end,1:end);

%% Plotting X, Y, Z vs M
L = [reshape(x1p(hxMp:end,hxMp:end,:),1,[]).',
reshape(y1p(hxMp:end,hxMp:end,:),1,[]).',
reshape(z1p(hxMp:end,hxMp:end,:),1,[]).'];
L_Z = reshape(Mp(hxMp:end,hxMp:end,:),1,[]);
colormap jet
[f,g,c] = plot3k(L, 'PlotType', 'scatter', 'ColorData', L_Z,
'MarkerType', '.', 'ColorRange', [min(L_Z) max(L_Z)]);
xlabel('X (mm)');
ylabel('Y (mm)');
zlabel('Z (mm)');
c.Label.String = ('Mutual Inductance (\muH)');
title('X, Y, Z, vs Mutual Inductance');
xlim([-40 40])
ylim([-40 40])
% plot3k function developed by Ken Garrard

%% Vector reshaping

x = -22.5:0.5:22.5;
y = -22.5:0.5:22.5;
[x1p,y1p,z1p] = ndgrid(x,y,zp);

Xp = reshape(x1p,[],1);
Yp = reshape(y1p,[],1);
Zp = reshape(z1p,[],1);

L_M1p = reshape(M1p,[],1);
L_M2p = reshape(M2p,[],1);
L_M3p = reshape(M3p,[],1);
L_M4p = reshape(M4p,[],1);

L_M123p = [L_M1p, L_M2p, L_M3p];
L_M124p = [L_M1p, L_M2p, L_M4p];
L_M341p = [L_M3p, L_M4p, L_M1p];

%% Backward conversion

n=50;

m1_domain = linspace(2,3.8,n);
m2_domain = linspace(2,3.8,n);
m3_domain = linspace(2,3.8,n);

[m1_Ip_box, m2_Ip_box, m3_Ip_box] = ndgrid(m1_domain, m2_domain,
m3_domain);
```

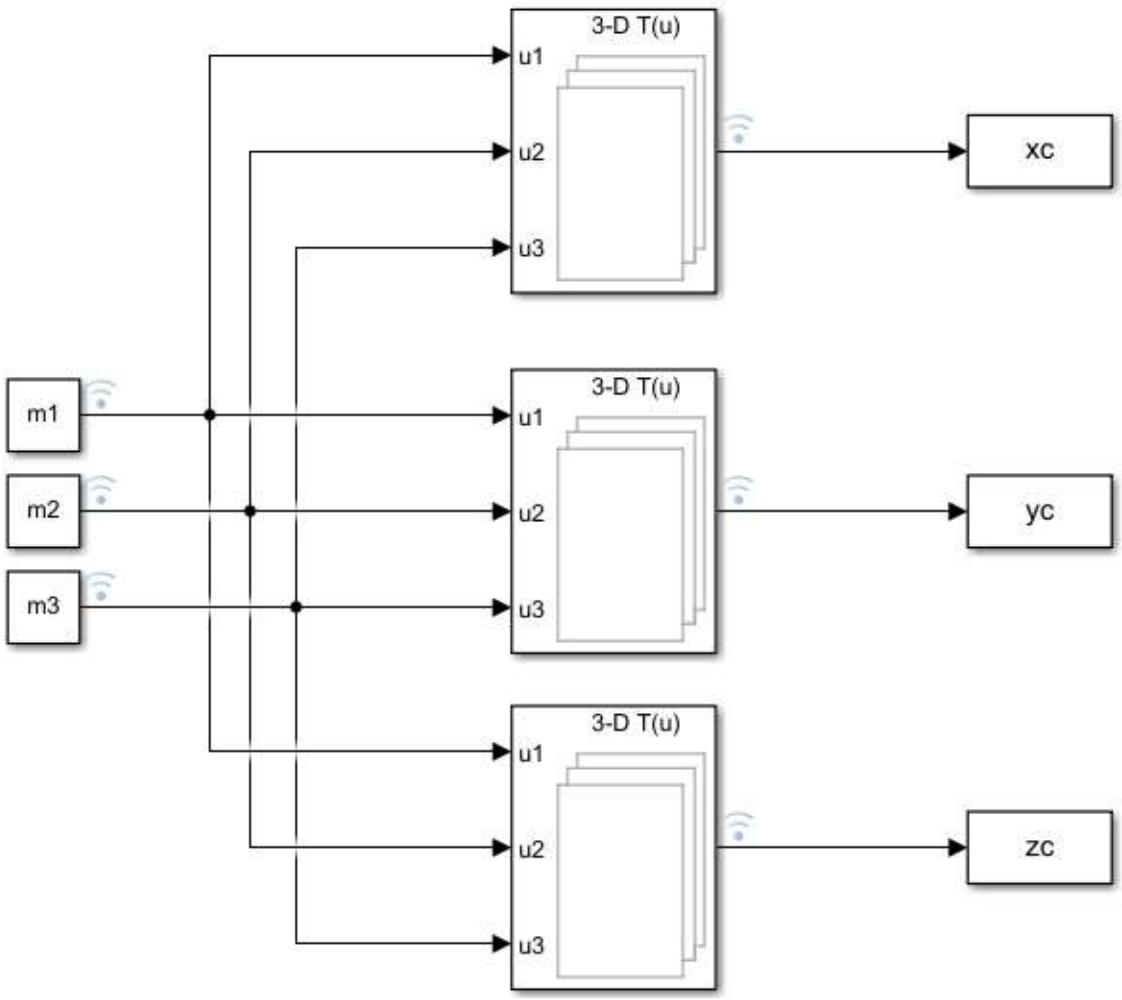
```
query_p = [reshape(m1_Ip_box,1,[]).' reshape(m2_Ip_box,1,[]).'
reshape(m3_Ip_box,1,[]).'];

xm = scatteredInterpolant(L_M124p, Xp,'linear','none');
xm1 = xm(m1_Ip_box, m2_Ip_box, m3_Ip_box);
xm_reshape = reshape(xm1,1,[]);

ym = scatteredInterpolant(L_M124p, Yp,'linear','none');
ym1 = ym(m1_Ip_box, m2_Ip_box, m3_Ip_box);
ym_reshape = reshape(ym1,1,[]);

zm = scatteredInterpolant(L_M124p, Zp,'linear','none');
zm1 = zm(m1_Ip_box, m2_Ip_box, m3_Ip_box);
zm_reshape = reshape(zm1,1,[]);


%% Simulink Lookup Table for one set of values

m1 = 2.698409451;
m2 = 3.861026917;
m3 = 3.744483007;
m4 = 2.0750;

sim('AJ_Lookup');
xv = xc.signals.values(1,1);
yv = yc.signals.values(1,1);
zv = zc.signals.values(1,1);

%% Plotting
colormap jet
[~,~,c1] = plot3k(L_M123p, 'PlotType', 'scatter', 'ColorData',
Xp, 'Marker', '.','ColorRange', [min(Xp) max(Xp)], 'MarkerSize',
2);
xlabel('M1 (\muH)');
ylabel('M2 (\muH)');
zlabel('M3 (\muH)');
c1.Label.String = 'X (mm)';
title('Mutual Inductance of coils 1,2,3 vs X (Interpolated)')

%% Plotting interpolation box
colormap jet
[~,~,c1] = plot3k(query_p, 'PlotType', 'scatter', 'ColorData',
xm_reshape, 'Marker', '.','ColorRange', [min(xm_reshape)
max(xm_reshape)], 'MarkerSize', 2);
xlabel('M1 (\muH)');
ylabel('M2 (\muH)');
zlabel('M3 (\muH)');
c1.Label.String = 'Z (mm)';
title('Mutual Inductance of coils 1,2,3 vs Z')

%% Lookup table for multiple sets
```

```
M1_check = [3.681263651 3.343870117 2.783447792 2.435601061
3.798450909 ...
    2.748434476 2.925483102 2.70630577 3.137204957 2.688048495
...
    2.878536585 2.557883307 2.340751157];
M2_check = [3.680371764 3.356608715 2.806937219 2.444346515
2.812195302 ...
    2.627524139 2.220557608 1.394191429 3.512412113 2.961174647
...
    3.654412354 3.135634915 3.188582667];
M3_check = [3.578156913 3.266395426 2.765106242 2.424117093
3.740096824 ...
    3.667862574 4.449976779 5.708685125 3.143477282 2.736463886
...
    2.939476691 2.911985991 3.002460792];

M1_check = M1_check';
M2_check = M2_check';
M3_check = M3_check';

for i=1:length(M1_check)
    m1 = M1_check(i); m2 = M2_check(i); m3 = M3_check(i);
    sim('AJ_Lookup');
    xv(i) = xc.signals.values(1,1);
    yv(i) = yc.signals.values(1,1);
    zv(i) = zc.signals.values(1,1);
end


x_r = [0 0 0 0 0 5 5 10 0 0 0 2 4];

y_r = [0 0 0 0 -5 0 5 10 2 2 4 4 6];

z_r = [2 5 10 15 5 5 5 5 5 10 5 8 5];

x_error = (abs(x_r') - abs(xv));
y_error = (abs(y_r') - abs(yv));
z_error = (abs(z_r') - abs(zv));

%% Plotting of Error vs Data points

num_p = 1:length(x_r);
plot(num_p,z_error,'.')
xlabel('Data points');
ylabel('Absolute error (mm)');
title('Absolute error in Z vs data points')

%% Lookup table for all combinations

x_check = -15:1:15;
y_check = -15:1:15;
z_check = 2:15;
```

```
[x_grid, y_grid, z_grid] = ndgrid(x_check,y_check,z_check);

x_index = zeros(length(x_check),1);
y_index = zeros(length(y_check),1);
z_index = zeros(length(z_check),1);
for i=1:length(x_check)
    x_index(i) = find(x == x_check(i));
    y_index(i) = find(y == y_check(i));
end

for i=1:length(z_check)
    z_index(i) = find(zp == z_check(i));
end

M1_check=zeros(length(x_index));
M2_check=zeros(length(y_index));
M3_check=zeros(length(z_index));
for k=1:length(z_index)
    for j=1:length(x_index)
        for i=1:length(x_index)
            M1_check(i,j,k) =
M1p(x_index(i),y_index(j),z_index(k));
            M2_check(i,j,k) =
M2p(x_index(i),y_index(j),z_index(k));
            M3_check(i,j,k) =
M3p(x_index(i),y_index(j),z_index(k));
        end
    end
end

xv=zeros(length(x_index));
yv=zeros(length(y_index));
zv=zeros(length(z_index));
for k=1:length(z_index)
    for j=1:length(y_index)
        for i=1:length(x_index)
            m1 = M1_check(i,j,k); m2 = M2_check(i,j,k); m3 =
M3_check(i,j,k);
            sim('AJ_Lookup');
            xv(i,j,k) = xc.signals.values(1,1);
            yv(i,j,k) = yc.signals.values(1,1);
            zv(i,j,k) = zc.signals.values(1,1);
        end
    end
end

