Magnetogram - Geomagnetic IoT services with LoRaWAN network and compatible embedded devices and sensors

UiT / NORGES ARKTISKE UNIVERSITET

Jon H.L Johansen, Anton Garri Fagerbakk, Torbjørn Tveito

May 14, 2018

1 Introduction and Motivation

In this project we set out to create a proof of concept for a high accuracy magnetometer using micro-sensors, and LoRaWAN-technology. The motivation for the project is to drive cost down for research in this field as well to test out different types of usage of the popular IoT technology.

There is a large community of physicists who study the ionosphere with radar facilities. In order to make sense of the measurements done with radar, one must know a variety of physical properties of the medium being investigated. In plasmas, these properties can change as a function of the magnetic field present, so accurate measurements of the magnetic field in regions where studies are being done can be a significant help for the scientific community.
Magnetic fields change how reflective plasmas are to electromagnetic radiation, because the frequency and velocity of electrons in the plasma is proportional to the magnitude of the magnetic field. Magnetometer installations able to provide such a high accuracy are expensive and are easily affected by nearby sources of noise, and so these installations are few and far between.
If, however, one was able to produce a cheap, small and easy to create device able to, through basic statistical methods, provide measurements with a "good enough" accuracy, an array of measuring devices could be placed in the area surrounding a radar facility.
This would make it possible to create a model of the spatial changes in the magnetic field through the ionosphere, which could potentially significantly assist in modeling the plasma.

In order to accurately measure changes in the magnetic field due to changes in the ionosphere, we must account for and minimize sources of random noise. This can include passing cars, nearby current-carrying wires, and the box itself. If our project is successful, we will obtain a magnetogram that closely matches both the magnitude and small-scale variations in the "official" UiT magnetogram. Currents in the box should provide a low, but relatively constant noise that should never exceed approximately 400 nT. This should be a near linear offset, and could easily be taken into account.

Our goal is then to create a proof-of-concept device able to cheaply obtain a magneto-gram that provides similar results to the official university magneto-gram for Tromsø with a high accuracy. If we are able to show that this is achievable, it could potentially be used as a supportive dataset to the EISCAT-3D facility, which will be used to create models of the arctic ionosphere.

"I disagree strongly with whatever work this quote is attached to" -Dr. Randal Munroe, PhD.

Contents

2 Technical and Physics Background

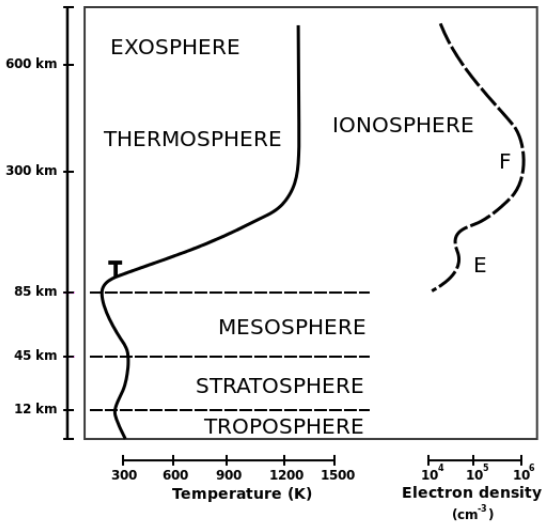
2.1 I2C

I^2C (Inter-Integrated Circuit), pronounced I-squared-C, is a synchronous, multi-master, multi-slave, packet switched, single-ended, serial computer bus invented in 1982 by Philips Semiconductor (now NXP Semiconductors). It is widely used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication.[1]

The I2C is the standard bus operator for this project as all libraries for sensors are written using I2C micropython libraries.

2.2 The Auroral Electrojet

The auroral electrojet is a transport current of electrons in the E-region of the ionosphere between 100 and 150 kilometers above the surface of the Earth. As a first-order approximation, it can be treated like a straight wire which carries a variable current that moves across the sky, both in the height direction and parallel to the ground. This current will then cause changes to the geomagnetic field based on distance to the measuring point and the magnitude of the current. If there are several different locations where measurements are done simultaneously, the measurements can then be used to find the distance to the electrojet for all measuring points, thereby triangulating the position of this current-carrying wire.



3 Design & Implementation

The design can be divided into two layers, hardware and software(which features a tiny bit of front-end). The device hardware is built on a LoPy, using an Pycom extension board.

3.1 Hardware

The hardware consist of:

- Hardplastic water-and-dust-proof encasing
- Sensors
  - MAG3110
  - MPU9265
  - STEVAL-MKI137v1 - LIS3MDL (unused in production)
  - Adafruit Ultimate GPS, FeatherWing (unused)
- LoPy
- Pycom extension board
- LoRaWAN Antenna
- Lithium Ion Battery
- LoPy connector to enable deep-sleep

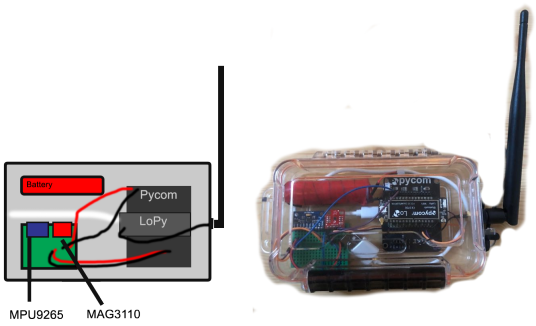
The sensors used in this project is MAG3110, STEVAL-MKI137v1 and MPU9265.

