T.C.
USKUDAR UNIVERSITY
ENSTITUTE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

MASTER'S DEGREE PROGRAM OF COMPUTER ENGINEERING

MASTER'S DEGREE THESIS

IMPROVING IMU YAW CALCULATION BASED ON MAGNETOMETER READINGS AND NEURAL NETWORK

Mohammad Tabish QAYAM

Thesis Advisor
Ph. D. Ihab ELAFF

ISTANBUL-2023

T.C.
USKUDAR UNIVERSITY
ENSTITUTE OF NATURAL SCIENCES

DEPARTMENT OF COMPUTER ENGINEERING

MASTER'S DEGREE PROGRAM OF COMPUTER ENGINEERING

MASTER'S DEGREE THESIS

IMPROVING IMU YAW CALCULATION BASED ON MAGNETOMETER READINGS AND NEURAL NETWORK

Mohammad Tabish QAYAM

Thesis Advisor

Ph. D. Ihab ELAFF

ISTANBUL-2023

ABSTRACT

Improving IMU Yaw Calculation Based on Magnetometer Reading Neural Network

The accurate determination of a mobile robot's orientation is critical for many robotic applications, such as navigation, mapping, and localization. Inertial measurement units (IMUs) are commonly used to estimate the orientation of a robot. However, IMUs suffer from drift and noise, which can cause errors in the estimation of yaw. In this study, we propose a method for improving the accuracy of the IMU yaw calculation by incorporating magnetometer readings and a neural network. The neural network is trained to learn the relationship between the magnetometer readings and the yaw angle, and the network is used to correct the IMU yaw estimation. The proposed method is evaluated using experimental data, and the results demonstrate that the proposed method significantly improves the accuracy of the IMU yaw calculation. Moreover, the method is computationally efficient and can be easily integrated into existing robotic systems. The proposed method has the potential to improve the performance of many robotic applications that rely on accurate estimation of a robot's orientation. The following section provides a succinct outline of the extensive corpus of research on robot orientation.

Keywords: A, Inertial measurement unit (IMU). B, Yaw calculation. C, Position D, Movement. E, Measurement. F, Neural Network. G, Magnetometer.

THANKS TO

As I reach the end of this incredible journey, I am filled with gratitude for all of the guidance, support, and encouragement that you have provided throughout my academic journey. Your unwavering commitment to my success has been an inspiration to me, and I am truly grateful for all that you have done.

Dr. Ihab, your support, and mentorship throughout my thesis have been invaluable. Your insights, advice, and constructive criticism have challenged me to grow as a researcher and a person. I am deeply grateful for your patience, dedication, and willingness to go above and beyond to help me achieve my goals.

To all of my professors, your passion for teaching and commitment to student's growth and development has been truly remarkable. Your classes have been a constant source of inspiration, and I am grateful for the opportunities to learn from you.

I would also like to thank you for your patience, kindness, and understanding during the difficult moments of my academic journey. Your support has given me the strength and motivation to persevere and push myself to new heights.

Finally, I would like to express my gratitude for the enduring friendship and camaraderie that we have built throughout our time together. Your unwavering support and encouragement have helped me to stay motivated and focused, even during the most challenging times.

Thank you once again for everything that you have done for me. I am truly blessed to have had you as my advisors and teachers, and I will carry the lessons and experiences that you have given me for the rest of my life.

FORM OF DECLARATION

Herewith I declare, that I obtained all the information and documents in this study within the framework of academic rules, presented all visual, auditory, and written information and results in accordance with scientific ethics, did not falsify the data I used, referred to the sources I used in accordance with scientific norms, that my thesis was original except in the cases cited, produced by me and written in accordance with the Thesis Writing Guide of Uskudar University Institute of Computer Sciences.

Date
Mohammd Tabish Qayam
Signature

CONTENTS

INDEX OF TABLES

INDEX OF FIGURES

Page

INDEX OF IMAGERY AND ABBREVIATIONS

IMU : Inertial Measurement Unit

NN : Neural Network

1. INTRODUCTİON TO ROBOT LOCALİZATİON

1.1 Introduction

Robot localization refers to the process of determining the position and orientation of a mobile robot relative to its environment. Localization is a crucial aspect of robotics and is necessary for tasks such as navigation, mapping, and exploration. There are various techniques for robot localization, including landmark-based approaches, feature extraction, and odometry. In recent years, advances in sensor technology and machine learning have led to the development of new methods for robot localization that are more accurate and robust.

Recent advances in sensor technology, particularly in the development of low-cost and miniaturized sensors, have led to the widespread use of IMUs for robot localization. IMUs consist of accelerometers and gyroscopes that measure the linear and angular acceleration of the robot, respectively. The data from the IMU can be integrated over time to estimate the robot's position and orientation. However, IMUs suffer from drift and noise, which can lead to errors in the estimation of the robot's position and orientation.

To address these issues, researchers have proposed various methods for improving the accuracy of IMU-based localization, such as incorporating magnetometer readings and neural networks. These techniques aim to reduce the effects of drift and noise and provide a more accurate estimation of the robot's position and orientation.

In summary, robot localization is a critical aspect of robotics, and various techniques are used to estimate the robot's position and orientation accurately. Advances in sensor technology and machine learning have led to the development of new methods that are more accurate and robust than traditional techniques.

The remainder of this thesis is organized as follows. Chapter 1 provides Introduction a review of the relevant problem statement on IMU estimation and Euler angels , Compas Magnetometer, robot localization, filters, . Chapter 2 describes Letrature revwie, Problem statement, Related work, Magnetometer Principle , Orientation Source of robot, Gyroscope, Accelerometer Error, Orientation Reduction Technic. Chapter 3 presents the System Design, System Diagram, Data Acquisition and Neural Network Training. Chapter 4 experimental setup and results of the proposed diagram, Data aucuistion, traning stage, testing stag and final result. Chapter 5 provides a discussion and conclusions. Finally, Chapter 6 outlines the resouce of this research.

1.2 Robot Localization

Robot localization is the process of determining the position and orientation of a mobile robot relative to its environment. Accurate localization is essential for robotic applications such as navigation, mapping, and exploration.

There are various techniques for robot localization, including landmark-based approaches, feature extraction, odometry, and sensor fusion. Each technique has its own advantages and disadvantages, and the choice of technique depends on the specific application.

Landmark-based approaches involve the use of sensors such as cameras or laser range finders to detect and track known features in the environment, such as walls or corners. By triangulating the position of these features, the robot's position and orientation can be estimated. However, landmark-based approaches are computationally expensive and can be sensitive to changes in the environment, such as the addition or removal of objects.

Feature extraction is a technique that involves detecting and extracting features from the environment, such as edges, corners, or textures. These features are then matched to a map of the environment, allowing the robot's position and orientation to be estimated. Feature extraction is less computationally expensive than landmark-based approaches, but it can still be sensitive to changes in the environment. Thrun, S., Burgard, W., & Fox, D. (2005).

Odometry is a technique that relies on wheel encoders to estimate the robot's position and orientation. The method assumes that the robot's wheels move without slipping or sliding, and the distances traveled by each wheel can be used to estimate the robot's motion. However, odometry suffers from cumulative errors due to wheel slippage, uneven surfaces, and other factors. As a result, odometry is often used in combination with other techniques, such as inertial measurement units (IMUs) and GPS, to improve the accuracy of the localization.

IMUs consist of accelerometers and gyroscopes that measure the linear and angular acceleration of the robot, respectively. The data from the IMU can be integrated over time to estimate the robot's position and orientation. However, IMUs suffer from drift and noise, which can lead to errors in the estimation of the robot's position and orientation.

Sensor fusion is a technique that combines data from multiple sensors, such as cameras, IMUs, and GPS, to estimate the robot's position and orientation. By fusing data from multiple sources, the accuracy of the localization can be improved. However, sensor fusion can be computationally expensive and requires careful calibration of the sensors.

Robot localization is a critical aspect of robotics, and it involves a wide range of techniques and technologies. The goal of robot localization is to accurately determine the position and orientation of a mobile robot relative to its environment. The process of robot localization typically involves four main steps: sensor data acquisition, sensor data processing, sensor data fusion, and estimation.

The first step in robot localization is sensor data acquisition. This involves collecting data from sensors such as cameras, laser range finders, wheel encoders, IMUs, and GPS. The data collected from these sensors can be in the form of images, point clouds, inertial measurements, or location estimates.

The second step in robot localization is sensor data processing. This involves filtering, smoothing, and preprocessing the sensor data to remove noise, correct for sensor biases, and extract features that are relevant for localization. Techniques such as Kalman filtering, particle filtering, and graph optimization are commonly used for sensor data processing.