%Checking for accuracy
acc_x = x_grid - xv;
acc_y = y_grid - yv;
acc_z = z_grid - zv(1:size(z_grid,1),1:size(z_grid,1),:);

%% Plotting residuals/error
```

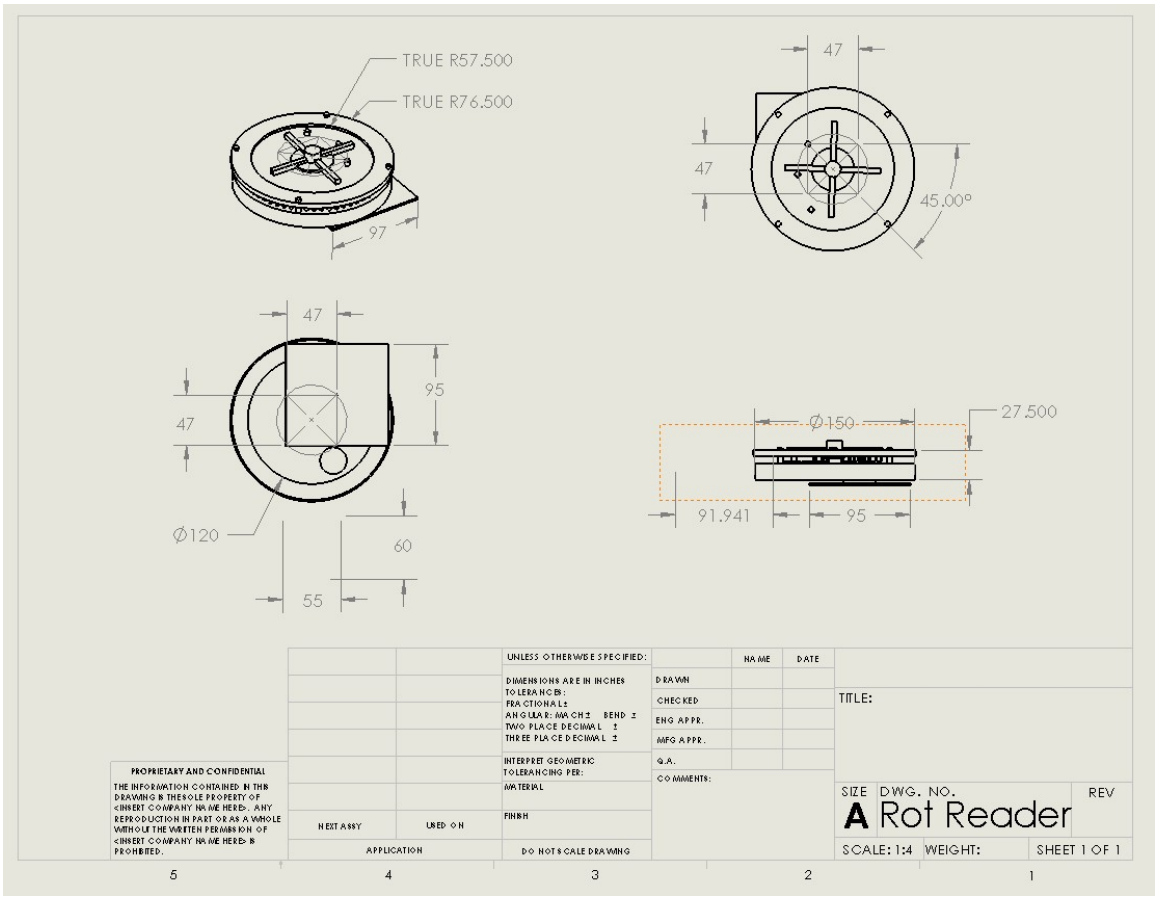
```
new_acc_x = acc_x;
new_acc_x(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) > 0.5)
= NaN;
new_acc_y = acc_y;
new_acc_y(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) > 0.5)
= NaN;
new_acc_z = acc_z;
new_acc_z(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) > 0.5)
= NaN;