The MAG3110 is a small, low-power digital 3D magnetic sensor with a wide dynamic range to allow operation in PCBs with high extraneous magnetic fields.
The MPU-9265 device combine a 3-axis gyroscope, 3-axis accelerometer and 3-axis compass in the same chip together with an on board Digital Motion Processor capable of processing the complex Motion Fusion algorithms.
The LIS3MDL is an ultra-low-power high-performance three-axis magnetic sensor.
To power the Pycom we have a 5*2cm cylinder lithium ion battery. The LoRaWAN antenna is detachable for convenience, and can be screwed onto the the outside of the casing.

Most of these sensors have both I2C and SPI interface. Our software utilizes the I2C interface.

The hardware sits inside a plastic waterproof hard case, this is for the protection of the device, since the placement of the device requires it to be placed in remote outdoors areas, and should be able to sustain most types of weather. The hardware require a sturdy and waterproof environment to avoid damage to any component. To be able to easily re-arrange the placement of the hardware inside the casing, all hardware has been attached to the encasement using Velcro-tape.

The sensors are rigged so that their axis align as perfect as possible. For the prototype, this has been done using a small breadboard, which aligns the MPU9265 and the MAG3110. The plastic see through casing is useful as the Pycom integrated led lights will display connection status to the LoRa network.

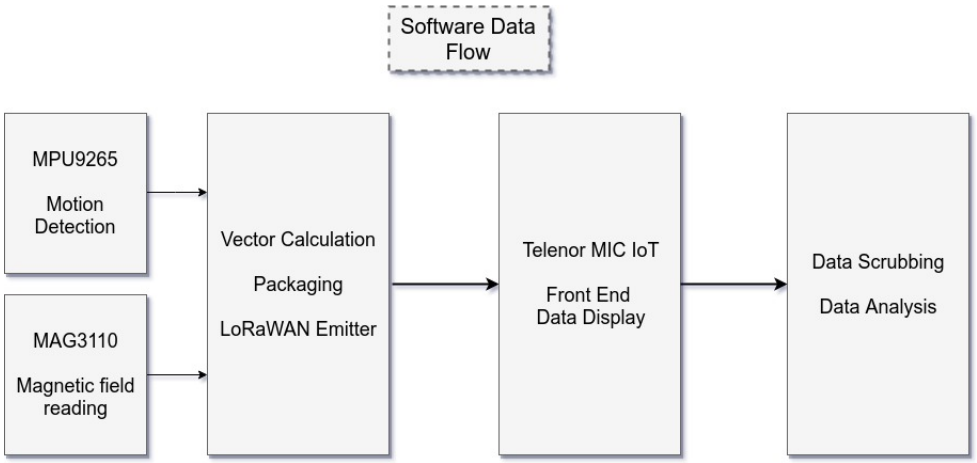


3.2 Software

The sensors themselves, calculations, shipping and scrubbing data all require a software. A description of the calculations and the software is to be described in the following chapter.

3.2.1 Computing Flowchart

The flowchart beneath shows the overall data flow from the sensors to the view in the front-end. The sensor libraries are written to compute and convert their respective types of data. Methods within the libraries are made so that the "middle stage" only has to worry about combining those data and shipping them of to the front end. As of right now some calculations are done on a device basis, as the calculations are rather small. For more scientific purposes we have written software to further scrub the data, but this can be placed after the MIC front-end display

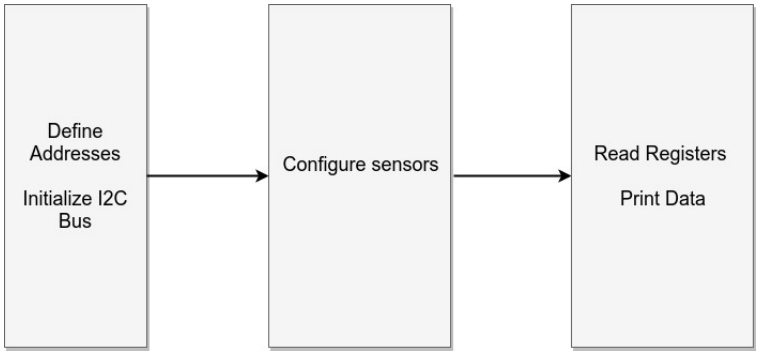


3.2.2 Computation, Where & Why?

The choice of pre-calculating some on the device is based on the trade-off of shipping large packages through the LoRaWAN versus the cost of pre-calculating the data on board. The first advantage to calculate on board is that the packets sent are smaller, and thus less of a cost in terms of network usage. The LoRaWAN network has some limitations and while its entirely possible for us to ship large amounts of data through it, it would surpass the recommended network usage for LoRaWAN. As its a small network, which has shared resources and can be considered limited, we took it into consideration. The pre-calculations made are 1500 measurements per 30 second range, boiled down to one result. Meaning if we were to move this calculation elsewhere it would require 1500 times more data to be shipped.