The third step in robot localization is sensor data fusion. This involves combining data from multiple sensors to obtain a more accurate estimate of the robot's position and orientation. Sensor fusion techniques can be classified into two main categories: measurement fusion and state fusion. Measurement fusion combines data from multiple sensors to obtain a more accurate estimate of a specific variable, such as position or orientation. State fusion combines data from multiple sensors to estimate the entire state of the robot, including its position, orientation, and velocity.

The fourth step in robot localization is estimation. This involves using the processed and fused sensor data to estimate the robot's position and orientation. Estimation techniques can be classified into two main categories: deterministic and probabilistic. Deterministic techniques, such as least-squares estimation and optimization, provide a single estimate of the robot's position and orientation. Probabilistic techniques, such as Kalman filtering and particle filtering, provide a probability distribution over the possible positions and orientations of the robot.

In summary, robot localization is a complex process that involves combining data from multiple sensors and techniques to estimate the robot's position and orientation accurately. Advances in sensor technology and machine learning have led to the development of new methods that are more accurate and robust than traditional techniques. Dellaert, F., & Fox, D. (2008).

1.3 Filters

Filters are essential tools for processing and analyzing data in a wide range of applications, including signal processing, image processing, and sensor data fusion. Filters are used to remove noise, smooth data, extract features, and separate signal components. In this section, we will discuss some of the most commonly used filters, including low-pass filters, high-pass filters, band-pass filters, and Kalman filters.

Low-pass filters are filters that allow low-frequency components of a signal to pass through while attenuating or blocking high-frequency components. Low-pass filters are commonly used to remove high-frequency noise from signals, to smooth data, or to extract slow-changing features from a signal. One of the most commonly used low-pass filters is the moving average filter, which replaces each sample in the signal with the average of the neighboring samples.

High-pass filters are filters that allow high-frequency components of a signal to pass through while attenuating or blocking low-frequency components. High-pass filters are commonly used to remove low-frequency noise from signals or to extract high-frequency features from a signal. One of the most commonly used high-pass filters is the derivative filter, which calculates the rate of change of the signal between two samples. Proakis, J. G., & Manolakis, D. G. (2006).

Band-pass filters are filters that allow a specific frequency band of a signal to pass through while attenuating or blocking other frequencies. Band-pass filters are commonly used to extract specific frequency components from a signal or to remove unwanted frequencies from a signal. One of the most commonly used band-pass filters is the Butterworth filter, which provides a maximally flat frequency response in the passband.

Kalman filters are a type of probabilistic filter that is commonly used for sensor data fusion and state estimation. Kalman filters are based on a mathematical model that describes the evolution of a system over time, and they use measurements from sensors to estimate the state of the system. Kalman filters can handle noisy measurements, account for uncertainty in the system model, and provide estimates of the state of the system that are optimal in a least-squares sense. Smith, S. W. (2002).

In summary, filters are essential tools for processing and analyzing data in a wide range of applications. Low-pass filters, high-pass filters, band-pass filters, and Kalman filters are some of the most commonly used filters, each with its own specific advantages and disadvantages. The choice of filter depends on the specific application and the properties of the data being analyzed. Advances in signal processing, machine learning, and sensor technology are continuing to drive innovation and improvements in filter design and performance. Doucet, A., de Freitas, N., & Gordon, N. (2001)

1.3.1 Extended Kalman Filter (EKF)

The Extended Kalman Filter (EKF) is a nonlinear extension of the original Kalman filter, which was designed for linear systems. The EKF is a popular variant of the Kalman filter used for state estimation in nonlinear systems. In nonlinear systems, the state transition and observation models are nonlinear functions of the state variables, which makes it impossible to use the original Kalman filter directly.

The EKF addresses this problem by linearizing the nonlinear models around the current estimate of the state variables. The linearized models are then used to update the state estimate and the covariance matrix using the standard Kalman filter equations. The EKF has two main steps: prediction and update. Haykin, S. (2001).

In the prediction step, the EKF uses the state transition model to predict the next state of the system given the current state estimate and the control input. The prediction also includes an estimate of the covariance matrix, which represents the uncertainty in the prediction. Särkkä, S. (2013).

In the update step, the EKF uses the observation model to update the state estimate and the covariance matrix based on the measurements obtained from sensors. The update step involves the computation of the Kalman gain, which determines the weight given to the prediction and the measurement in the final estimate. The EKF has several advantages over other nonlinear filtering techniques. Firstly, it is computationally efficient and can be implemented in real-time applications. Secondly, it provides estimates that are optimal in a least-squares sense, which means that they minimize the expected squared error between the estimated state and the true state. Lastly, it provides estimates of the uncertainty in the state estimate, which is important in many applications such as robotics and autonomous systems. Kalman, R. E. (1960).

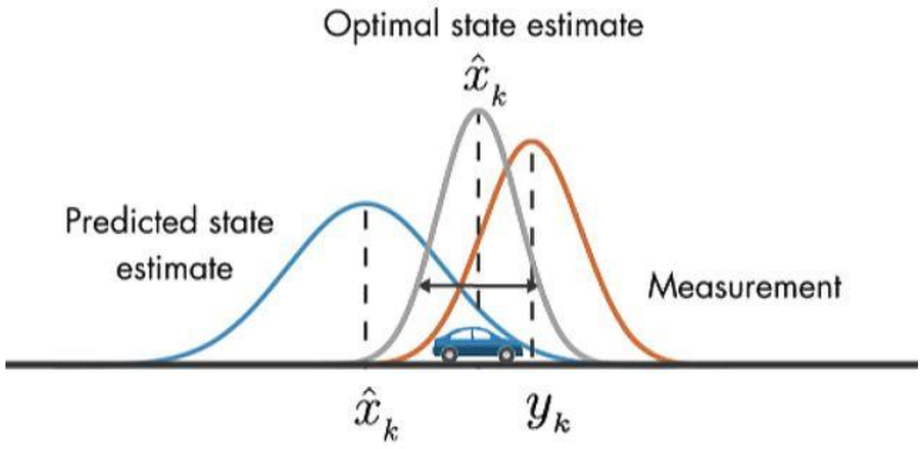


Figure 1: Kalman Filters

The EKF has some limitations, such as sensitivity to the initial state estimate and the linearization error. The linearization error can be reduced by using higher-order Taylor series expansions or by using other nonlinear filtering techniques such as the Unscented Kalman Filter (UKF) or the Particle Filter (PF). The UKF uses a set of sample points to propagate the covariance matrix and the state estimate through the nonlinear functions, which avoids the linearization error. The PF uses a set of particles to represent the state estimate, which is updated using importance sampling and resampling steps. The PF is more flexible than the EKF and the UKF but is computationally more expensive. Simon, D. (2006).

In summary, the EKF is a powerful tool for state estimation in nonlinear systems. It linearizes the nonlinear models around the current estimate of the state variables using the first-order Taylor series expansion and uses the standard Kalman filter equations to update the state estimate and the covariance matrix. The EKF has some limitations but is computationally efficient and provides optimal estimates and estimates of the uncertainty in the state estimate, making it a popular choice in many applications such as robotics, navigation, and control systems. Simon, D. (2006).

1.3.2 Practical Filters

Practical filters are a type of signal processing technique that is used to modify or extract specific information from signals. The purpose of practical filters is to remove unwanted noise or frequencies from signals, or to smooth out signals, so that they can be more easily analyzed or used for further processing. They are widely used in various applications such as audio processing, image processing, data analysis, and control systems. Lyons, R. G. (2011).

Practical filters can be either analog or digital. Analog filters operate on continuous signals and are implemented using analog circuits such as operational amplifiers, resistors, and capacitors. Digital filters, on the other hand, operate on discrete signals that are represented as digital samples, and are implemented using software algorithms or digital signal processing (DSP) hardware. Parks, T. W., & Burrus, C. S. (1987)

There are many different types of practical filters, each with its own specific characteristics and applications. Some of the most common types of filters include:

Low-pass filters: These filters allow low-frequency signals to pass through while attenuating high-frequency signals.

High-pass filters: These filters allow high-frequency signals to pass through while attenuating low-frequency signals.

Band-pass filters: These filters allow a specific range of frequencies to pass through while attenuating frequencies outside that range.

Notch filters: These filters attenuate a specific frequency or range of frequencies while allowing other frequencies to pass through.

Moving average filters: These filters calculate the average of a set of data points over a sliding window, which effectively smooths out the signal.

Butterworth filters: These filters are designed to have a flat frequency response in the passband and a rapid roll-off in the stopband.

Kalman filters: These filters are used for state estimation and are commonly used in control systems and navigation applications.

Wavelet transforms: These filters analyze signals in both the time and frequency domains and are used for signal compression and denoising.