new_x_grid = x_grid;
new_x_grid(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) >
0.5) = NaN;
new_y_grid = y_grid;
new_y_grid(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) >
0.5) = NaN;
new_z_grid = z_grid;
new_z_grid(abs(acc_x) > 0.5 | abs(acc_y) > 0.5 | abs(acc_z) >
0.5) = NaN;
```

7.1.3.1.1 Simulink Diagram

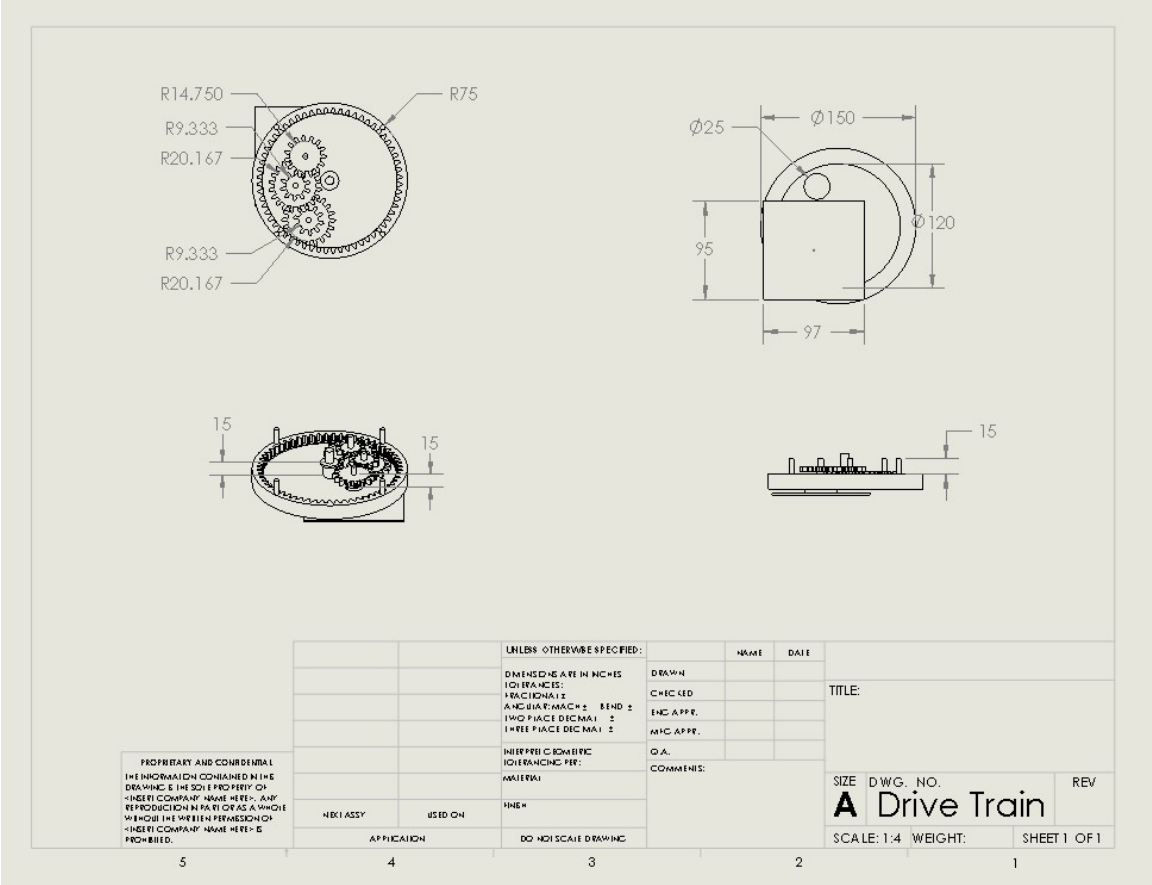


7.2.: Rotary Reader Engineering Drawings

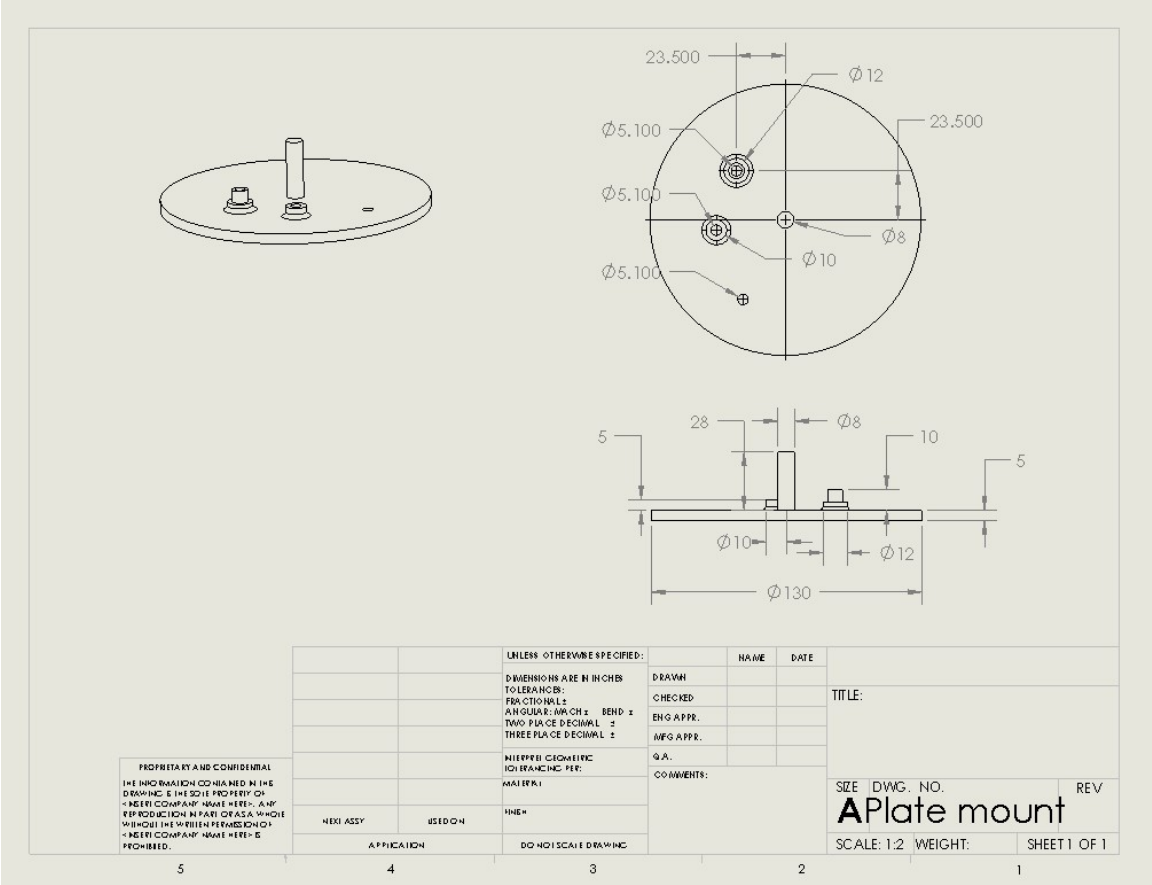
7.2.1.: Rotary Reader



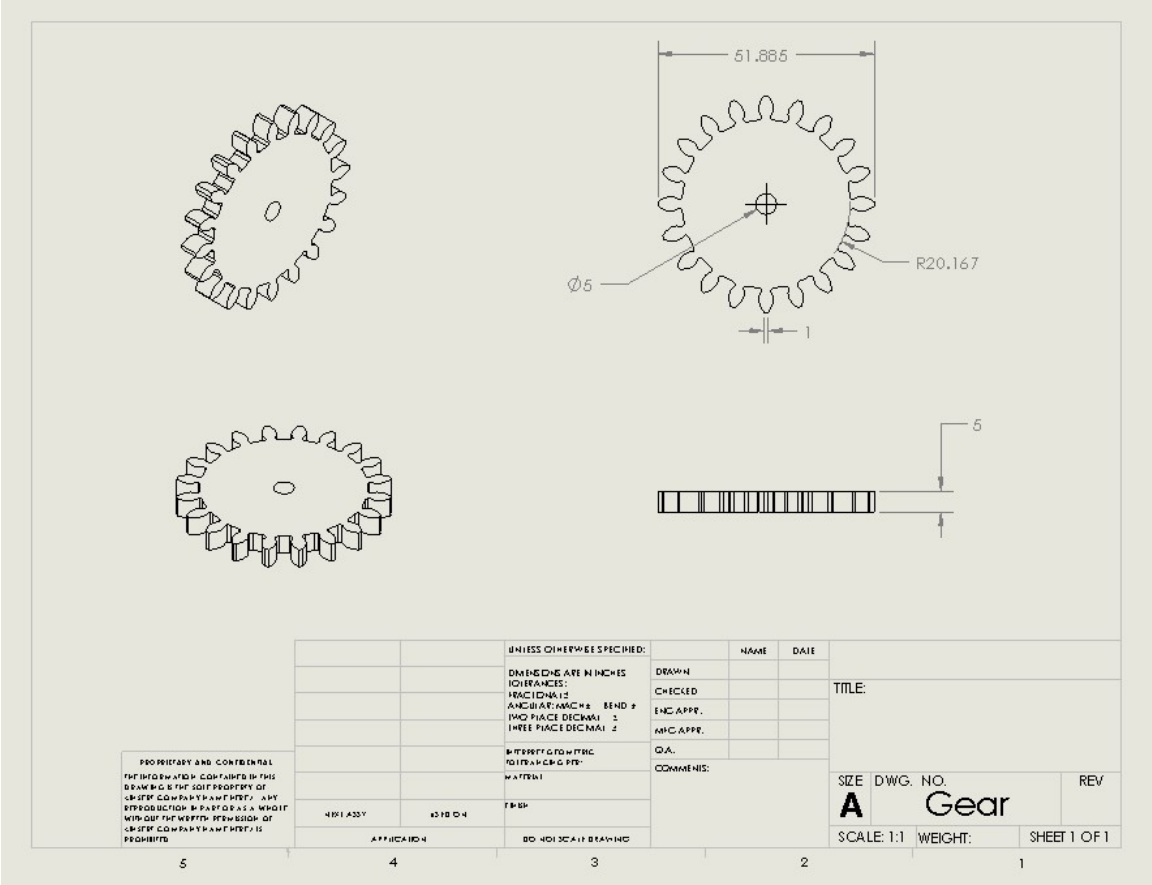
7.2.2.: Drive Train



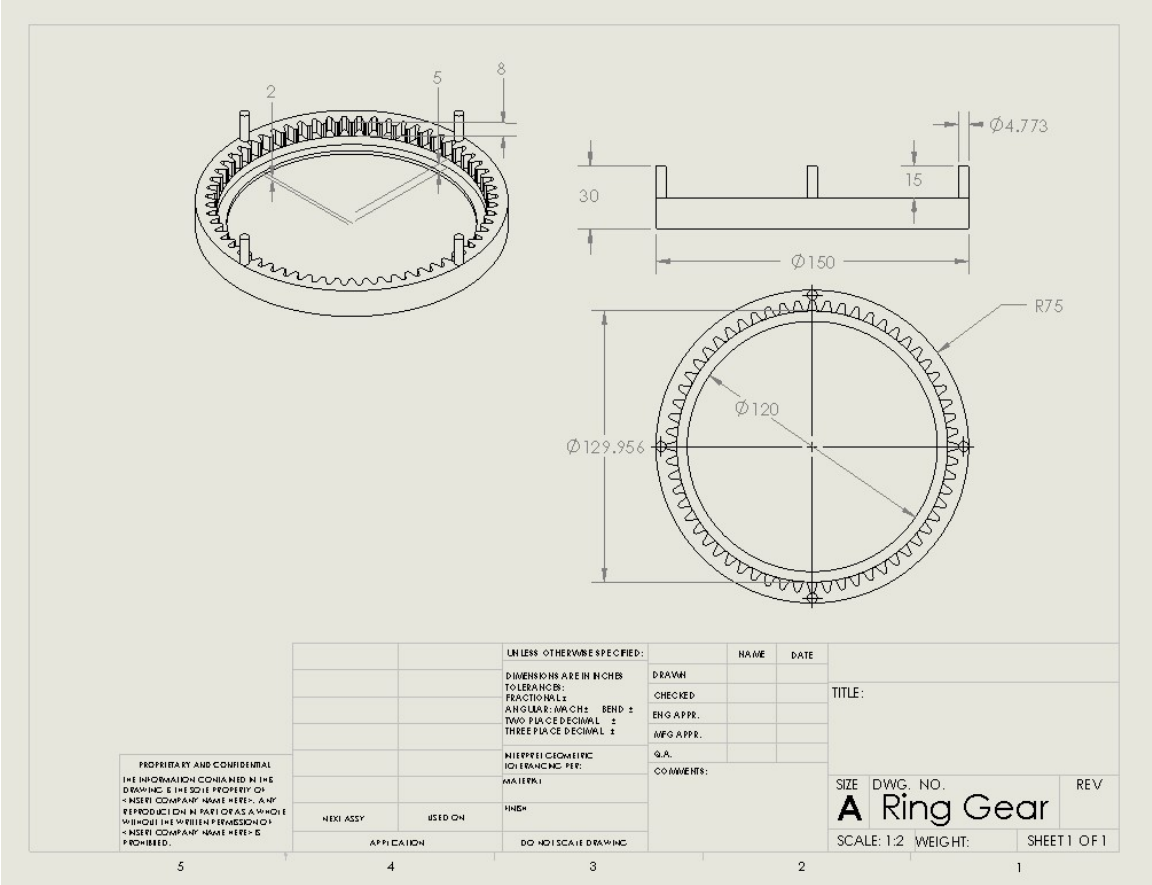
7.2.3.: Plate Mount



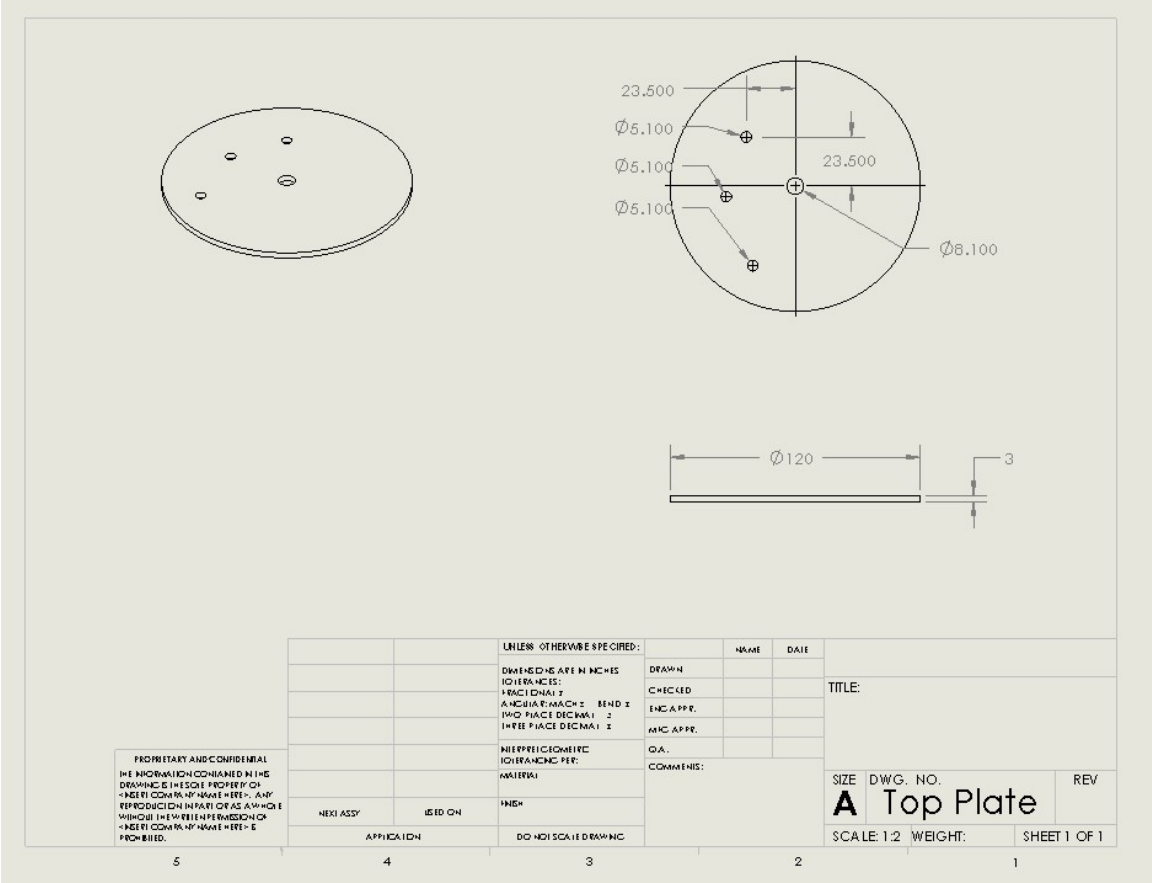
7.2.4.: Gear


51.885
Ø5
R20.167
1
5
UNLESS OTHERWISE SPECIFIED:
DIMENSIONS ARE IN INCHES
TOLERANCES:
FRACTIONAL ±
ANGULAR: MACH ± BEND ±
TWO PLACE DECIMAL ±
THREE PLACE DECIMAL ±
INTERPRET GEOMETRIC
TOLERANCING PER:
MATERIAL
FINISH
NEXT ASSY
USED ON
APPLICATION
DO NOT SCALE DRAWING
NAME
DATE
DRAWN
CHECKED
ENG APPR.
MFG APPR.
Q.A.
COMMENTS:
TITLE:
SIZE
A
DWG. NO.
Gear
REV
SCALE: 1:1
WEIGHT:
SHEET 1 OF 1
PROPRIETARY AND CONFIDENTIAL
THE INFORMATION CONTAINED IN THIS
DRAWING IS THE SOLE PROPERTY OF
<INSERT COMPANY NAME HERE>. ANY
REPRODUCTION IN PART OR AS A WHOLE
WITHOUT THE WRITTEN PERMISSION OF
<INSERT COMPANY NAME HERE> IS
PROHIBITED
5
4
3
2
1

7.2.5.: Ring Gear



7.2.6.: Top Plate



7.3 COMSOL Multiphysics Simulation