3.2.3 Sensor Software

Libraries are created for all the sensors used in this project. The library designs are based on python-classes. The libraries contain methods to configure and setup, retrieve, print and convert the data from that specific sensor so that the parent function can use it without having to convert elsewhere where specific information about the sensors conversion rates i.e does not belong. These libraries are based on the data-sheets found on the Internet for the sensors to the degree where it was possible. These libraries are written so that they can easily be re-purposed at a later time.



MAG3110 When initialized it sets the correct modus of operation as well as defining some general resources for the sensor. The library features a fetch data function which fetches and returns a tuple containing the data. A set of conversion methods for correctly converting the data from two sets of bytes into an int, as well as from a raw value to the nanotesla, which is the desired unit of measurement. The sensor also features a low accuracy temperature sensor, but this has not been used in our project, as the MPU9265 sensor yields more accurate results. A separate function for printing the data (used for debugging) is the final method of the class.

MPU9265 & AK8963 When the sensor is initialized it runs the configuration for the MPU and the AK sensors. The configuration takes the sensors out of sleep mode, prepares the right modes for reading continuous output. The sensors reads magnetic field, gyroscope, accelerometer and temperature data continuously. The data is retrieved using a fetching method which takes the readings from the gyroscope, accelerometer and temperature and packs it into a tuple. The tuple can then be accessed for readings and further calculations. The accelerometer data is also used to calculate the vector of the sensor to guarantee that the magnetic data is correct.

LIS3MDL The LIS3MDL sensor has a under-developed library due to an conclusion early on in the project about the quality of data it outputs. The superior accuracy MAG3110 magnetometer debunked the need for use of this sensor, and the development of this library was stagnated. Thus it only contains basic initialization, and fetching data. It is in any way functional, but does not feature all the conversion and printing functions necessary to integrate it with our project.

3.2.4 Status Lights

The devices see through casing enables us to use the LED lights to display status of the device. It currently features these light statuses:

Blue Debug mode. The device has been intentionally set to not connect to the LoRA network, and all output will only be viewable with REPL.

Red LoRA mode. The device is set in LoRA mode and awaits initial connection to the LoRA network. Once a connection has been established the light will turn green.

Green Data reading mode. The device is in LoRA mode and is currently reading data from the sensors. Once the data has been read, it will attempt sending the data and the light will turn yellow.

Yellow Sending data mode. The device is in LoRA mode and attempting to send data. Once data has been sent the device will enter data reading mode and the light will turn green.

3.2.5 Mitigating False Data

The MAG3110 sensor is rigged to perform 1500 measurements within a 30 second range, and then average the measurements, to further reduce deviating values. It also increases the accuracy of the measurements, due to the fact that the sensor originally only has an accuracy of 33 nT, while averaging them with a possible float value will yield any value in between that range.

Data Rotation

In order to acquire a reliable coordinate system, we have implemented a rotation function. In Layman's terms it fixes the data if the box were to be angled wrongly, by attempting to figure out what the data would have been if the box was placed in the correct orientation. The function takes in two three-dimensional vectors (Magnetic data and accelerometer), and finds the rotation matrix required to make one of the vectors equal to a preset vector, (0,0,1) in this case. It then dot-multiplies the other vector with the same rotation matrix. This means that when the vector used to define the matrix is the force vector supplied by the accelerometer, the positive z-direction is defined as "down".

The rotation matrix is a unitary matrix, so when performing a dot product of our magnetometer sensor and the rotation matrix, the magnitude of the measured magnetic field is unchanged (although there are numerical errors in calculation, but these are on a scale of 10^{-16}, and are therefore negligible). This allows us to conserve the accuracy of our measurements through the operation.

The rotation matrix between vectors a and b is found through several steps. First, one finds the cross product, $v = a \times b$. Then, one needs the cosine of the angle between them, in the plane defined by having the surface normal parallel to v, which is found by $c = cos(\theta) = a \cdot b$. We then need the skew-symmetric cross-product matrix of v, which is found as:

$$\begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$$

The final rotation matrix is then calculated as:

$$R = I + [V]_x + [v]_x^2 \frac{1}{1+c}$$

Where I is the three-dimensional identity matrix. After finding R, we can then rotate any vector through left-handed matrix multiplication of the vector and this matrix. The way it is defined gives aR=b in all cases of a and b, unless the vectors point in exactly opposite directions, because then the cosine of the angle between them is equal to -1, giving a division by zero error.