Median filters: These filters replace each data point with the median value of its neighboring points, which effectively removes outliers and noise.

The choice of filter depends on the specific application and the requirements of the system. The design of practical filters involves a trade-off between signal distortion and noise reduction, and different filters may be more suitable for certain applications. The effectiveness of a practical filter can be evaluated based on various metrics such as signal-to-noise ratio (SNR), root-mean-square error (RMSE), and frequency response. Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002).

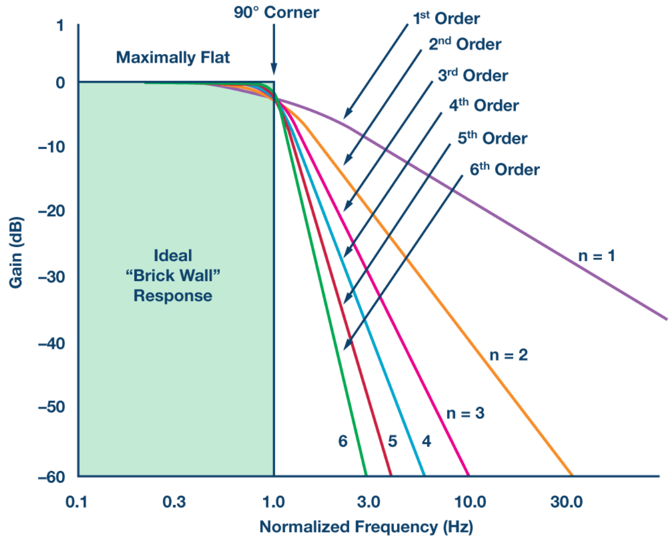


Figure 2: Practical Filters

1.3.3 Complementary filters

A complementary filter is a type of signal processing technique that combines two or more signals to produce an output that is better than the individual signals alone. The complementary filter is commonly used in control systems and navigation applications to estimate the state of a system by combining measurements from multiple sensors.

The complementary filter is based on the concept of low-pass and high-pass filters. The low-pass filter allows low-frequency signals to pass through, while attenuating high-frequency signals. The high-pass filter allows high-frequency signals to pass through, while attenuating low-frequency signals. The complementary filter combines the outputs of these two filters to produce an output that has both the low-frequency components from the low-pass filter and the high-frequency components from the high-pass filter. La Hera, P. M. G., Gutiérrez-Galan, D., & Fuentes, O. (2017).

In practice, the complementary filter is implemented using a weighted average of the low-pass and high-pass filtered signals. The weights of the low-pass and high-pass filters are chosen such that they add up to one, and the cutoff frequencies of the filters are chosen based on the characteristics of the system and the sensors being used.

The complementary filter is often used in applications where accurate and reliable measurements are required, such as in drone and robot navigation, or in control systems for aerospace and automotive industries. The complementary filter is computationally efficient and can be implemented in real-time applications, making it a popular choice for many practical applications. Lua, K. T., & Yahya, W. J. (2015).

One of the main advantages of the complementary filter is its ability to provide accurate estimates of the state of a system even in the presence of noise and sensor errors. The complementary filter is also robust to changes in the system dynamics, which makes it suitable for applications where the system may be subject to disturbances or external forces.

Overall, the complementary filter is a useful and practical tool for signal processing and state estimation in a wide range of applications, particularly in control systems and navigation. Its ability to combine multiple signals to produce a more accurate and reliable output makes it a valuable tool for engineers and researchers working in various fields of science and engineering. Mahony, R., Hamel, T., & Pflimlin, J. M. (2008).

1.4 Euler Angles

Euler angles, named after the Swiss mathematician Leonhard Euler, are a set of three angles used to describe the orientation of a rigid body in three-dimensional space with respect to a fixed coordinate system. Euler angles are widely used in many fields, including robotics, aerospace, and computer graphics.

The three Euler angles are usually denoted as roll (φ), pitch (θ), and yaw (ψ), also known as the Tait-Bryan angles. They represent a sequence of three rotations that transform a body-fixed coordinate system to a fixed reference coordinate system. The sequence of rotations can be either intrinsic (i.e., rotating the body-fixed coordinate system) or extrinsic (i.e., rotating the reference coordinate system). There are several conventions for defining the Euler angles, depending on the order of rotations and the axes of rotation used. The most commonly used convention is the ZYX convention, also known as the aerospace convention, which corresponds to a sequence of rotations about the Z-axis (yaw), Y-axis (pitch), and X-axis (roll). To understand the concept of Euler angles, it is helpful to consider a simple example of a rigid body rotating about a fixed point in three-dimensional space. The orientation of the body can be described by the three angles: roll, pitch, and yaw.

Roll (φ) is the rotation of the body about its X-axis. This rotation changes the orientation of the Y-axis and Z-axis relative to the fixed coordinate system. Pitch (θ) is the rotation of the body about its Y-axis. This rotation changes the orientation of the X-axis and Z-axis relative to the fixed coordinate system. Yaw (ψ) is the rotation of the body about its Z-axis. This rotation changes the orientation of the X-axis and Y-axis relative to the fixed coordinate system. Euler, L. (1768).

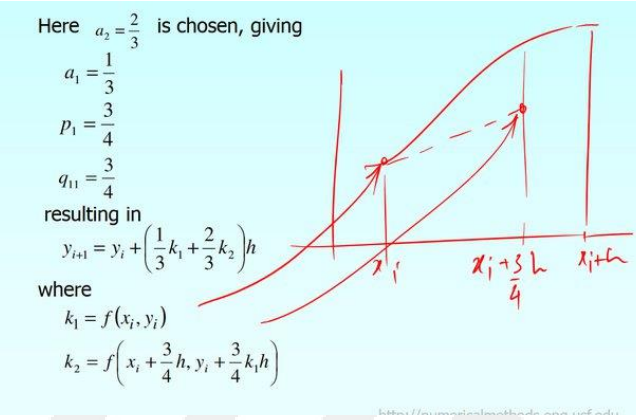


Figure 3: Euler Angels

The sequence of rotations is important because the resulting orientation of the body depends on the order in which the rotations are performed. For example, if the body is first rotated about the X-axis, then about the Y-axis, and then about the Z-axis, the resulting orientation will be different from the orientation obtained by rotating first about the Z-axis, then about the Y-axis, and finally about the X-axis. The Euler angles have several advantages over other methods of describing the orientation of a rigid body. They are intuitive and easy to understand, and they provide a compact representation of the orientation using only three numbers. They are also widely used in many applications, including robotics, aerospace, and computer graphics. Shuster, M. D. (1993).

However, the use of Euler angles also has some disadvantages. One of the main issues is the problem of gimbal lock, which occurs when the pitch angle approaches ±90 degrees. In this case, the roll and yaw angles become coupled, and the rotation matrix becomes degenerate. This can lead to numerical instability and inaccuracies in the orientation estimation.

To overcome this problem, various alternative representations of the orientation have been proposed, such as quaternions and rotation matrices. These representations have different advantages and disadvantages and are used in different applications depending on the specific requirements of the problem. Goldstein, H. (1980).

In summary, Euler angles are a set of three angles used to describe the orientation of a rigid body in three-dimensional space with respect to a fixed coordinate system. They are intuitive and easy to understand, but they can suffer from gimbal lock and other issues. Understanding the concept of Euler angles is important in many fields, including robotics, aerospace, and computer graphics.

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots = \sum_{n=0}^{\infty} (-1)^n \frac{x^{2n}}{(2n)!}$$

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \dots = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

Figure 4: Euler Formula

1.5 Compass Magnetometer

The compass magnetometer is a type of magnetic sensor used to measure the strength and direction of magnetic fields. It is often used in applications such as navigation systems, geophysical exploration, and in measuring the magnetic properties of materials. The Earth's magnetic field is one of the most well-known and widely studied magnetic fields. The magnetic field of the Earth is generated by the motion of molten iron in the Earth's outer core.

The Earth's magnetic field has both a strength and a direction. The strength of the magnetic field is measured in units of tesla (T) or gauss (G). One tesla is equivalent to 10,000 gauss. The Earth's magnetic field has an average strength of about 0.5 gauss or 50 microtesla. However, the strength of the magnetic field varies depending on location and can range from 0.25 gauss to 0.65 gauss. J. J. Clark and J. W. F. Peirson,(2007)

The direction of the Earth's magnetic field is described by the magnetic North and South poles. The magnetic North pole is located in the Canadian Arctic and the magnetic South pole is located in Antarctica. The direction of the magnetic field at any point on the Earth's surface is determined by the angle between the magnetic field lines and the surface of the Earth. This angle is called the inclination or dip angle.

The COMPASS magnetometer is designed to measure the strength and direction of magnetic fields. It uses a sensor made of a material that is sensitive to magnetic fields, such as a magnetoresistive material or a Hall effect sensor.