```
function out = model
%
% Multicoil_MATLAB_Copy2.m
%
% Model exported on Jun 20 2019, 16:01 by COMSOL 5.4.0.225.

import com.comsol.model.*
import com.comsol.model.util.*

model = ModelUtil.create('Model');

model.modelPath('/volume/NFS/pnf6/rSim/COMSOL/Wireless /Mutual
Inductance/Simulations');

model.label('Multicoil_MATLAB_Copy.mph');

model.param.set('b', '0.00065', 'Half of width of reader');
```

```
model.param.set('a', '0.0033', 'Half of Increasing length of reader per
rurn');
model.param.set('t1', '0.0001', 'Thickness of reader');
model.param.set('t2', '0.0005', 'Height of supportor in reader');
model.param.set('c', '0.008', 'Length of lumped port in reader');
model.param.set('r1', 'a*40', 'Radius of air');
model.param.set('r2', 'r1/10', 'Thickness of infinity area');
model.param.set('a1', '0.00865', 'Half of Increasing length of reader
in the first turn');
model.param.set('z_t', '0.01', 'The distance between the center of
transponder and orgin in Z');
model.param.set('x_t', '0', 'The distance between the center of
transponder and orgin in X');
model.param.set('y_t', '0', 'The distance between the center of
transponder and orgin in Y');
model.param.set('b_t', '0.00065', 'Half of width of transponder');
model.param.set('a_t', '0.0033', 'Half of Increasing length of
transponder per rurn');
model.param.set('t1_t', '0.0001', 'Thickness of transponder');
model.param.set('t2_t', '0.001', 'Height of supportor in transponder');
model.param.set('c_t', '0.004', 'Length of lumped port in
transponder');
model.param.set('a1_t', '0.00865', 'Half of Increasing length of
transponder in the first turn');
model.param.set('con', '0.0084');
model.param.set('Z', '0.002[m]', 'Coil Separation');
model.param.set('X', '0.00[m]', 'X Offset');
model.param.set('Y', '0.00[m]', 'Y Offset');
model.param.set('freq', '200000', 'Frequency (Hz)');
model.param.set('P', '1', 'Multiplier');

model.component.create('comp1', false);

model.component('comp1').geom.create('geom1', 3);

model.result.table.create('tbl1', 'Table');
model.result.table.create('tbl2', 'Table');
model.result.table.create('tbl3', 'Table');
model.result.table.create('evl3', 'Table');
model.result.table.create('tbl4', 'Table');
model.result.table.create('tbl5', 'Table');
model.result.table.create('tbl6', 'Table');

model.component('comp1').mesh.create('mesh1');

model.component('comp1').geom('geom1').geomRep('comsol');
model.component('comp1').geom('geom1').create('wp1', 'WorkPlane');
model.component('comp1').geom('geom1').feature('wp1').set('quickz', '-
Z/2');
model.component('comp1').geom('geom1').feature('wp1').set('displ',
{'X/2' 'Y/2'});
model.component('comp1').geom('geom1').feature('wp1').set('unite',
true);
model.component('comp1').geom('geom1').feature('wp1').geom.create('pol1
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
1').set('type', 'open');
```

```
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
1').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
1').set('table', {'-1*(a1-b)' '0';  ...
'-1*(a1-b)' '(a1-b)';  ...
'(a1-b)' '(a1-b)';  ...
'(a1-b)' '-1*(a1-b)';  ...
'-1*(a1+a-b)' '-1*(a1-b)';  ...
'-1*(a1+a-b)' '(a1+a-b)';  ...
'(a1+a-b)' '(a1+a-b)';  ...
'(a1+a-b)' '-1*(a1+a-b)';  ...
'-1*(a1+2*a-b)' '-1*(a1+a-b)';  ...
'-1*(a1+2*a-b)' '(a1+2*a-b)';  ...
'(a1+2*a-b)' '(a1+2*a-b)';  ...
'(a1+2*a-b)' '-1*(a1+2*a-b)';  ...
'-1*(a1+3*a-b)' '-1*(a1+2*a-b)';  ...
'-1*(a1+3*a-b)' '(a1+3*a-b)';  ...
'(a1+3*a-b)' '(a1+3*a-b)';  ...
'(a1+3*a-b)' '-1*(a1+3*a-b)';  ...
'-1*(a1+4*a-b)' '-1*(a1+3*a-b)';  ...
'-1*(a1+4*a-b)' '(a1+4*a-b)';  ...
'(a1+4*a-b)' '(a1+4*a-b)';  ...
'(a1+4*a-b)' '-1*(a1+4*a-b)';  ...
'-1*(a1+5*a-b)' '-1*(a1+4*a-b)';  ...
'-1*(a1+5*a-b)' '(a1+5*a-b)';  ...
'(a1+5*a-b)' '(a1+5*a-b)';  ...
'(a1+5*a-b)' '-1*(a1+5*a-b)';  ...
'-1*(a1+6*a-b)' '-1*(a1+5*a-b)';  ...
'-1*(a1+6*a-b)' '(a1+6*a-b)';  ...
'(a1+6*a-b)' '(a1+6*a-b)';  ...
'(a1+6*a-b)' '-1*(a1+6*a-b)';  ...
'-1*(a1+7*a-b)' '-1*(a1+6*a-b)';  ...
'-1*(a1+7*a-b)' '(a1+7*a-b)';  ...
'(a1+7*a-b)' '(a1+7*a-b)';  ...
'(a1+7*a-b)' '-1*(a1+7*a-b)';  ...
'-1*(a1+8*a-b)' '-1*(a1+7*a-b)';  ...
'-1*(a1+8*a-b)' '(a1+8*a-b)';  ...
'(a1+8*a-b)' '(a1+8*a-b)';  ...
'(a1+8*a-b)' '-1*(a1+8*a-b)';  ...
'-1*(a1+9*a-b)' '-1*(a1+8*a-b)';  ...
'-1*(a1+9*a-b)' '(a1+9*a-b)';  ...
'(a1+9*a-b)' '(a1+9*a-b)';  ...
'(a1+9*a-b)' '-1*(a1+9*a-b)';  ...
'-1*(a1+10*a-b)' '-1*(a1+9*a-b)';  ...
'-1*(a1+10*a-b)' '(a1+10*a-b)';  ...
'(a1+10*a-b)' '(a1+10*a-b)';  ...
'(a1+10*a-b)' '-1*(a1+10*a-b)';  ...
'-1*(a1+11*a-b)' '-1*(a1+10*a-b)';  ...
'-1*(a1+11*a-b)' '(a1+11*a-b)';  ...
'(a1+11*a-b)' '(a1+11*a-b)';  ...
'(a1+11*a-b)' '-1*(a1+11*a-b)';  ...
'-1*(a1+12*a-b)' '-1*(a1+11*a-b)';  ...
'-1*(a1+12*a-b)' '(a1+12*a-b)';  ...
'(a1+12*a-b)' '(a1+12*a-b)';  ...
'(a1+12*a-b)' '-1*(a1+12*a-b)';  ...
'-1*(a1-b-con)' '-1*(a1+12*a-b)'});
```

```
model.component('comp1').geom('geom1').feature('wp1').geom.create('pol2
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
2').set('type', 'open');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
2').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
2').set('table', {'-1*(a1+b)' '0';  ...
'-1*(a1+b)' '(a1+b)';  ...
'(a1+b)' '(a1+b)';  ...
'(a1+b)' '-1*(a1+b)';  ...
'-1*(a1+a+b)' '-1*(a1+b)';  ...
'-1*(a1+a+b)' '(a1+a+b)';  ...
'(a1+a+b)' '(a1+a+b)';  ...
'(a1+a+b)' '-1*(a1+a+b)';  ...
'-1*(a1+2*a+b)' '-1*(a1+a+b)';  ...
'-1*(a1+2*a+b)' '(a1+2*a+b)';  ...
'(a1+2*a+b)' '(a1+2*a+b)';  ...
'(a1+2*a+b)' '-1*(a1+2*a+b)';  ...
'-1*(a1+3*a+b)' '-1*(a1+2*a+b)';  ...
'-1*(a1+3*a+b)' '(a1+3*a+b)';  ...
'(a1+3*a+b)' '(a1+3*a+b)';  ...
'(a1+3*a+b)' '-1*(a1+3*a+b)';  ...
'-1*(a1+4*a+b)' '-1*(a1+3*a+b)';  ...
'-1*(a1+4*a+b)' '(a1+4*a+b)';  ...
'(a1+4*a+b)' '(a1+4*a+b)';  ...
'(a1+4*a+b)' '-1*(a1+4*a+b)';  ...
'-1*(a1+5*a+b)' '-1*(a1+4*a+b)';  ...
'-1*(a1+5*a+b)' '(a1+5*a+b)';  ...
'(a1+5*a+b)' '(a1+5*a+b)';  ...
'(a1+5*a+b)' '-1*(a1+5*a+b)';  ...
'-1*(a1+6*a+b)' '-1*(a1+5*a+b)';  ...
'-1*(a1+6*a+b)' '(a1+6*a+b)';  ...
'(a1+6*a+b)' '(a1+6*a+b)';  ...
'(a1+6*a+b)' '-1*(a1+6*a+b)';  ...
'-1*(a1+7*a+b)' '-1*(a1+6*a+b)';  ...
'-1*(a1+7*a+b)' '(a1+7*a+b)';  ...
'(a1+7*a+b)' '(a1+7*a+b)';  ...
'(a1+7*a+b)' '-1*(a1+7*a+b)';  ...
'-1*(a1+8*a+b)' '-1*(a1+7*a+b)';  ...
'-1*(a1+8*a+b)' '(a1+8*a+b)';  ...
'(a1+8*a+b)' '(a1+8*a+b)';  ...
'(a1+8*a+b)' '-1*(a1+8*a+b)';  ...
'-1*(a1+9*a+b)' '-1*(a1+8*a+b)';  ...
'-1*(a1+9*a+b)' '(a1+9*a+b)';  ...
'(a1+9*a+b)' '(a1+9*a+b)';  ...
'(a1+9*a+b)' '-1*(a1+9*a+b)';  ...
'-1*(a1+10*a+b)' '-1*(a1+9*a+b)';  ...
'-1*(a1+10*a+b)' '(a1+10*a+b)';  ...
'(a1+10*a+b)' '(a1+10*a+b)';  ...
'(a1+10*a+b)' '-1*(a1+10*a+b)';  ...
'-1*(a1+11*a+b)' '-1*(a1+10*a+b)';  ...
'-1*(a1+11*a+b)' '(a1+11*a+b)';  ...
'(a1+11*a+b)' '(a1+11*a+b)';  ...
'(a1+11*a+b)' '-1*(a1+11*a+b)';  ...
'-1*(a1+12*a+b)' '-1*(a1+11*a+b)';  ...
'-1*(a1+12*a+b)' '(a1+12*a+b)';  ...
```