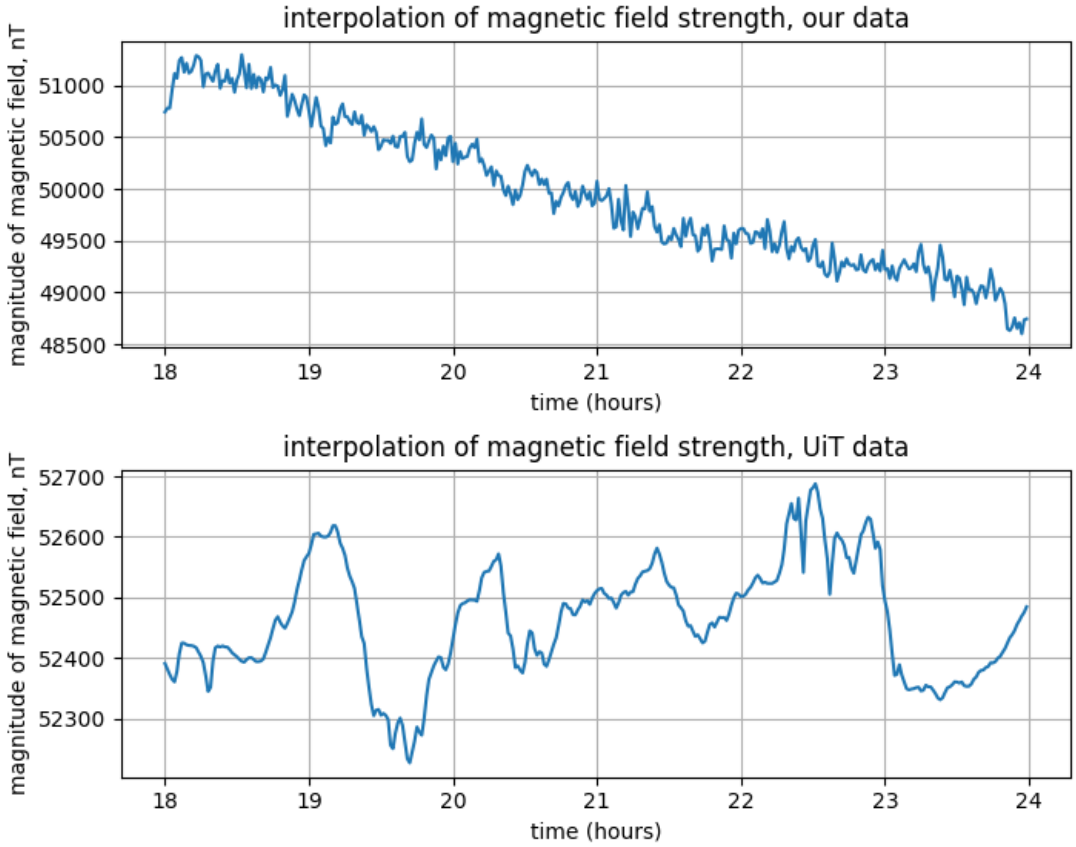
4 Results & Discussion

4.1 Data Collection

We placed the device in a relatively remote area on Tromsøya, and recorded data from may 11th at approximately 02:30 to may 12th 02:10. The data collected was not to a satisfying accuracy, with an interesting time period being may 11th 18:00 to may 12th 00:00. Here, the collected data show a clear negative linear trend, while the UiT magnetometer has significant deviations from mean.