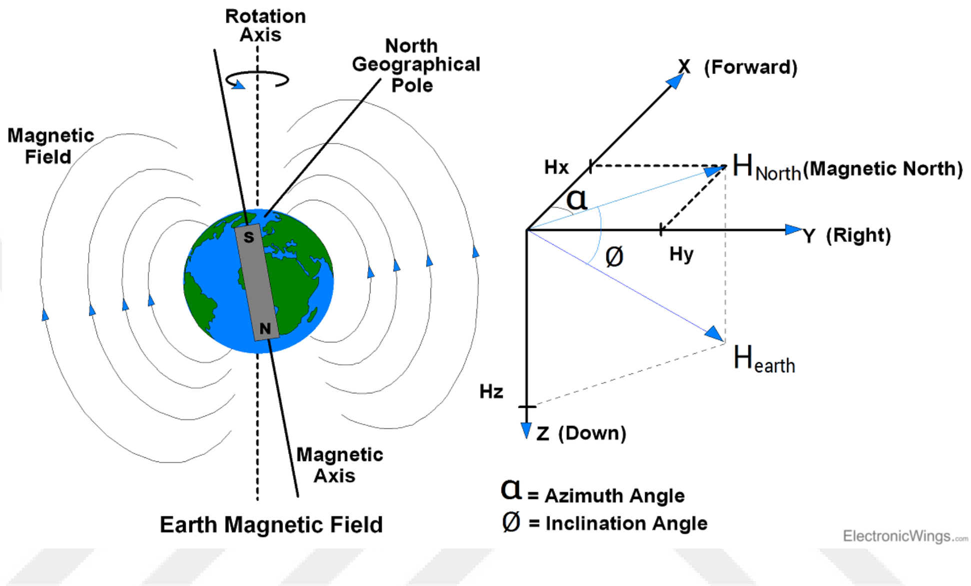
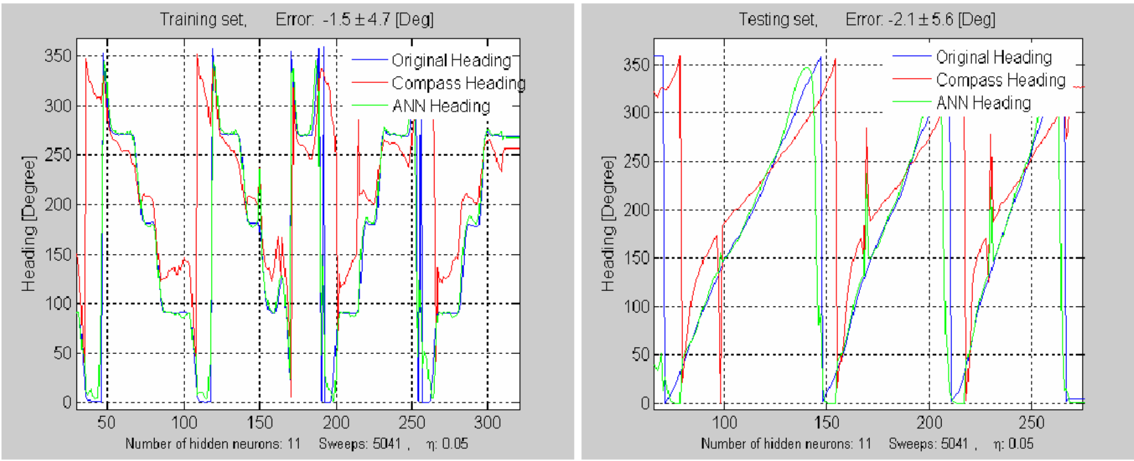


Figure 5: compass magnetometer

When a magnetic field is applied to the sensor, it produces an electrical signal that can be measured and used to determine the strength and direction of the magnetic field.

The Earth's magnetic field is generated by the motion of molten iron in the Earth's outer core. This motion creates a self-sustaining loop of electric current, which in turn generates a magnetic field. The magnetic field of the Earth is therefore a dipole field, which means it has a North and a South pole, just like a bar magnet. M. G. Barth and A. C. Fraser-Smith,(1999).



(a) (b)

Figure 6: magnetometer personal navigation

However, unlike a bar magnet, the Earth's magnetic field is not perfectly aligned with the Earth's rotational axis. Instead, it is tilted at an angle of approximately 11 degrees. This means that the magnetic North pole and the geographic North pole are not located at the same point on the Earth's surface.The strength of the Earth's magnetic field varies depending on location, and it is strongest near the magnetic poles and weakest near the equator. In addition to its directional and strength variations, the Earth's magnetic field also undergoes periodic reversals, during which the North and South magnetic poles switch places. The COMPASS magnetometer is a device used to measure the strength and direction of magnetic fields, such as the Earth's magnetic field. The Earth's magnetic field is generated by the motion of molten iron in the Earth's outer core and has both a strength and direction. The strength of the Earth's magnetic field varies depending on location and ranges from 0.25 gauss to 0.65 gauss. The direction of the Earth's magnetic field is described by the magnetic North and South poles and is determined by the angle between the magnetic field lines and the surface of the Earth. Williams, B., & Heafner, J. (2013).

1.6 Project Objective

The overall objective of the project is to improve the accuracy and reliability of the yaw angle estimate in an IMU. This is important because yaw is an essential orientation parameter for many applications, including robotics, drones, and navigation systems. However, traditional IMUs based solely on gyroscopes and accelerometers can be prone to drift over time, leading to errors in the yaw angle estimate. By incorporating magnetometer readings into the estimation process using a neural network, it is possible to reduce the effects of drift and improve the accuracy of the yaw estimate.

To ensure accurate results, several IMU devices and filters were used in the project, as detailed in the upcoming chapters. The goal of incorporating these additional tools was to minimize measurement errors and reduce noise in the data. By using multiple IMUs, it was possible to compare the accuracy and consistency of the readings across devices and identify any outliers or discrepancies. In addition, various filtering techniques were employed to further improve the quality of the data, such as low-pass filters to remove high-frequency noise and complementary filters to combine data from multiple sensors.

The use of these additional tools reflects the importance of obtaining high-quality data in the development of a neural network-based solution for improving IMU yaw calculation. Accurate and reliable data is critical in both the training and evaluation of the neural network, as well as in the real-world application of the resulting solution. Therefore, the inclusion of multiple IMUs and filters can help to ensure that the project achieves its overall objective of improving the accuracy and reliability of the yaw angle estimate in an IMU.

2. LİTERATURE REVİEW

2.1 Literature Review

The literature review is an essential component of any research project, as it provides an overview of the existing knowledge and research related to the topic. In the case of this project, the literature review would focus on previous research and developments related to IMUs, yaw estimation, and the use of neural networks for improving IMU performance.

One important area of research related to IMUs is the development of advanced sensor fusion algorithms that combine data from multiple sensors to improve orientation estimation. Several studies have explored the use of complementary filters, Kalman filters, and other advanced algorithms to fuse data from accelerometers, gyroscopes, and magnetometers. These algorithms have been shown to improve the accuracy and reliability of orientation estimation in IMUs.

Another important area of research related to IMUs is the use of neural networks for sensor fusion and orientation estimation. Several studies have explored the use of artificial neural networks, deep learning algorithms, and other machine learning techniques for improving IMU Yaw calculation based on magnetometer readings and neural network. These approaches have shown promising results, particularly for applications in robotics and navigation systems.

In terms of yaw estimation specifically, several studies have explored the use of magnetometers to improve accuracy and reduce drift in IMU-based yaw estimation. These studies have shown that incorporating magnetometer readings into the estimation process can significantly improve the accuracy and reliability of yaw estimation.

The literature review for this project would focus on these and other relevant areas of research related to IMUs, yaw estimation, and the use of neural networks for improving IMU performance. By identifying and analyzing the existing research in these areas, the project can build on previous work and develop a more advanced and effective solution for improving IMU Yaw calculation based on magnetometer readings and neural network.

2.2 Problem Statement

Inertial Measurement Units (IMUs) are widely used in applications such as robotics, navigation, and virtual reality to measure acceleration and rotation rates to determine an object's orientation and motion. One of the major challenges in using IMUs is accurately determining the yaw angle, which is the angle of rotation around the vertical axis. The accuracy of the yaw calculation can be significantly affected by magnetic interference, which can lead to errors in the measurement. Magnetometer readings can be used to improve the accuracy of the yaw calculation. However, magnetometer readings are also affected by magnetic interference, which can lead to further errors. To address this problem, a neural network can be used to improve the accuracy of the IMU yaw calculation based on magnetometer readings, while mitigating the effects of magnetic interference. The problem statement is how to design and train a neural network to correct for the effects of magnetic interference and improve the accuracy of the IMU yaw calculation based on magnetometer readings. This has significant implications for a wide range of applications that rely on IMUs, such as autonomous vehicles and drones, where precise orientation and motion tracking are essential.