```
'(a1+12*a+b)' '(a1+12*a+b)';  ...
'(a1+12*a+b)' '-1*(a1+12*a+b)';  ...
'-1*(a1-b-con)' '-1*(a1+12*a+b)'});
model.component('comp1').geom('geom1').feature('wp1').geom.create('pol3
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
3').set('type', 'open');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
3').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
3').set('table', {'-1*(a1-b)' '0'; '-1*(a1+b)' '0'});
model.component('comp1').geom('geom1').feature('wp1').geom.create('pol4
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
4').set('type', 'closed');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
4').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('pol
4').set('table', {'-1*(a1-b-con)' '-1*(a1+12*a-b)'; '-1*(a1-b-con)' '-
1*(a1+12*a+b)'});
model.component('comp1').geom('geom1').feature('wp1').geom.create('csol
1', 'ConvertToSolid');
model.component('comp1').geom('geom1').feature('wp1').geom.feature('cso
l1').selection('input').set({'pol1' 'pol2' 'pol3' 'pol4'});
model.component('comp1').geom('geom1').create('ext1', 'Extrude');
model.component('comp1').geom('geom1').feature('ext1').setIndex('distan
ce', 't1', 0);
model.component('comp1').geom('geom1').feature('ext1').selection('input
').set({'wp1'});
model.component('comp1').geom('geom1').create('blk1', 'Block');
model.component('comp1').geom('geom1').feature('blk1').set('pos', {'-
1*(a1+b)+X/2' '-2*b+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk1').set('size',
{'2*b' '2*b' 't2+t1'});
model.component('comp1').geom('geom1').create('blk2', 'Block');
model.component('comp1').geom('geom1').feature('blk2').set('pos', {'-
1*(a1+b)+X/2' '-1*(a1+12*a+b)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk2').set('size',
{'2*b' '2*b' 't2+t1'});
model.component('comp1').geom('geom1').create('blk5', 'Block');
model.component('comp1').geom('geom1').feature('blk5').set('pos', {'-
1*(a1+b)+X/2' '-1*(a1+12*a+b)+2*b+Y/2' 't2-Z/2'});
model.component('comp1').geom('geom1').feature('blk5').set('size',
{'2*b' '(a1+12*a+b)-4*b' 't1'});
model.component('comp1').geom('geom1').create('sph1', 'Sphere');
model.component('comp1').geom('geom1').feature('sph1').set('pos', [0 0
0]);
model.component('comp1').geom('geom1').feature('sph1').set('layername',
{'Layer 1'});
model.component('comp1').geom('geom1').feature('sph1').setIndex('layer'
, 'r2+Z/10+X/10+Y/10', 0);
model.component('comp1').geom('geom1').feature('sph1').set('r', 'r1');
model.component('comp1').geom('geom1').create('blk6', 'Block');
model.component('comp1').geom('geom1').feature('blk6').set('pos', {'-
1*(a1+b)+X/2' '-1*(a1+12*a+b+con/2)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk6').set('size',
{'2*b' 'con/2' 't1'});
```

```
model.component('comp1').geom('geom1').create('blk7', 'Block');
model.component('comp1').geom('geom1').feature('blk7').set('pos', {'-
1*(a1-b-con)+X/2' '-1*(a1+12*a+b+con/2)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk7').set('size',
{'2*b' 'con/2' 't1'});
model.component('comp1').geom('geom1').create('blk8', 'Block');
model.component('comp1').geom('geom1').feature('blk8').set('pos', {'-
1*(a1-b)+X/2' '-1*(a1+12*a+b+con/2)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk8').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').create('blk9', 'Block');
model.component('comp1').geom('geom1').feature('blk9').set('pos', {'-
1*(a1-b-con/3)+X/2' '-1*(a1+12*a+b+con/2)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk9').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').create('blk10', 'Block');
model.component('comp1').geom('geom1').feature('blk10').set('pos', {'-
1*(a1-b-2*con/3)+X/2' '-1*(a1+12*a+b+con/2)+Y/2' '-Z/2'});
model.component('comp1').geom('geom1').feature('blk10').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').create('wp2', 'WorkPlane');
model.component('comp1').geom('geom1').feature('wp2').set('quickz',
'Z/2');
model.component('comp1').geom('geom1').feature('wp2').set('displ', {'-
X/2' '-Y/2'});
model.component('comp1').geom('geom1').feature('wp2').set('unite',
true);
model.component('comp1').geom('geom1').feature('wp2').geom.create('pol1
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
1').set('type', 'open');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
1').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
1').set('table', {'-1*(a1-b)' '0';  ...
'-1*(a1-b)' '(a1-b)';  ...
'(a1-b)' '(a1-b)';  ...
'(a1-b)' '-1*(a1-b)';  ...
'-1*(a1+a-b)' '-1*(a1-b)';  ...
'-1*(a1+a-b)' '(a1+a-b)';  ...
'(a1+a-b)' '(a1+a-b)';  ...
'(a1+a-b)' '-1*(a1+a-b)';  ...
'-1*(a1+2*a-b)' '-1*(a1+a-b)';  ...
'-1*(a1+2*a-b)' '(a1+2*a-b)';  ...
'(a1+2*a-b)' '(a1+2*a-b)';  ...
'(a1+2*a-b)' '-1*(a1+2*a-b)';  ...
'-1*(a1+3*a-b)' '-1*(a1+2*a-b)';  ...
'-1*(a1+3*a-b)' '(a1+3*a-b)';  ...
'(a1+3*a-b)' '(a1+3*a-b)';  ...
'(a1+3*a-b)' '-1*(a1+3*a-b)';  ...
'-1*(a1+4*a-b)' '-1*(a1+3*a-b)';  ...
'-1*(a1+4*a-b)' '(a1+4*a-b)';  ...
'(a1+4*a-b)' '(a1+4*a-b)';  ...
'(a1+4*a-b)' '-1*(a1+4*a-b)';  ...
'-1*(a1+5*a-b)' '-1*(a1+4*a-b)';  ...
'-1*(a1+5*a-b)' '(a1+5*a-b)';  ...
'(a1+5*a-b)' '(a1+5*a-b)';  ...
```

```
'(a1+5*a-b)' '-1*(a1+5*a-b)';  ...
'-1*(a1+6*a-b)' '-1*(a1+5*a-b)';  ...
'-1*(a1+6*a-b)' '(a1+6*a-b)';  ...
'(a1+6*a-b)' '(a1+6*a-b)';  ...
'(a1+6*a-b)' '-1*(a1+6*a-b)';  ...
'-1*(a1+7*a-b)' '-1*(a1+6*a-b)';  ...
'-1*(a1+7*a-b)' '(a1+7*a-b)';  ...
'(a1+7*a-b)' '(a1+7*a-b)';  ...
'(a1+7*a-b)' '-1*(a1+7*a-b)';  ...
'-1*(a1+8*a-b)' '-1*(a1+7*a-b)';  ...
'-1*(a1+8*a-b)' '(a1+8*a-b)';  ...
'(a1+8*a-b)' '(a1+8*a-b)';  ...
'(a1+8*a-b)' '-1*(a1+8*a-b)';  ...
'-1*(a1+9*a-b)' '-1*(a1+8*a-b)';  ...
'-1*(a1+9*a-b)' '(a1+9*a-b)';  ...
'(a1+9*a-b)' '(a1+9*a-b)';  ...
'(a1+9*a-b)' '-1*(a1+9*a-b)';  ...
'-1*(a1+10*a-b)' '-1*(a1+9*a-b)';  ...
'-1*(a1+10*a-b)' '(a1+10*a-b)';  ...
'(a1+10*a-b)' '(a1+10*a-b)';  ...
'(a1+10*a-b)' '-1*(a1+10*a-b)';  ...
'-1*(a1+11*a-b)' '-1*(a1+10*a-b)';  ...
'-1*(a1+11*a-b)' '(a1+11*a-b)';  ...
'(a1+11*a-b)' '(a1+11*a-b)';  ...
'(a1+11*a-b)' '-1*(a1+11*a-b)';  ...
'-1*(a1+12*a-b)' '-1*(a1+11*a-b)';  ...
'-1*(a1+12*a-b)' '(a1+12*a-b)';  ...
'(a1+12*a-b)' '(a1+12*a-b)';  ...
'(a1+12*a-b)' '-1*(a1+12*a-b)';  ...
'-1*(a1-b-con)' '-1*(a1+12*a-b)'});
model.component('comp1').geom('geom1').feature('wp2').geom.create('pol2
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
2').set('type', 'open');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
2').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
2').set('table', {'-1*(a1+b)' '0';  ...
'-1*(a1+b)' '(a1+b)';  ...
'(a1+b)' '(a1+b)';  ...
'(a1+b)' '-1*(a1+b)';  ...
'-1*(a1+a+b)' '-1*(a1+b)';  ...
'-1*(a1+a+b)' '(a1+a+b)';  ...
'(a1+a+b)' '(a1+a+b)';  ...
'(a1+a+b)' '-1*(a1+a+b)';  ...
'-1*(a1+2*a+b)' '-1*(a1+a+b)';  ...
'-1*(a1+2*a+b)' '(a1+2*a+b)';  ...
'(a1+2*a+b)' '(a1+2*a+b)';  ...
'(a1+2*a+b)' '-1*(a1+2*a+b)';  ...
'-1*(a1+3*a+b)' '-1*(a1+2*a+b)';  ...
'-1*(a1+3*a+b)' '(a1+3*a+b)';  ...
'(a1+3*a+b)' '(a1+3*a+b)';  ...
'(a1+3*a+b)' '-1*(a1+3*a+b)';  ...
'-1*(a1+4*a+b)' '-1*(a1+3*a+b)';  ...
'-1*(a1+4*a+b)' '(a1+4*a+b)';  ...
'(a1+4*a+b)' '(a1+4*a+b)';  ...
'(a1+4*a+b)' '-1*(a1+4*a+b)';  ...
```