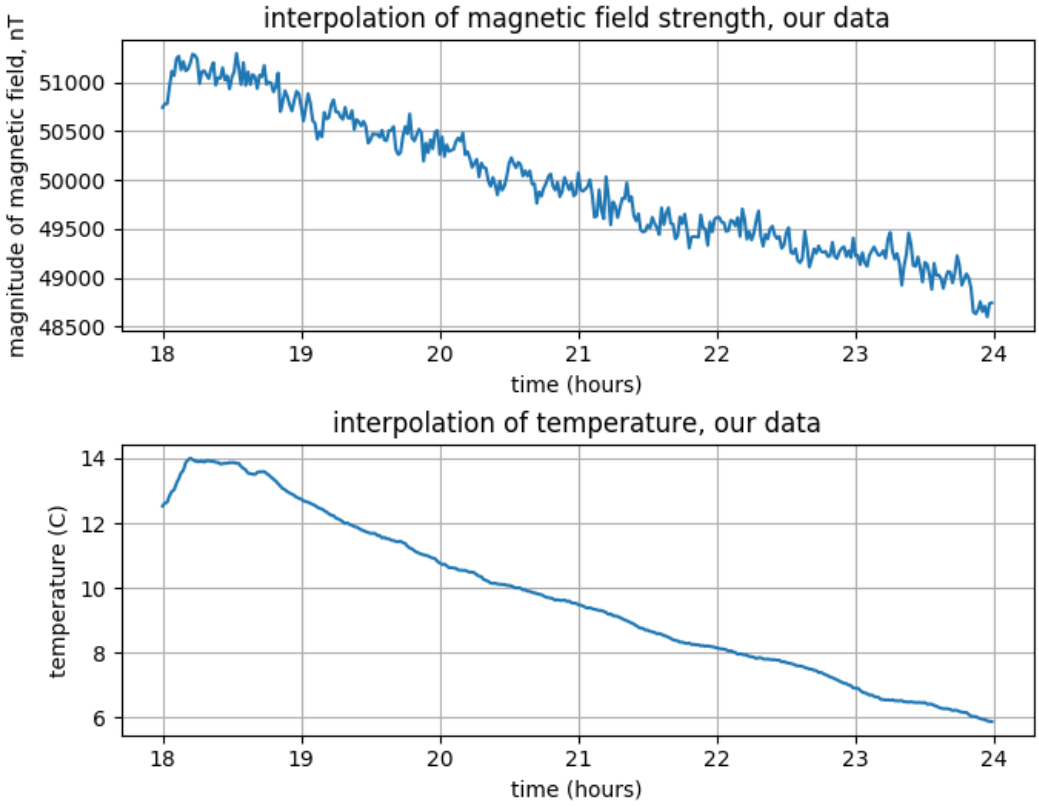
We collected reference data from the website of the Tromsø Geophysical Observatory (TGO) from their website. We used the archived 10-second average data for the Z-axis, and interpolated it in order to obtain datasets with the same time axis. We selected a time period where there were significant variations in the measured magnetic field, although the time period selected is ultimately arbitrary.

4.2 Data Analysis

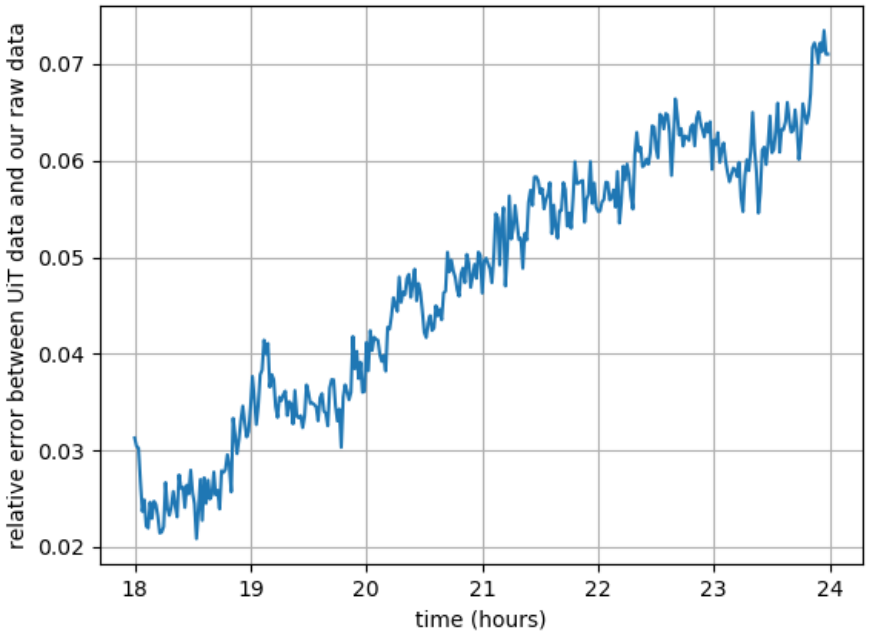


As seen in the figure above, neither the magnitude nor small-scale variations are comparable, and the datasets actually have a small negative correlation. This is probably because the mean value of the UiT data increases with time, while the mean value of our data decreases with time, and not an actually significant finding. The decline in magnitude is thought to be attributed to a dependence on temperature in the sensor. A plot of the magnitude of the magnetic field in the Z-direction and the temperature is shown below. As can be clearly seen, there is a strong correlation between temperature and measured magnetic field (calculated at r = 0.9836, which is a very strong correlation). In an attempt to decorrelate the data, we used the function

$$decorrelated\,data = interpolated\,data + interpolated\,data \cdot (25 - temperature) \cdot 0.0061$$