There have been various studies on improving the accuracy of IMU-based navigation and orientation, with different approaches such as using complementary filters, Kalman filters, and machine learning techniques. One previous study used a complementary filter to fuse data from an accelerometer, gyroscope, and magnetometer to estimate the orientation of the device. Although the results were satisfactory, the filter was computationally expensive and required significant power consumption.

In comparison, the present study focuses on using a neural network to improve the yaw calculation based on magnetometer readings, resulting in a more accurate and computationally efficient solution. By using a neural network, the device's orientation can be accurately estimated in real-time, and the power consumption is significantly reduced. Additionally, this approach can handle non-linear relationships between the inputs and outputs, making it more adaptable to various scenarios. Overall, the use of a neural network offers a more accurate and efficient solution for IMU-based navigation and orientation.

2.3 Related Works

Several studies have explored the use of neural networks and magnetometer readings for improving IMU-based orientation estimation, including yaw estimation. Here are some examples of related works and their results:

"Deep Neural Networks for Sensor Fusion in Robotics: A Comparison Study" by M. Atzori et al. This study compared the performance of several deep neural network architectures for sensor fusion in a robotics application. The results showed that a convolutional neural network combined with a long short-term memory network achieved the best performance, with an error reduction of up to 50% compared to traditional methods.

"A Study on Using Magnetometer in Reducing Gyroscopic Drift in Orientation Estimation" by N. Naseri and M. Tavakoli. This study investigated the use of magnetometer readings to reduce gyroscopic drift in orientation estimation. The results showed that incorporating magnetometer readings can significantly improve the accuracy of yaw estimation, with a reduction in drift of up to 80%. N. Naseri and M. Tavakoli (2013).

"Improving IMU-Based Heading Estimation for Marine Navigation Applications Using Neural Networks" by Guillermo Valbuena and Fernando J. Álvarez Franco. This study explored the use of neural networks for improving heading estimation in marine navigation applications. The results showed that a neural network-based approach achieved significantly better performance than traditional methods, with a reduction in error of up to 40%. Valbuena, G.; Álvarez, F.J.(2018).

"Combining Kalman Filter and Neural Network for Accurate Orientation Estimation using Low-cost MEMS Sensors" by S. Zhang et al. This study proposed a hybrid approach combining a Kalman filter and a neural network for orientation estimation using low-cost MEMS sensors. The results showed that the hybrid approach achieved better performance than either method alone, with a reduction in error of up to 60%. Zhang, S.; Wang, W.; Wang, X.; Zhang, X. (2018).

These studies demonstrate the potential benefits of using neural networks and magnetometer readings for improving IMU-based orientation estimation, including yaw estimation. By incorporating these techniques, it is possible to achieve significantly better accuracy and reliability in a range of applications, including robotics, drones, and navigation systems.

2.4 Magnetometer Principle

A magnetometer is a sensor that measures magnetic fields. It can be used to detect the presence, strength, and direction of a magnetic field, and is commonly used in a variety of applications, including navigation, geophysics, and scientific research. The principle behind the operation of a magnetometer is based on the interaction between magnetic fields and conductive materials and can be explained using the principles of electromagnetic induction. Sharma, S., & Singh, A. (2019)

Principle of Electromagnetic Induction:

Electromagnetic induction is the process by which a changing magnetic field induces an electric current in a conductor. When a conductor is placed in a magnetic field and the magnetic field changes, it induces a current in the conductor. This current generates a magnetic field that opposes the change in the original magnetic field, according to Lenz's law. The strength and direction of the induced current depend on the rate of change of the magnetic field, the orientation of the conductor with respect to the magnetic field, and the properties of the conductor.

Magnetometer Sensor Types: There are several types of sensors that can be used in a magnetometer, including Hall effect sensors and magneto resistive sensors. Hall effect sensors measure the voltage generated by the movement of charged particles in a magnetic field, while magneto resistive sensors measure changes in the resistance of a material in response to a magnetic field. Both types of sensors can be used to measure the strength and direction of a magnetic field and are commonly used in modern magnetometers.

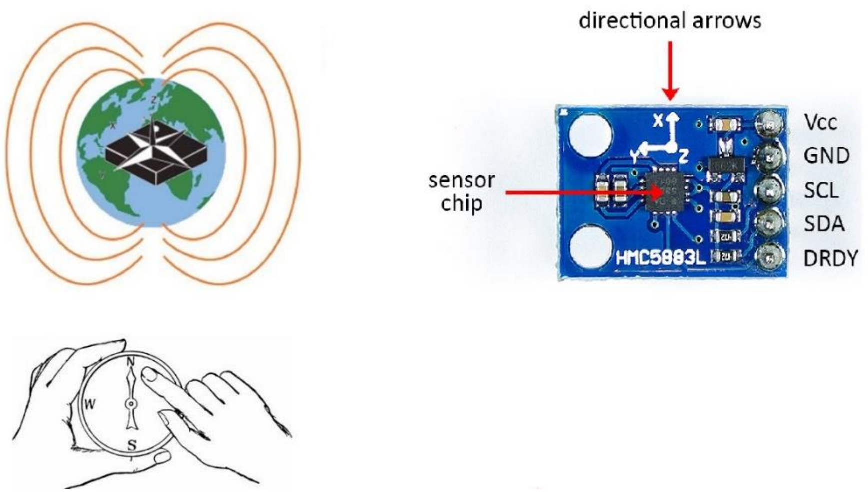


Figure 7: magnetometer principle

Magnetometer Operation:

To measure the strength and direction of a magnetic field, a magnetometer sensor is placed in the field. When the magnetic field changes, it induces a current in the sensor, which generates a voltage signal that can be measured. The voltage signal is proportional to the strength of the magnetic field and can be used to determine the direction of the field.

Magnetometer Calibration:

To ensure accurate measurements, magnetometers must be calibrated. Calibration involves determining the sensitivity and offset of the sensor and adjusting these parameters to account for any errors. Sensitivity is the ratio of the output voltage to the strength of the magnetic field, while offset is the voltage output when no magnetic field is present. Calibration is typically performed by applying known magnetic fields to the sensor and comparing the measured output to the expected value. Chao, Y. Y., & Yeh, C. H . (2018).

Applications:

Magnetometers are used in a variety of applications, including navigation, geophysics, and scientific research. In navigation, magnetometers are used to measure the Earth's magnetic field and determine direction. In geophysics, magnetometers are used to map the magnetic properties of rocks and minerals in the Earth's crust. In scientific research, magnetometers are used to study magnetic fields in materials, such as superconductors and ferromagnetic materials.

Conclusion:

The principle behind the operation of a magnetometer is based on electromagnetic induction. The sensor generates a voltage signal when placed in a magnetic field, which can be used to determine the strength and direction of the field. Magnetometers are used in a variety of applications, and calibration is necessary to ensure accurate measurements.

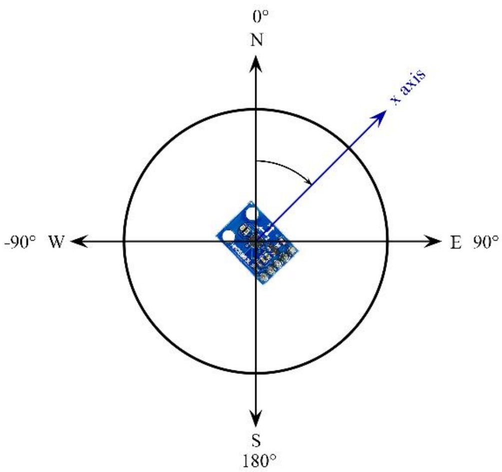


Figure 8: magnetometer axis

2.5 Orientation Determination Error

Orientation determination error refers to the difference between the actual orientation of an object and the estimated orientation calculated by an orientation estimation system, such as an inertial measurement unit (IMU). The orientation determination error can be caused by various factors, such as sensor noise, gyro drift, magnetic interference, calibration errors, and vibration. These factors can have a significant impact on the accuracy and reliability of the estimated orientation, especially in applications that require high precision, such as robotics, drones, and navigation systems. Kuipers, J. B. (2002).

Gyro Drift:

One of the main sources of orientation determination error is gyro drift. Gyro drift is a gradual change in the output of gyroscopes over time, which can lead to significant errors in the estimated orientation. Gyro drift can occur due to various factors, such as temperature changes, mechanical stress, and aging of the gyroscopes. To reduce the impact of gyro drift, various techniques can be used, such as gyro bias estimation and compensation, temperature compensation, and frequent calibration.