```
'-1*(a1+5*a+b)' '-1*(a1+4*a+b)';  ...
'-1*(a1+5*a+b)' '(a1+5*a+b)';  ...
'(a1+5*a+b)' '(a1+5*a+b)';  ...
'(a1+5*a+b)' '-1*(a1+5*a+b)';  ...
'-1*(a1+6*a+b)' '-1*(a1+5*a+b)';  ...
'-1*(a1+6*a+b)' '(a1+6*a+b)';  ...
'(a1+6*a+b)' '(a1+6*a+b)';  ...
'(a1+6*a+b)' '-1*(a1+6*a+b)';  ...
'-1*(a1+7*a+b)' '-1*(a1+6*a+b)';  ...
'-1*(a1+7*a+b)' '(a1+7*a+b)';  ...
'(a1+7*a+b)' '(a1+7*a+b)';  ...
'(a1+7*a+b)' '-1*(a1+7*a+b)';  ...
'-1*(a1+8*a+b)' '-1*(a1+7*a+b)';  ...
'-1*(a1+8*a+b)' '(a1+8*a+b)';  ...
'(a1+8*a+b)' '(a1+8*a+b)';  ...
'(a1+8*a+b)' '-1*(a1+8*a+b)';  ...
'-1*(a1+9*a+b)' '-1*(a1+8*a+b)';  ...
'-1*(a1+9*a+b)' '(a1+9*a+b)';  ...
'(a1+9*a+b)' '(a1+9*a+b)';  ...
'(a1+9*a+b)' '-1*(a1+9*a+b)';  ...
'-1*(a1+10*a+b)' '-1*(a1+9*a+b)';  ...
'-1*(a1+10*a+b)' '(a1+10*a+b)';  ...
'(a1+10*a+b)' '(a1+10*a+b)';  ...
'(a1+10*a+b)' '-1*(a1+10*a+b)';  ...
'-1*(a1+11*a+b)' '-1*(a1+10*a+b)';  ...
'-1*(a1+11*a+b)' '(a1+11*a+b)';  ...
'(a1+11*a+b)' '(a1+11*a+b)';  ...
'(a1+11*a+b)' '-1*(a1+11*a+b)';  ...
'-1*(a1+12*a+b)' '-1*(a1+11*a+b)';  ...
'-1*(a1+12*a+b)' '(a1+12*a+b)';  ...
'(a1+12*a+b)' '(a1+12*a+b)';  ...
'(a1+12*a+b)' '-1*(a1+12*a+b)';  ...
'-1*(a1-b-con)' '-1*(a1+12*a+b)'});
model.component('comp1').geom('geom1').feature('wp2').geom.create('pol3
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
3').set('type', 'open');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
3').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
3').set('table', {'-1*(a1-b)' '0'; '-1*(a1+b)' '0'});
model.component('comp1').geom('geom1').feature('wp2').geom.create('pol4
', 'Polygon');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
4').set('type', 'closed');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
4').set('source', 'table');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('pol
4').set('table', {'-1*(a1-b-con)' '-1*(a1+12*a-b)'; '-1*(a1-b-con)' '-
1*(a1+12*a+b)'});
model.component('comp1').geom('geom1').feature('wp2').geom.create('csol
1', 'ConvertToSolid');
model.component('comp1').geom('geom1').feature('wp2').geom.feature('cso
l1').selection('input').set({'pol1' 'pol2' 'pol3' 'pol4'});
model.component('comp1').geom('geom1').create('ext2', 'Extrude');
model.component('comp1').geom('geom1').feature('ext2').setIndex('distan
ce', 't1', 0);
```

```
model.component('comp1').geom('geom1').feature('ext2').selection('input
').set({'wp2'});
model.component('comp1').geom('geom1').create('blk11', 'Block');
model.component('comp1').geom('geom1').feature('blk11').set('pos', {'-
1*(a1+b)-X/2' '-2*b-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk11').set('size',
{'2*b' '2*b' 't2+t1'});
model.component('comp1').geom('geom1').create('blk12', 'Block');
model.component('comp1').geom('geom1').feature('blk12').set('pos', {'-
1*(a1+b)-X/2' '-1*(a1+12*a+b)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk12').set('size',
{'2*b' '2*b' 't2+t1'});
model.component('comp1').geom('geom1').create('blk13', 'Block');
model.component('comp1').geom('geom1').feature('blk13').set('pos', {'-
1*(a1+b)-X/2' '-1*(a1+12*a+b)+2*b-Y/2' 't2+Z/2'});
model.component('comp1').geom('geom1').feature('blk13').set('size',
{'2*b' '(a1+12*a+b)-4*b' 't1'});
model.component('comp1').geom('geom1').create('blk14', 'Block');
model.component('comp1').geom('geom1').feature('blk14').set('pos', {'-
1*(a1+b)-X/2' '-1*(a1+12*a+b+con/2)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk14').set('size',
{'2*b' 'con/2' 't1'});
model.component('comp1').geom('geom1').create('blk15', 'Block');
model.component('comp1').geom('geom1').feature('blk15').set('pos', {'-
1*(a1-b-con)-X/2' '-1*(a1+12*a+b+con/2)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk15').set('size',
{'2*b' 'con/2' 't1'});
model.component('comp1').geom('geom1').create('blk16', 'Block');
model.component('comp1').geom('geom1').feature('blk16').set('pos', {'-
1*(a1-b)-X/2' '-1*(a1+12*a+b+con/2)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk16').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').create('blk17', 'Block');
model.component('comp1').geom('geom1').feature('blk17').set('pos', {'-
1*(a1-b-con/3)-X/2' '-1*(a1+12*a+b+con/2)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk17').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').create('blk18', 'Block');
model.component('comp1').geom('geom1').feature('blk18').set('pos', {'-
1*(a1-b-2*con/3)-X/2' '-1*(a1+12*a+b+con/2)-Y/2' 'Z/2'});
model.component('comp1').geom('geom1').feature('blk18').set('size',
{'con/3' '2*b' 't1'});
model.component('comp1').geom('geom1').run;
model.component('comp1').geom('geom1').run('fin');

model.component('comp1').selection.create('sel1', 'Explicit');
model.component('comp1').selection('sel1').set([8 10 12 14 15 16 20
26]);
model.component('comp1').selection.create('sel3', 'Explicit');
model.component('comp1').selection('sel3').geom('geom1', 3, 2,
{'exterior'});
model.component('comp1').selection('sel3').set([6 7 8 9 10 11 12 13 14
15 16 17 20 21 26 27]);
model.component('comp1').selection('sel1').label('Reader');
model.component('comp1').selection('sel3').label('Reader boud');

model.component('comp1').material.create('mat1', 'Common');
```

```
model.component('comp1').material.create('mat2', 'Common');
model.component('comp1').material.create('mat3', 'Common');
model.component('comp1').material('mat1').selection.set([1 2 3 4 5 18
19 22 23 24 25]);
model.component('comp1').material('mat1').propertyGroup('def').func.cre
ate('eta', 'Piecewise');
model.component('comp1').material('mat1').propertyGroup('def').func.cre
ate('Cp', 'Piecewise');
model.component('comp1').material('mat1').propertyGroup('def').func.cre
ate('rho', 'Analytic');
model.component('comp1').material('mat1').propertyGroup('def').func.cre
ate('k', 'Piecewise');
model.component('comp1').material('mat1').propertyGroup('def').func.cre
ate('cs', 'Analytic');
model.component('comp1').material('mat1').propertyGroup.create('Refract
iveIndex', 'Refractive index');
model.component('comp1').material('mat2').selection.set([8 10 12 14 15
16 20 26]);
model.component('comp1').material('mat2').propertyGroup.create('linzRes
', 'Linearized resistivity');
model.component('comp1').material('mat3').selection.named('sel3');
model.component('comp1').material('mat3').propertyGroup.create('linzRes
', 'Linearized resistivity');

model.component('comp1').coordSystem.create('ie1', 'InfiniteElement');
model.component('comp1').coordSystem('ie1').selection.set([1 2 3 4 22
23 24 25]);

model.component('comp1').physics.create('mf', 'InductionCurrents',
'geom1');
model.component('comp1').physics('mf').selection.set([1 2 3 4 5 18 19
22 23 24 25]);
model.component('comp1').physics('mf').create('lport1', 'LumpedPort',
2);
model.component('comp1').physics('mf').feature('lport1').selection.set(
[234 235 236 238]);
model.component('comp1').physics('mf').create('lport2', 'LumpedPort',
2);
model.component('comp1').physics('mf').feature('lport2').selection.set(
[230 231 232 237]);
model.component('comp1').physics('mf').create('imp1', 'Impedance', 2);
model.component('comp1').physics('mf').feature('imp1').selection.named(
'sel3');

model.component('comp1').mesh('mesh1').create('dis1', 'Distribution');
model.component('comp1').mesh('mesh1').create('ftri1', 'FreeTri');
model.component('comp1').mesh('mesh1').create('swe1', 'Sweep');
model.component('comp1').mesh('mesh1').create('swe2', 'Sweep');
model.component('comp1').mesh('mesh1').create('swe3', 'Sweep');
model.component('comp1').mesh('mesh1').create('swe4', 'Sweep');
model.component('comp1').mesh('mesh1').create('swe5', 'Sweep');
model.component('comp1').mesh('mesh1').create('swe6', 'Sweep');
model.component('comp1').mesh('mesh1').create('ftet1', 'FreeTet');
model.component('comp1').mesh('mesh1').feature('ftri1').selection.set([
16 20 164 168 183 187 188 209 210 213 214 231 232 236 242 246 268
272]);
```

```
model.component('comp1').mesh('mesh1').feature('swe1').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe1').selection.set([1
2 3 4 22 23 24 25]);
model.component('comp1').mesh('mesh1').feature('swe2').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe2').selection.set([7
9 17 19 21 27]);
model.component('comp1').mesh('mesh1').feature('swe3').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe3').selection.set([6
8 16 18 20 26]);
model.component('comp1').mesh('mesh1').feature('swe4').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe4').selection.set([1
2 13]);
model.component('comp1').mesh('mesh1').feature('swe5').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe5').selection.set([1
0 14]);
model.component('comp1').mesh('mesh1').feature('swe6').selection.geom('
geom1', 3);
model.component('comp1').mesh('mesh1').feature('swe6').selection.set([1
1 15]);
model.component('comp1').mesh('mesh1').feature('ftet1').selection.geom(
'geom1', 3);
model.component('comp1').mesh('mesh1').feature('ftet1').selection.set([
5]);
model.component('comp1').mesh('mesh1').feature('ftet1').create('size1',
'Size');

model.result.table('tbl1').comments('Global Evaluation 1
(mf.Zport_1)');
model.result.table('tbl2').comments('Global Evaluation 2 {gev2}
(mf.Zport_2)');
model.result.table('tbl3').comments('Global Evaluation 1 {gev1}
(mf.Zport_2)');
model.result.table('evl3').label('Evaluation 3D');
model.result.table('evl3').comments('Interactive 3D values');
model.result.table('tbl4').comments('Global Evaluation 1 {gev1}
(mf.Zport_1, imag(mf.Zport1)/(2*pi)/freq,
real(mf.Vport_2)/real(mf.Vport_1)*imag(mf.Zport1)/(2*pi*freq))');
model.result.table('tbl5').comments('Global Evaluation 1 {gev1}
(mf.Zport_1, imag(mf.Zport_1)/(2*pi)/freq,
real(mf.Vport_2)/real(mf.Vport_1)*imag(mf.Zport_1)/(2*pi*freq))');
model.result.table('tbl6').comments('Global Evaluation 1 {gev1}
(mf.Zport_1, imag(mf.Zport_1)/(2*pi)/freq,
real(mf.Vport_2/(mf.Vport_1))*(imag(mf.Zport_1)/(2*pi)/freq),
mf.Vport_2, mf.Vport_1)');

model.capeopen.label('Thermodynamics Package');

model.component('comp1').view('view1').set('renderwireframe', true);
model.component('comp1').view('view1').set('showgrid', false);
model.component('comp1').view('view1').set('scenelight', false);
model.component('comp1').view('view1').set('transparency', true);
```