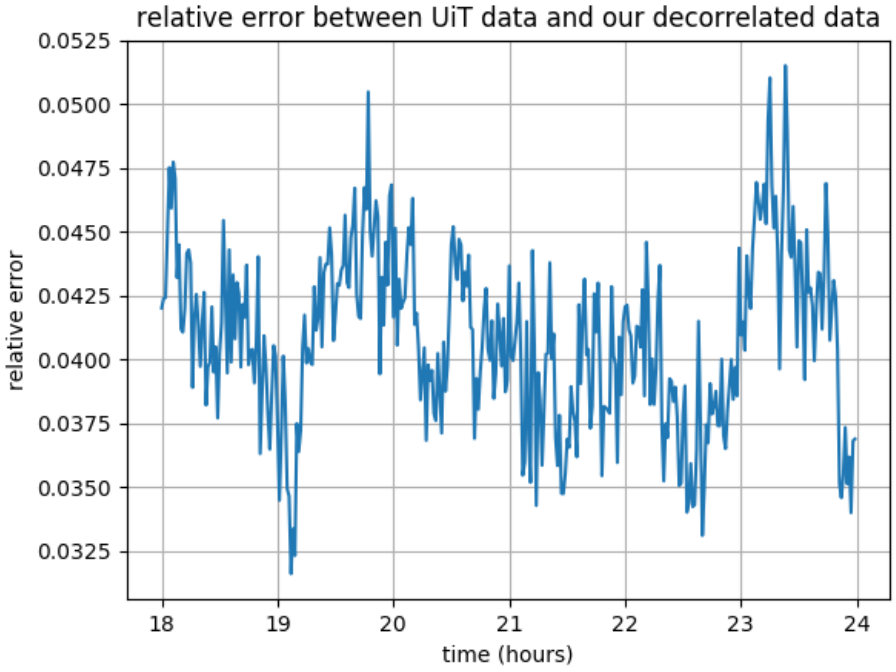


We determined the scalar 0.0061 by starting at 0.1%, as suggested in the documentation of the sensor and increased the scalar slightly until a Pearson R test gave a number close to zero. If we knew the exact relationship between measured magnetic field and temperature, we might obtain more accurate results.



The above figure shows the relative difference of data gathered from our device and the UiT data. As can be clearly seen, the error increases with time, and reaches more than 7% of the total

magnitude. If this error is entirely attributed to temperature, the dependence could be isolated and removed, but since we do not know the exact effects that temperature have on the sensor output, we are unable to say with certainty that we have removed it entirely.



After our attempt at decorrelating the results with temperature, the relative error no longer increased with time. This shows a fairly large constant error as well as random fluctuations around this constant value. This constant error could be caused by currents in our device, or errors in calibrating the sensor itself. The random errors could be due to our poor solution to the temperature dependence, or caused by actual magnetic noise in the area.

The sensor output entirely nonsensical values for the X and Y axis, and we have therefore ignored them. The Z-axis seems unaffected. We are not sure what exactly is disturbing the X and Y axes, because the disturbance is unaffected by whether the sensor is in the box, near the electronics. It seems like an error in the sensor itself.

5 Future Design & Implementations

5.1 Prototype To Production

None of the wires are soldered and have not been known to come loose under duress. As the device is a prototype and the wire diagram may or may not be subject to changes towards the final version we saw no need to secure these further. In the final version a more permanent and "cleaner" solution will be in place. We imagine that making a single circuit board containing all the necessary chips would be more suitable for a large scale production device. This would also result in perfect alignment of the axis between the magnetic sensor and the accelerating sensor.

5.2 Large Spread Network

If the concept prototype were to function, creating a network of devices which will give us a better understanding of the behavior of the devices and data in different geographical locations.

The idea would be for each device to send data to a back-end server. This back end server would do different scrubbing and calculations of data as well as analyze the data for anomalies that isn't picked up on within the devices own scrubbing. With a lot of data-logs, running diagnostics would give us a good amount of information into the behavior of the magnetic field in a larger area, but also how the device readings behave according to their geographical location. It would

be easier to detect disturbances and anomalies. If a device sends suspect data over a longer period of time the back end server could communicate to the device to run reset itself, to see if the device just needed a small fix, before going on site for manual support.

A large network would require a good and stable connection between nodes in the network and the signal host. In this field it would require signal in more remote places than the middle of Tromsøya, which could become an issue.

5.3 Front End

For this project data is handled with use of Telenors IoT MIC API (for viewing purposes). The data should, in the future, be viewed from a more suited front end. The Telenor MIC API enables us to use web-hooks, which can easily pass on the data to a separate front-end, where we can scrub the data before it is displayed. While Telenor MIC is suitable for this project, the future project requires different analysis tools, which cannot be found within the Mic platform. We have written these ourselves as an "aftermath" piece for the use in this project. On the future front-end, users would have access to different types of analytics and diagnostic tools. It would be possible to display log data for different scenarios and time frame (Not unlike the Mic). We want to enable for example graphical 3D modeling of the magnetic field. This front-end would work as an extension of the Telenor MIC, allowing for a wider use of the data.

5.4 GPS Usage

For future use, making use of a GPS unit could prove to be valuable. It has been on the backlog for us throughout the development of this project, due to the importance of achieving and proving the actual magnetic data to be accurate. A GPS unit would serve as both a way to timestamp the data, which would be valuable due to the unstable connectivity of the LoRa-network causing time delays on the data. With time-stamping packages which have trouble sending could be saved on the PyCom LoPy using a SD card, and retrieved at a later point, for a full set of data.
The GPS would also prove vital in a large scale use of the device. This project aims to prove that this device can be used for 3D modeling of the geomagnetic field, with the use of many small units, but knowing the location for each of the devices should not be done manually.
A GPS coordinate signal that would be sent initially, and repeated at slow intervals to map out the network of devices. It is also extremely important for the future 3D modeling that the position the devices are measuring at is accurate.