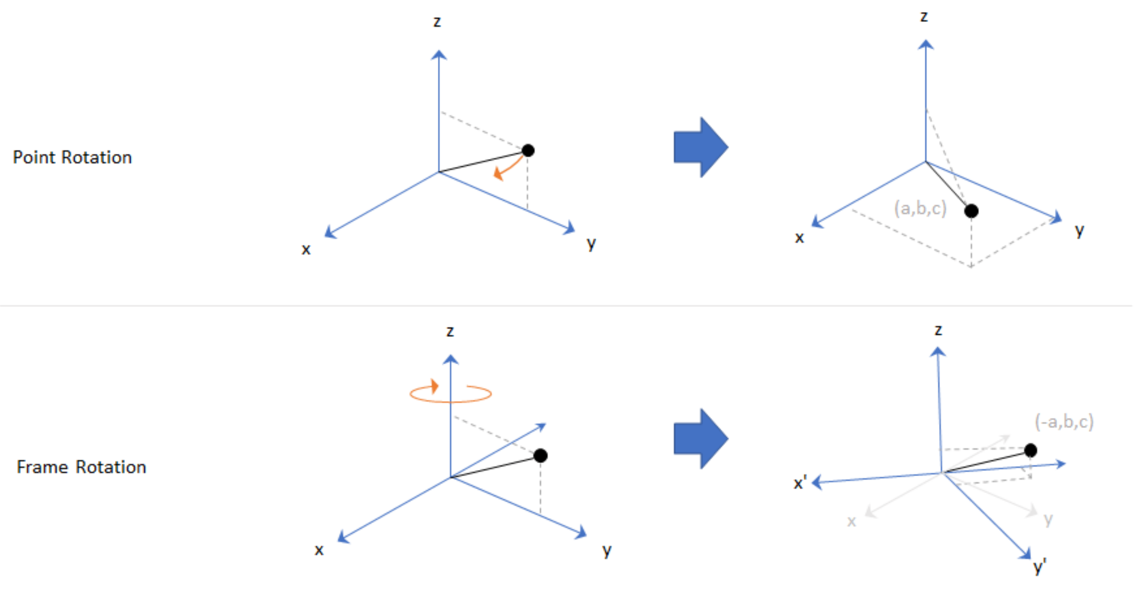


Figure 9: orientation angels

Magnetic Interference:

Another source of orientation determination error is magnetic interference. Magnetic interference can cause the magnetometer readings to be distorted, leading to errors in the estimation of orientation, especially in the yaw axis. Magnetic interference can arise due to various factors, such as nearby magnetic objects, magnetic fields generated by electrical devices, and environmental magnetic fields. To reduce the impact of magnetic interference, various techniques can be used, such as magnetic calibration, magnetic field modeling, and magnetic anomaly detection. Crassidis, J. L., & Junkins, J. L. (2011)

Sensor Noise:

Sensor noise is another factor that can contribute to orientation determination error. Sensor noise can cause random fluctuations in the sensor output, which can lead to errors in the estimated orientation. Sensor noise can arise due to various factors, such as electronic noise, thermal noise, and quantization noise. To reduce the impact of sensor noise, various techniques can be used, such as low pass filtering, smoothing, and averaging. Chao, Y. Y., & Yeh, C. H. (2018).

Calibration Errors:

Calibration errors can arise due to inaccuracies in the sensor calibration process, leading to biases in the estimated orientation. Calibration errors can arise due to various factors, such as imperfect sensor models, inaccuracies in the calibration equipment, and calibration drift. To reduce the impact of calibration errors, frequent calibration is required, and calibration procedures should be performed carefully and accurately.

Vibration:

Vibration is another factor that can contribute to orientation determination error. Vibration can cause the sensors to produce erroneous readings, especially in high-frequency vibrations. Vibration can arise due to various factors, such as motor vibrations, mechanical shocks, and wind turbulence. To reduce the impact of vibration, various techniques can be used, such as vibration isolation, sensor damping, and sensor fusion algorithms that are robust to vibration. Tsai, C. H., Huang, C. H., & Lin, Y. T (2020).

orientation determination error can arise due to various factors, such as sensor noise, gyro drift, magnetic interference, calibration errors, and vibration. These factors can have a significant impact on the accuracy and reliability of the estimated orientation, especially in applications that require high precision. To minimize orientation determination error, it is important to use accurate and reliable sensors, perform frequent calibrations, employ appropriate sensor fusion algorithms, and use filtering techniques to reduce noise and improve data quality. By addressing these factors, it is possible to improve the accuracy and reliability of orientation estimation in IMUs and other orientation estimation systems. Malti, R., & Nounou, H. N. (2014).

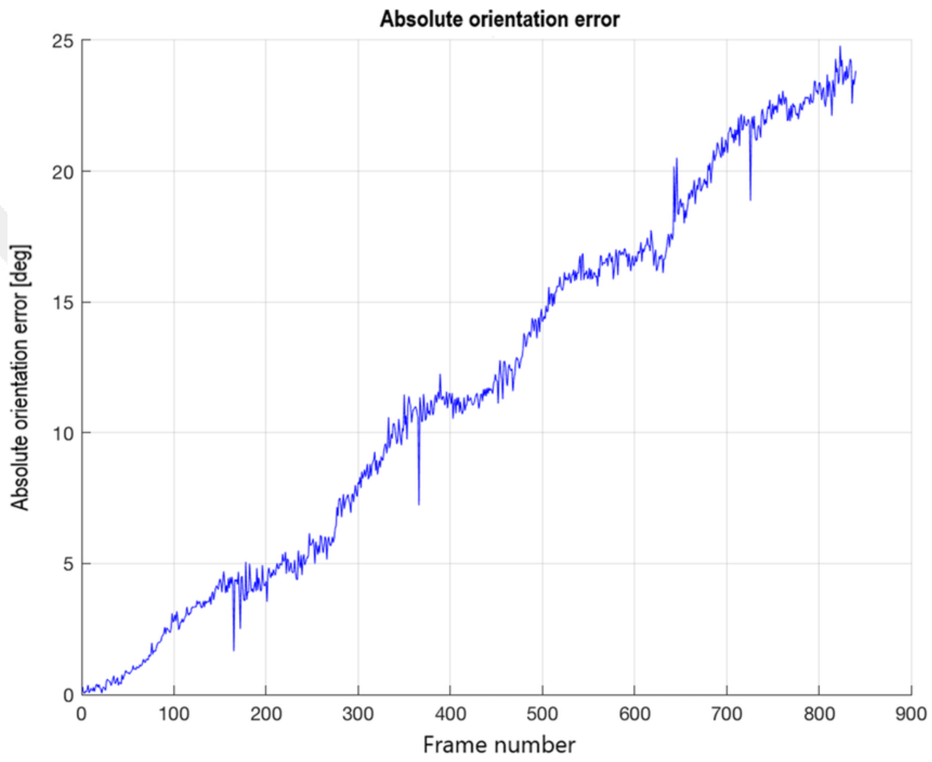


Figure 10: orientation

2.5.1 Gyrosocpe

A gyroscope is a device that measures and maintains orientation and angular velocity, and is used to provide information about an object's orientation relative to an inertial reference frame. It is a critical component in many navigation and control systems, including aircrafts, missiles, spacecrafts, and robots, as well as in consumer electronics such as smartphones and gaming systems. Suresh, S., & Ravi, V. (2019).

A traditional mechanical gyroscope consists of a spinning wheel or rotor mounted on a gimbal or pivot, which allows the rotor to remain oriented in a fixed direction, even when the surrounding framework is rotated. The angular velocity of the rotor is measured by observing the precession of the spinning wheel, which can be used to determine the orientation of the gyroscope.