```
model.component('comp1').view('view2').axis.set('xmin', -
0.257860004901886);
model.component('comp1').view('view2').axis.set('xmax',
0.2290651798248291);
model.component('comp1').view('view2').axis.set('ymin', -
0.12917043268680573);
model.component('comp1').view('view2').axis.set('ymax',
0.1293497085571289);
model.component('comp1').view('view3').axis.set('xmin', -
0.20479559898376465);
model.component('comp1').view('view3').axis.set('xmax',
0.20479559898376465);
model.component('comp1').view('view3').axis.set('ymin', -0.125);
model.component('comp1').view('view3').axis.set('ymax', 0.125);

model.component('comp1').material('mat1').label('Air');
model.component('comp1').material('mat1').set('family', 'air');
model.component('comp1').material('mat1').propertyGroup('def').func('et
a').set('arg', 'T');
model.component('comp1').material('mat1').propertyGroup('def').func('et
a').set('pieces', {'200.0' '1600.0' '-8.38278E-7+8.35717342E-8*T^1-
7.69429583E-11*T^2+4.6437266E-14*T^3-1.06585607E-17*T^4'});
model.component('comp1').material('mat1').propertyGroup('def').func('Cp
').set('arg', 'T');
model.component('comp1').material('mat1').propertyGroup('def').func('Cp
').set('pieces', {'200.0' '1600.0' '1047.63657-
0.372589265*T^1+9.45304214E-4*T^2-6.02409443E-7*T^3+1.2858961E-
10*T^4'});
model.component('comp1').material('mat1').propertyGroup('def').func('rh
o').set('expr', 'pA*0.02897/8.314/T');
model.component('comp1').material('mat1').propertyGroup('def').func('rh
o').set('args', {'pA' 'T'});
model.component('comp1').material('mat1').propertyGroup('def').func('rh
o').set('dermethod', 'manual');
model.component('comp1').material('mat1').propertyGroup('def').func('rh
o').set('argders', {'pA' 'd(pA*0.02897/8.314/T,pA)'; 'T'
'd(pA*0.02897/8.314/T,T)'});
model.component('comp1').material('mat1').propertyGroup('def').func('rh
o').set('plotargs', {'pA' '0' '1'; 'T' '0' '1'});
model.component('comp1').material('mat1').propertyGroup('def').func('k'
).set('arg', 'T');
model.component('comp1').material('mat1').propertyGroup('def').func('k'
).set('pieces', {'200.0' '1600.0' '-0.00227583562+1.15480022E-4*T^1-
7.90252856E-8*T^2+4.11702505E-11*T^3-7.43864331E-15*T^4'});
model.component('comp1').material('mat1').propertyGroup('def').func('cs
').set('expr', 'sqrt(1.4*287*T)');
model.component('comp1').material('mat1').propertyGroup('def').func('cs
').set('args', {'T'});
model.component('comp1').material('mat1').propertyGroup('def').func('cs
').set('dermethod', 'manual');
model.component('comp1').material('mat1').propertyGroup('def').func('cs
').set('argders', {'T' 'd(sqrt(1.4*287*T),T)'});
model.component('comp1').material('mat1').propertyGroup('def').func('cs
').set('plotargs', {'T' '0' '1'});
model.component('comp1').material('mat1').propertyGroup('def').set('rel
permeability', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
```

```
model.component('comp1').material('mat1').propertyGroup('def').set('rel
permittivity', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat1').propertyGroup('def').set('dyn
amicviscosity', 'eta(T[1/K])[Pa*s]');
model.component('comp1').material('mat1').propertyGroup('def').set('rat
ioofspecificheat', '1.4');
model.component('comp1').material('mat1').propertyGroup('def').set('ele
ctricconductivity', {'0.0001[S/m]' '0' '0' '0' '0.0001[S/m]' '0' '0'
'0' '0.0001[S/m]'});
model.component('comp1').material('mat1').propertyGroup('def').set('hea
tcapacity', 'Cp(T[1/K])[J/(kg*K)]');
model.component('comp1').material('mat1').propertyGroup('def').set('den
sity', 'rho(pA[1/Pa],T[1/K])[kg/m^3]');
model.component('comp1').material('mat1').propertyGroup('def').set('the
rmalconductivity', {'k(T[1/K])[W/(m*K)]' '0' '0' '0'
'k(T[1/K])[W/(m*K)]' '0' '0' '0' 'k(T[1/K])[W/(m*K)]'});
model.component('comp1').material('mat1').propertyGroup('def').set('sou
ndspeed', 'cs(T[1/K])[m/s]');
model.component('comp1').material('mat1').propertyGroup('def').addInput
('temperature');
model.component('comp1').material('mat1').propertyGroup('def').addInput
('pressure');
model.component('comp1').material('mat1').propertyGroup('RefractiveInde
x').set('n', '');
model.component('comp1').material('mat1').propertyGroup('RefractiveInde
x').set('ki', '');
model.component('comp1').material('mat1').propertyGroup('RefractiveInde
x').set('n', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat1').propertyGroup('RefractiveInde
x').set('ki', {'0' '0' '0' '0' '0' '0' '0' '0' '0'});
model.component('comp1').material('mat2').active(false);
model.component('comp1').material('mat2').label('Copper');
model.component('comp1').material('mat2').set('family', 'copper');
model.component('comp1').material('mat2').propertyGroup('def').set('rel
permeability', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat2').propertyGroup('def').set('ele
ctricconductivity', {'5.998e7[S/m]' '0' '0' '0' '5.998e7[S/m]' '0' '0'
'0' '5.998e7[S/m]'});
model.component('comp1').material('mat2').propertyGroup('def').set('hea
tcapacity', '385[J/(kg*K)]');
model.component('comp1').material('mat2').propertyGroup('def').set('rel
permittivity', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat2').propertyGroup('def').set('emi
ssivity', '0.5');
model.component('comp1').material('mat2').propertyGroup('def').set('den
sity', '8700[kg/m^3]');
model.component('comp1').material('mat2').propertyGroup('def').set('the
rmalconductivity', {'400[W/(m*K)]' '0' '0' '0' '400[W/(m*K)]' '0' '0'
'0' '400[W/(m*K)]'});
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'rho0', '');
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'alpha', '');
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'Tref', '');
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'rho0', '1.72e-8[ohm*m]');
```

```
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'alpha', '3.9e-3[1/K]');
model.component('comp1').material('mat2').propertyGroup('linzRes').set(
'Tref', '273.15[K]');
model.component('comp1').material('mat2').propertyGroup('linzRes').addI
nput('temperature');
model.component('comp1').material('mat3').label(' Copper 1');
model.component('comp1').material('mat3').set('family', 'copper');
model.component('comp1').material('mat3').propertyGroup('def').set('rel
permeability', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat3').propertyGroup('def').set('ele
ctricconductivity', {'5.998e7[S/m]' '0' '0' '0' '5.998e7[S/m]' '0' '0'
'0' '5.998e7[S/m]'});
model.component('comp1').material('mat3').propertyGroup('def').set('hea
tcapacity', '385[J/(kg*K)]');
model.component('comp1').material('mat3').propertyGroup('def').set('rel
permittivity', {'1' '0' '0' '0' '1' '0' '0' '0' '1'});
model.component('comp1').material('mat3').propertyGroup('def').set('emi
ssivity', '0.5');
model.component('comp1').material('mat3').propertyGroup('def').set('den
sity', '8700[kg/m^3]');
model.component('comp1').material('mat3').propertyGroup('def').set('the
rmalconductivity', {'400[W/(m*K)]' '0' '0' '0' '400[W/(m*K)]' '0' '0'
'0' '400[W/(m*K)]'});
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'rho0', '');
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'alpha', '');
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'Tref', '');
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'rho0', '1.72e-8[ohm*m]');
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'alpha', '3.9e-3[1/K]');
model.component('comp1').material('mat3').propertyGroup('linzRes').set(
'Tref', '273.15[K]');
model.component('comp1').material('mat3').propertyGroup('linzRes').addI
nput('temperature');

model.component('comp1').coordSystem('ie1').set('ScalingType',
'Spherical');

model.component('comp1').physics('mf').prop('PortSweepSettings').set('f
ormat', 'DB');
model.component('comp1').physics('mf').feature('al1').set('AllowDepreca
tedCurves', true);
model.component('comp1').physics('mf').feature('lport1').set('PortType'
, 'UserDefined');
model.component('comp1').physics('mf').feature('lport1').set('TerminalT
ype', 'Current');
model.component('comp1').physics('mf').feature('lport1').set('hPort',
'2*b');
model.component('comp1').physics('mf').feature('lport1').set('wPort',
'con/3');
model.component('comp1').physics('mf').feature('lport1').set('ahPort',
[1; 0; 0]);
```

```
model.component('comp1').physics('mf').feature('lport2').set('PortType'
, 'UserDefined');
model.component('comp1').physics('mf').feature('lport2').set('I0',
'0[A]');
model.component('comp1').physics('mf').feature('lport2').set('TerminalT
ype', 'Current');
model.component('comp1').physics('mf').feature('lport2').set('hPort',
'2*b');
model.component('comp1').physics('mf').feature('lport2').set('wPort',
'con/3');
model.component('comp1').physics('mf').feature('lport2').set('ahPort',
[1; 0; 0]);

model.component('comp1').mesh('mesh1').feature('size').set('hauto', 3);
model.component('comp1').mesh('mesh1').feature('ftet1').feature('size1'
).set('hauto', 3);
model.component('comp1').mesh('mesh1').run;

model.study.create('std1');
model.study('std1').create('freq', 'Frequency');
model.study('std1').feature('freq').set('activate', {'mf' 'on'});

model.sol.create('sol1');
model.sol('sol1').study('std1');
model.sol('sol1').attach('std1');
model.sol('sol1').create('st1', 'StudyStep');
model.sol('sol1').create('v1', 'Variables');
model.sol('sol1').create('s1', 'Stationary');
model.sol('sol1').feature('s1').create('p1', 'Parametric');
model.sol('sol1').feature('s1').create('fc1', 'FullyCoupled');
model.sol('sol1').feature('s1').create('i1', 'Iterative');
model.sol('sol1').feature('s1').feature('i1').create('mg1',
'Multigrid');
model.sol('sol1').feature('s1').feature('i1').feature('mg1').feature('p
r').create('sv1', 'SORVector');
model.sol('sol1').feature('s1').feature('i1').feature('mg1').feature('p
o').create('sv1', 'SORVector');
model.sol('sol1').feature('s1').feature.remove('fcDef');

model.result.dataset.create('dset7', 'Solution');
model.result.dataset.remove('dset1');
model.result.export.create('data1', 'Data');

model.study('std1').feature('freq').set('plist', 200000);
model.study('std1').feature('freq').set('preusesol', 'yes');
model.study('std1').feature('freq').set('discretization', {'mf'
'physics'});

model.sol('sol1').attach('std1');
model.sol('sol1').feature('v1').set('clistctrl', {'p1'});
model.sol('sol1').feature('v1').set('cname', {'freq'});
model.sol('sol1').feature('v1').set('clist', {'200000[Hz]'});
model.sol('sol1').feature('s1').feature('p1').set('pname', {'freq'});
model.sol('sol1').feature('s1').feature('p1').set('plistarr',
[200000]);
model.sol('sol1').feature('s1').feature('p1').set('punit', {'Hz'});
```