5.5 Bad Data & Data Scrubbing

The project is based around a high accuracy of magnetic data. This means that there should be very small variations between nearby data nodes. Collecting data points as close together in time as possible is important to ensure that we do not miss important variations. Averaging several measurements allows us to reduce random measurement uncertainty. There must therefore be a balance between frequency of data transmission and number of data points averaged together.
We must also ensure that what we are measuring is in fact the geomagnetic field. This can be challenging due to environmental noise, but also due to other dependencies of the sensor. If we can find a function that accurately describes the relationship between the temperature and output of the sensor, we would be able to remove this dependency, which would result in more accurate measurements.

5.5.1 Discarding Bad Data

One future method of discovering bad data can be to make use of the system which takes a set of measurement and averages them. If any data within the set deviate too much from the others. As this is a high accuracy magnetometer, measuring very subtle changes over time, a rapid change within the set would mean that the magnetometer has picked up some disturbance, and should discard the data.

It might also be needed to write an algorithm data detection to ensure what type of data is being processed. Is it an extraordinary occurrence in the sensor data or just bad readings? Defining and understanding what type of data to keep or discard will also be a subject of consideration, since the trade of being limited bandwidth and resources.

5.5.2 Temperature Data

As of right now the temperature data gets inaccurate due to the sensor being inside the encasing. The box acts as a thermal insulator and greenhouse, amplifying the effects of solar heating while shielding the box from rapid temperature changes. Due to the dependence the sensor output has on temperature, the greenhouse effect severely affects the measured magnitude during the day. We have no reason to believe that the relation between measured magnitude and temperature is different for each axis. This in-case sensor has actually helped us understand and measure the correlation between sensor-temperature and data error, and in the future making a less green-housy encasing could result in less error of data. The temperature from inside the casing, should not be read as the "accurate outdoor temperature".

6 Conclusion

6.1 Hardships

6.1.1 Isolated Environments of Measuring Data

In order to detect changes in the geomagnetic field, other sources of magnetic fields must be either known or small in comparison to changes caused by geomagnetism. Since currents cause magnetic fields, we cannot place our device close to areas where we can expect relatively large or highly varying currents. Metallic objects that move quickly, such as cars, can also induce a magnetic field. We must therefore place our device in a relatively remote area in order to minimize this noise. Due to the high sensitivity, we still expect to see some disturbances, but we assume them to be sufficiently small.
A possible solution to the temperature dependency is to place the device in an isolated container filled with ice water. This will maintain a temperature of zero degrees Celsius for as long as there is still solid ice in the container, providing a very stable environment. This could then be used to compare RMS deviations from mean to the UiT magnetometer data. If the data then shows satisfactory accuracy, work can be done to quantify the temperature dependence and remove it from the dataset.

6.1.2 Accuracy

The sensor we used did not provide measurement with a high enough degree of accuracy. The data had a very strong dependence on temperature, and we are unaware of the exact relationship between these quantities, which casts doubt on the measurements. Further, the data did not match the UiT reference data, which we assume to be caused by a linear offset due to currents in the box and random errors caused both by the uncertain temperature relation and sensor inaccuracy. In order to obtain a usable dataset, accuracy must be improved significantly.

6.1.3 Rotation of the Magnetic Data Axis

One issue is that rotating our vector in the plane perpendicular to the direction of gravity is not necessarily easy using this approach, and further rotation might become necessary.

A more important issue was due to the erroneous data in the X and Y axis of the magnetic data. The error rendered us unable to use the rotation. The errors in the X and Y axis of the magnetic data caused any rotation to result in obscure result data. We were able to obtain useful data despite this by manually placing the device on very flat ground (We still used accelerometer data to ensure that the device did not accidentally un-flat itself throughout our readings).

6.1.4 Libraries & Documentation

One of the hardships with this project was to find documentation or libraries for the sensors we implemented. The micro sensors and micro controllers are not yet used by the public consumers to a capacity where there are a lot of documentation or open source libraries for different languages or micro-controllers. The MPU9265 sensor for instance did not have its own data/register-sheet, but used the MPU9250 data sheet and register sheet to write software libraries for it, since these sensors were in the same 92XX series.

6.2 Findings

The device, at the current time, is not accurate enough to model currents in the ionosphere. Sensor accuracy is the largest challenge, and we have shown that the accuracy of the sensor we used was insufficient to model currents in the ionosphere with a reasonable accuracy. If we used a more sensitive sensor, with a known dependence on temperature, and a more strictly controlled or noiseless environment, we might have been able to obtain precise enough measurements. Use of accurate timestamps would also be necessary, but is a secondary concern.

The absolute difference of our measurements and the UiT measurements are too large to be productively used to measure currents in the ionosphere, and significant increases in measurement accuracy must be made before our device is useful for scientific purposes.