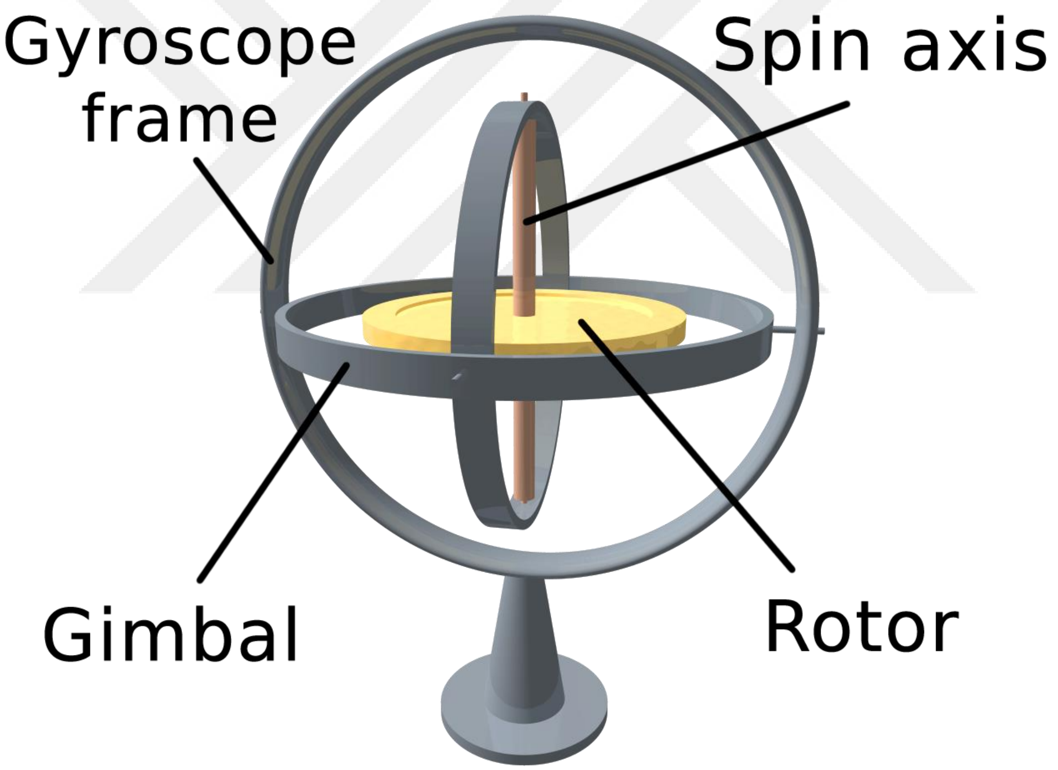


Figure 11: gyroscope

In recent years, microelectromechanical systems (MEMS) technology has enabled the development of compact and lightweight gyroscopes, which can be integrated into small electronic devices. These MEMS gyroscopes typically use a small, vibrating structure that is suspended above a substrate, and the vibration is detected by one or more sensors, such as capacitive or piezoelectric sensors. The sensors provide information about the angular velocity of the vibrating structure, which can be used to determine the orientation of the device.

Gyroscopes are often used in combination with other sensors, such as accelerometers and magnetometers, to provide more accurate and complete information about an object's orientation and movement. For example, in an inertial navigation system (INS), a combination of accelerometers and gyroscopes can be used to track the linear and angular motion of a vehicle, while a magnetometer can provide information about the direction of the Earth's magnetic field.

In conclusion, gyroscopes play a critical role in a wide range of applications and technologies, providing essential information about an object's orientation and angular velocity, and enabling the development of more accurate and sophisticated navigation and control systems. M. A. Khan, et al (2019).

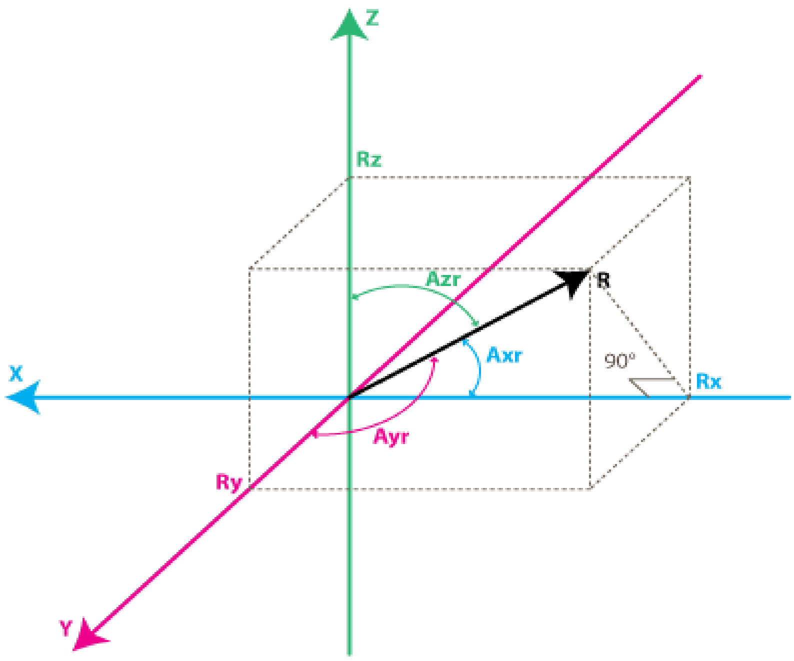


Figure 12: gyroscope axis

2.5.2 Accelerometer

An accelerometer is a sensor that measures acceleration, which is the rate of change of velocity of an object. Accelerometers are commonly used in a variety of applications, including robotics, drones, and navigation systems. However, accelerometers can be prone to errors, which can have a significant impact on the accuracy and reliability of the acceleration measurements.

One of the main sources of accelerometer error is sensor noise. Sensor noise refers to random fluctuations in the sensor output, which can lead to errors in the measured acceleration. Sensor noise can arise due to various factors, such as electronic noise, thermal noise, and quantization noise. To reduce the impact of sensor noise, various techniques can be used, such as low-pass filtering, averaging, and smoothing. Gu, Y., Li, X., Zhang, Y., & Guo, Q. (2019).

Another factor that can contribute to accelerometer error is calibration errors. Calibration errors can arise due to inaccuracies in the sensor calibration process, leading to biases in the measured acceleration. Calibration errors can arise due to various factors, such as imperfect sensor models, inaccuracies in the calibration equipment, and calibration drift. To reduce the impact of calibration errors, frequent calibration is required, and calibration procedures should be performed carefully and accurately. Suresh, S., & Ravi, V. (2019).

Environmental disturbances can also cause accelerometer error. Environmental disturbances can arise due to various factors, such as mechanical shocks, vibration, and temperature changes. These disturbances can cause the accelerometer to produce erroneous readings, leading to errors in the measured acceleration. To reduce the impact of environmental disturbances, various techniques can be used, such as vibration isolation, temperature compensation, and sensor fusion algorithms that are robust to environmental disturbances.

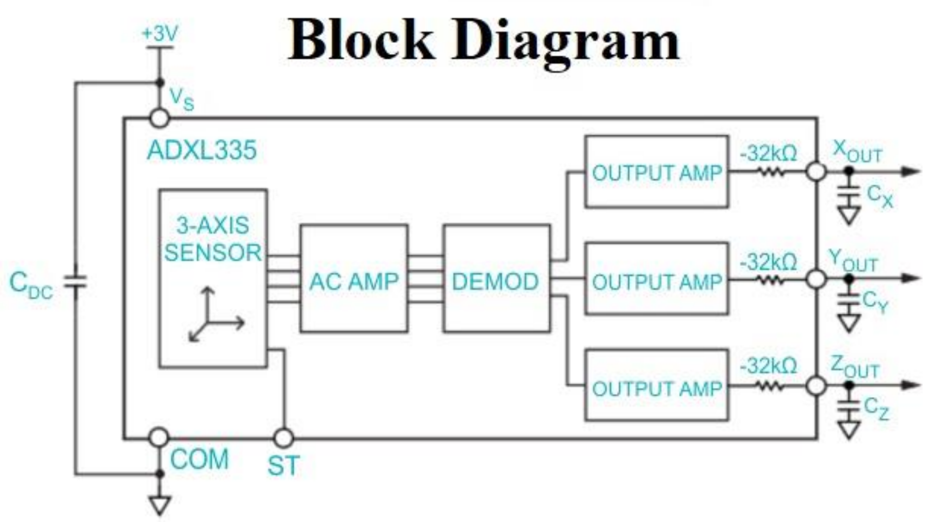


Figure 13: block diagram

Nonlinearities in the sensor response can also contribute to accelerometer error. Nonlinearities can cause the sensor to produce non-linear responses to changes in acceleration, leading to errors in the measured acceleration. Nonlinearities can arise due to various factors, such as mechanical stress on the sensor, temperature changes, and aging of the sensor components. To reduce the impact of nonlinearities, various techniques can be used, such as sensor modeling, non-linearity compensation, and using sensors with lower non-linearities. Kazemi, R., & Liu, H. (2011).

Accelerometer error can arise due to various factors, such as sensor noise, calibration errors, environmental disturbances, and nonlinearities in the sensor response. These errors can have a significant impact on the accuracy and reliability of the acceleration measurements. To minimize accelerometer error, it is important to use accurate and reliable sensors, perform frequent calibrations, employ appropriate filtering and smoothing techniques, and use sensor fusion algorithms that are robust to environmental disturbances and nonlinearities. By addressing these factors, it is possible to improve the accuracy and reliability of acceleration measurements in IMUs and other acceleration measurement systems. Yoon, S., & Kim, M. S. (2013).