```
model.sol('sol1').feature('s1').feature('p1').set('pcontinuationmode',
'no');
model.sol('sol1').feature('s1').feature('p1').set('preusesol', 'yes');
model.sol('sol1').feature('s1').feature('i1').set('linsolver',
'bicgstab');
model.sol('sol1').feature('s1').feature('i1').set('prefuntype',
'right');
model.sol('sol1').runAll;

model.result.export('data1').set('looplevelinput', {'manual'});
model.result.export('data1').set('expr', {'mf.Zport_1'
'imag(mf.Zport_1)/(2*pi)/freq'
'real(mf.Vport_2/(mf.Vport_1))*(imag(mf.Zport_1)/(2*pi)/freq)'
'mf.freq'});
model.result.export('data1').set('unit', {['ohm' ] 'H' 'H' 'Hz'});
model.result.export('data1').set('descr', {'Lumped port impedance'
'Coil 1 Impedance' 'Mutual Inductance' 'Frequency'});
model.result.export('data1').set('filename',
'/volume/NFS/pnf6/rSim/COMSOL/Wireless /Mutual
Inductance/Simulations/10kHz to 17MHz onriginal model modified
mesh.txt');
model.result.export('data1').set('smooth', 'internal');

out = model;
```

8. References

[1] Farrar Charles R and Worden Keith, "An introduction to structural health monitoring," *Philos. Trans. R. Soc. Math. Phys. Eng. Sci.*, vol. 365, no. 1851, pp. 303–315, Feb. 2007.

[2] S. W. James, M. L. Dockney, and R. P. Tatam, "Simultaneous independent temperature and strain measurement using in-fibre Bragg grating sensors," *Electron. Lett.*, vol. 32, no. 12, pp. 1133–1134, Jun. 1996.

[3] S. M. Melle, A. T. Alavie, S. Karr, T. Coroy, K. Liu, and R. M. Measures, "A Bragg grating-tuned fiber laser strain sensor system," *IEEE Photonics Technol. Lett.*, vol. 5, no. 2, pp. 263–266, Feb. 1993.

[4] "OSA | Multiplexed fiber Bragg grating strain-sensor system with a fiber Fabry–Perot wavelength filter." [Online]. Available: https://www.osapublishing.org/ol/abstract.cfm?uri=ol-18-16-1370. [Accessed: 05-Jul-2018].

[5] "OSA | Fiber-optic Bragg grating strain sensor with drift-compensated high-resolution interferometric wavelength-shift detection." [Online]. Available: https://www.osapublishing.org/ol/abstract.cfm?uri=ol-18-1-72. [Accessed: 05-Jul-2018].

[6] I. Kang, M. J. Schulz, J. H. Kim, V. Shanov, and D. Shi, "A carbon nanotube strain sensor for structural health monitoring," *Smart Mater. Struct.*, vol. 15, no. 3, p. 737, 2006.

[7] P. Dharap, Z. Li, S. Nagarajaiah, and E. V. Barrera, "Nanotube film based on single-wall carbon nanotubes for strain sensing," *Nanotechnology*, vol. 15, no. 3, p. 379, 2004.

[8] Bao *et al.*, "Experimental Study of Highly Sensitive Sensor Using a Surface Acoustic Wave Resonator for Wireless Strain Detection," *Jpn. J. Appl. Phys.*, vol. 51, no. 7S, p. 07GC23, Jul. 2012.

[9] Á. Ledeczi, T. Hay, P. Volgyesi, D. R. Hay, A. Nadas, and S. Jayaraman, "Wireless Acoustic Emission Sensor Network for Structural Monitoring," *IEEE Sens. J.*, vol. 9, no. 11, pp. 1370–1377, Nov. 2009.

[10] A. Qusba, A. K. RamRakhyani, J. H. So, G. J. Hayes, M. D. Dickey, and G. Lazzi, "On the Design of Microfluidic Implant Coil for Flexible Telemetry System," *IEEE Sens. J.*, vol. 14, no. 4, pp. 1074–1080, Apr. 2014.

[11] X. Xu and H. Huang, "Battery-less wireless interrogation of microstrip patch antenna for strain sensing," *Smart Mater. Struct.*, vol. 21, no. 12, p. 125007, 2012.

[12] A. Daliri, A. Galehdar, S. John, C. H. Wang, W. S. T. Rowe, and K. Ghorbani, "Wireless strain measurement using circular microstrip patch antennas," *Sens. Actuators Phys.*, vol. 184, pp. 86–92, Sep. 2012.

[13] X. Yi, T. Wu, Y. Wang, R. T. Leon, M. M. Tentzeris, and G. Lantz, "Passive wireless smart-skin sensor using RFID-based folded patch antennas," *Int. J. Smart Nano Mater.*, vol. 2, no. 1, pp. 22–38, Feb. 2011.

[14] V. Kalinin, "Passive wireless strain and temperature sensors based on SAW devices," in *Proceedings. 2004 IEEE Radio and Wireless Conference (IEEE Cat. No.04TH8746)*, 2004, pp. 187–190.

[15] R. Stoney, D. Geraghty, and G. E. O'Donnell, "Characterization of Differentially Measured Strain Using Passive Wireless Surface Acoustic Wave (SAW) Strain Sensors," *IEEE Sens. J.*, vol. 14, no. 3, pp. 722–728, Mar. 2014.
[16] "Wireless passive SAW sensor systems for industrial and domestic applications - IEEE Conference Publication." [Online]. Available: https://ieeexplore.ieee.org/abstract/document/717961/. [Accessed: 05-Jul-2018].
[17] R. W. Brocato, "Passive Wireless Sensing Tags Nasa Inflatable Structures.," Sandia National Laboratories, SAND2006-1288, Mar. 2006.
[18] B. Donohoe, D. Geraghty, and G. E. O'Donnell, "Wireless Calibration of a Surface Acoustic Wave Resonator as a Strain Sensor," *IEEE Sens. J.*, vol. 11, no. 4, pp. 1026–1032, Apr. 2011.
[19] J. Chuang, D. J. Thomson, and G. E. Bridges, "Embeddable wireless strain sensor based on resonant rf cavities," *Rev. Sci. Instrum.*, vol. 76, no. 9, p. 094703, Sep. 2005.
[20] W. C. Wilson *et al.*, "Orthogonal Frequency Coded SAW Sensors for Aerospace SHM Applications," *IEEE Sens. J.*, vol. 9, no. 11, pp. 1546–1556, Nov. 2009.
[21] Y. Jia, K. Sun, F. J. Agosto, and M. T. Quiñones, "Design and characterization of a passive wireless strain sensor," *Meas. Sci. Technol.*, vol. 17, no. 11, p. 2869, 2006.
[22] J. C. Butler, A. J. Vigliotti, F. W. Verdi, and S. M. Walsh, "Wireless, passive, resonant-circuit, inductively coupled, inductive strain sensor," *Sens. Actuators Phys.*, vol. 102, no. 1–2, pp. 61–66, Dec. 2002.
[23] R. Nopper, R. Niekrawietz, and L. Reindl, "Wireless Readout of Passive LC Sensors," *IEEE Trans. Instrum. Meas.*, vol. 59, no. 9, pp. 2450–2457, Sep. 2010.
[24] R. Matsuzaki and A. Todoroki, "Passive wireless strain monitoring of actual tire using capacitance–resistance change and multiple spectral features," *Sens. Actuators Phys.*, vol. 126, no. 2, pp. 277–286, Feb. 2006.
[25] A. Mazzeo, A. Stein, D. Trumper, and R. Hocken, "Atomic force microscope for accurate dimensional metrology," *Precis. Eng.*, vol. 33, no. 2, pp. 135–149, 2009.
[26] A. D. Mazzeo, "Accurate capacitive metrology for atomic force microscopy," Thesis, Massachusetts Institute of Technology, 2005.
[27] P. Ferri, "A Piezoelectric-based Wireless Sensor for Strain Monitoring." .
[28] L. Heinzle, "Measuring the Mutual Interaction between Coaxial Cylindrical Coils with the Bode 100," *Smart Meas. Solut.*, pp. 1–11, 2013.
[29] L. Heinzle, "A Theoretical Model for Mutual Interaction between Coaxial Cylindrical Coils," *Omicron Lab*, pp. 1–15, 2011.
[30] "Modeling of a 3D Inductor." [Online]. Available: https://www.comsol.com/model/modeling-of-a-3d-inductor-10299. [Accessed: 11-Feb-2019].
[31] Y. Cheng and Y. Shu, "A New Analytical Calculation of the Mutual Inductance of the Coaxial Spiral Rectangular Coils," *IEEE Trans. Magn.*, vol. 50, no. 4, pp. 1–6, Apr. 2014.
[32] "Interpolation for 3-D gridded data in meshgrid format - MATLAB interp3." [Online]. Available: https://www.mathworks.com/help/matlab/ref/interp3.html. [Accessed: 10-Jun-2019].

[33] “Interpolate 2-D or 3-D scattered data - MATLAB.” [Online]. Available: https://www.mathworks.com/help/matlab/ref/scatteredinterpolant.html. [Accessed: 17-Jun-2019].
[34] “Approximate n-dimensional function - Simulink.” [Online]. Available: https://www.mathworks.com/help/simulink/slref/ndlookuptable.html. [Accessed: 10-Jun-2019].
[35] “Color coded 3D scatterplot - File Exchange - MATLAB Central.” [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/9519. [Accessed: 06-Jun-2019].
[36] “Bode Automation Interface: Bode Automation Interface.” [Online]. Available: https://www.omicron-lab.com/fileadmin/assets/software/AutomationInterfaceHelp/output/html/index.html. [Accessed: 25-Mar-2019].