To better these measurements, expensive magnetic sensors could be implemented for better accuracy. For example for 2,100 U.S dollars we get a Fluxgate magnetic field sensor, which are known for great accuracy. This sensor would make it possible to model the auroral electro-jet and give precision measurements. The device prototype we have produced in this project has a total cost of 150 U.S dollars, where the two magnetic project sensors, LIS3MD & MAG3110, cost between 10-30 U.S dollars. Seeing as the government spent about 36 million U.S. dollars on the Eiscat 3D project, 2,100 U.S dollars for an IoT device which would provide valuable data and insight for researchers, students and the Eiscat 3D project seams reasonable in cost versus benefit. A suggested price-ceiling from a professor at the faculty for physics and technology was 1000 Norwegian crowns, which currently eliminates Fluxgate technology due to price considerations.

7 Appendix

7.1 Development Process Summary

7.1.1 Milestone 1 - Project Concept

Planning & Development Defining team member roles:

- **Torbjørn Tveito**
  Physics student. Calculations and data-analytics and provide knowledge and insight of ionospheric effects, statistical signal analysis, and general magnetic theory.
- **Anton Garri Fagerbakk**
  Comp.Sci student. Development and configuration of software.
- **Jon H.L. Johansen**
  Comp.Sci student. Development and configuration of software.

Results/Evaluation In the first milestone we lay out the project concept. We map and discuss foreseeable challenges to overcome, limits of the project and choose which sensors to use. We brushed up on some general knowledge about magnetic fields.

7.1.2 Milestone 2 - Design

Planning & Development In milestone two we decide and overlaying software design. This design should make it easy to divide assignments more separate and write and test featuring functions without disturbing the others works. Decisions made regarding what is to be calculated where, and why. We also map out key risks for our implementation and produced a time-line and a scope for the future of the project.
Attempt to connect to MIC platform to gain some knowledge, and start focusing work on specific sensors.

Results/Evaluation

- Find and order suitable magnetic sensors, check that these are fine with the TA's. **- Torbjørn**
- Produced an overlaying design for code structure **- Anton, Jon**
- Successfully utilized the MIC API + Startiot library for the first time **- Anton**
- MAG3110 sensor library complete**- Jon**

7.1.3 Milestone 3 - Prototype/Alpha

Planning & Development In milestone three we have sensor software alpha version, but not fully implemented hardware design. Produce more sensor libraries. Try to calibrate and mitigate data for correct data output.
Developing software or mitigating bad data. Doing calibration and troubleshooting sensor problems.

Results/Evaluation

- MPU9265 library complete and functional **- Anton**
- Rotation "library" complete **- Torbjørn**
- Combining accelerometer data with Mag data **- Anton, Torbjørn**
- Planning and assembling protoype v1 "box" **- Jon**

7.1.4 Milestone 4 - Beta

Planning & Development Finish up a working box ready for testing data. Develop data representation on Telenors IoT MIC site, so we can analyze incoming data through LoRa network, and prototype-test the device. Find errors and issues and resolve them to the extent of our ability. Compile earlier notes and write a resulting report on our findings.

Results/Evaluation

- Develop functional Telenor MIC "view" and set up additional parameters for use of the Telenor MIC Analyze export tool **- Jon**
- Wrote scrubbing software and full data analysis of our collected data **- Torbjørn**
- Develop small library for testing the LIS3MDL sensor vs MAG3110. **- Jon**
- Refactoring code (Physicists do not have pretty code.) and compiling general report **- Anton**
- Report all findings, experiences and learnings and conclude. **- All**

References

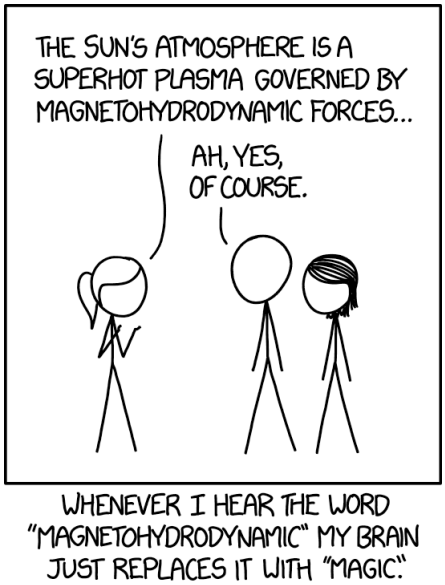
`https://www.nxp.com/docs/en/application-note/AN10216.pdf` [1] - Information about I2C bus/interface

`https://www.invensense.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf` - Product Specification for MPU9250

`https://www.invensense.com/wp-content/uploads/2015/02/RM-MPU-9250A-00-v1.6.pdf` - Register Maps and Descriptions for MPU9250

`https://www.nxp.com/docs/en/data-sheet/MAG3110.pdf` - Data-sheet for MAG3110

xkcd https://xkcd.com/1851/



`https://xkcd.com/1942/`