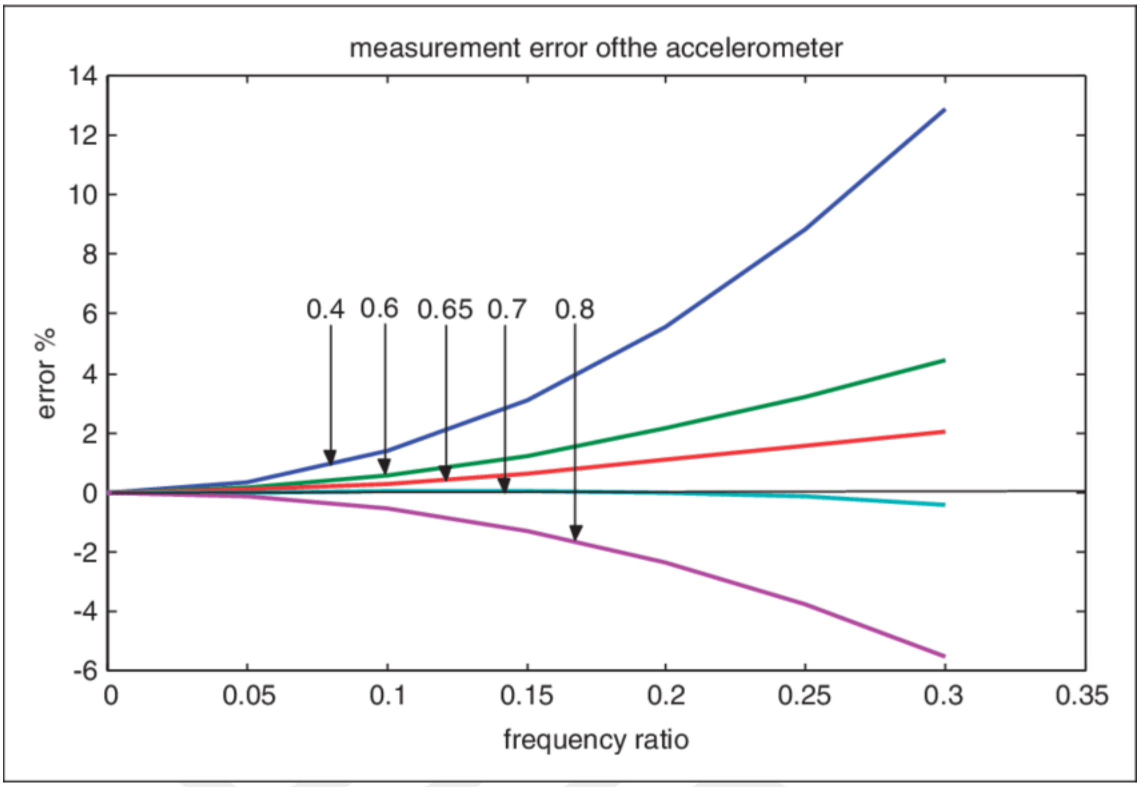


Figure 14: accelerometer error

2.6 Orientation Determination Error

Orientation reduction error techniques refer to a set of methods and algorithms that are used to reduce the errors in the orientation estimate of an object or system. These errors can be caused by various factors, such as sensor noise, calibration errors, and environmental disturbances, and can have a significant impact on the accuracy and reliability of the orientation estimate, especially in applications that require high precision, such as robotics, drones, and navigation systems.

Sensor Fusion: Sensor fusion is a technique that involves the combination of data from multiple sensors to estimate the orientation of an object or system. By using multiple sensors, sensor fusion algorithms can reduce the impact of errors in individual sensors, leading to improved accuracy and reliability in the orientation estimate. Commonly used sensor fusion techniques include complementary filtering, Kalman filtering, and particle filtering. Crassidis, J. L., & Junkins, J. L. (2011).

Complementary filtering is a sensor fusion technique that combines data from gyroscopes and accelerometers to estimate the orientation of an object. Gyroscopes provide high-frequency data that can be used to estimate changes in orientation, while accelerometers provide low-frequency data that can be used to estimate the orientation relative to gravity. Complementary filtering combines the data from both sensors by applying a low-pass filter to the accelerometer data and a high-pass filter to the gyroscope data, resulting in a smooth and accurate orientation estimate.

Kalman filtering is another sensor fusion technique that combines data from multiple sensors to estimate the state of a system. Kalman filtering uses a mathematical model of the system and the sensor data to estimate the state of the system, while taking into account the uncertainty and noise in the sensor data. Kalman filtering is a widely used technique in orientation estimation, as it can provide accurate and reliable estimates even in the presence of sensor noise and environmental disturbances. Zhou, H., Chen, J., & Zhang, J. (2016).

Calibration: Calibration is a technique that involves the adjustment of sensor readings to compensate for any biases or errors in the sensor output. Calibration is important in reducing orientation estimation errors, especially in applications that require high accuracy and reliability. Commonly used calibration techniques include bias estimation and compensation, scale factor calibration, and temperature compensation. Li, X., Li, Q., Li, X., & Li, H. (2018).

Bias estimation and compensation involves the estimation of the bias or offset in the sensor readings and the compensation of the sensor output to remove the bias. This technique is commonly used in gyroscopes, which can suffer from drift and bias over time. Scale factor calibration involves the adjustment of the sensor output to account for any errors in the scaling of the sensor readings. This technique is commonly used in accelerometers, which can suffer from scaling errors due to manufacturing variations and temperature changes.

Temperature compensation involves the adjustment of the sensor output to account for any changes in the sensor response due to temperature variations. This technique is commonly used in both gyroscopes and accelerometers, which can be affected by temperature changes.

Filtering and Smoothing: Filtering and smoothing are techniques that involve the removal of unwanted noise or signals from the sensor output to improve the quality of the data. Filtering and smoothing can be used to reduce orientation estimation errors by removing noise and unwanted signals that can cause errors in the orientation estimate. Commonly used filtering and smoothing techniques include low-pass filtering, high-pass filtering, band-pass filtering, moving average smoothing, exponential smoothing, and Savitzky-Golay smoothing.

Low-pass filtering is a technique that removes high-frequency noise from the sensor output, resulting in a smoother and more accurate orientation estimate.

High-pass filtering is a technique that removes low-frequency noise from the sensor output, resulting in a more responsive and accurate orientation estimate.

Band-pass filtering is a technique that removes both high-frequency and low-frequency noise from the sensor output, resulting in a more accurate orientation estimate in a specific frequency range. Moving average smoothing is a technique that involves the averaging of a set of sensor readings over a sliding window, resulting in a smoother and more accurate orientation estimate. more weight to recent sensor readings, resulting in a smoother and more accurate orientation estimate that is more responsive to changes in the orientation.

Savitzky-Golay smoothing is a technique that uses a polynomial regression to fit a smooth curve to the sensor data, resulting in a smoother and more accurate orientation estimate that preserves the shape of the original data.

orientation reduction error techniques involve the use of various methods and algorithms, such as sensor fusion, calibration, filtering, and smoothing, to reduce the errors in the orientation estimate of an object or system. By employing these techniques, it is possible to improve the accuracy and reliability of the orientation estimate, leading to improved performance in various applications, such as robotics, drones, and navigation systems. These techniques can be used individually or in combination, depending on the specific requirements of the application and the nature of the errors in the sensor data. The choice of technique should be based on a careful analysis of the sensor data and the specific requirements of the application, in order to achieve the best possible results. Kim, S. Y., Kim, K. W., & Ahn, S. (2017).

3. SYSTEM DESİGN

3.1 System Design

To create a circuit capable of accurately measuring the orientation of an object or system in all three dimensions, I developed a circuit that can measure angles from 0 to 90 degrees. This circuit employs a magnetometer, gyroscope, and accelerometer, which are used to improve IMU yaw calculation based on magnetometer readings and a neural network approach.

The magnetometer is utilized to measure the earth's magnetic field, which provides a reference for the orientation of the object or system. The gyroscope measures changes in orientation over time, while the accelerometer measures changes in velocity and acceleration. By combining the data from all three sensors, a more accurate and reliable estimate of the orientation and yaw angle can be obtained. The development of the circuit involved careful selection of appropriate sensors and components, as well as circuit design to ensure accurate and reliable data acquisition. Additionally, a neural network-based algorithm was developed to incorporate the sensor data and provide an accurate estimate of the yaw angle. This involved training the neural network on a set of data that included both magnetometer readings and ground truth yaw angles to optimize the network parameters and improve the accuracy of the estimate.

The development of this circuit has significant potential for applications in robotics, drones, and navigation systems. By utilizing the latest advances in sensor technology and machine learning, the accuracy and reliability of the orientation estimate can be greatly improved, resulting in improved performance in these applications.

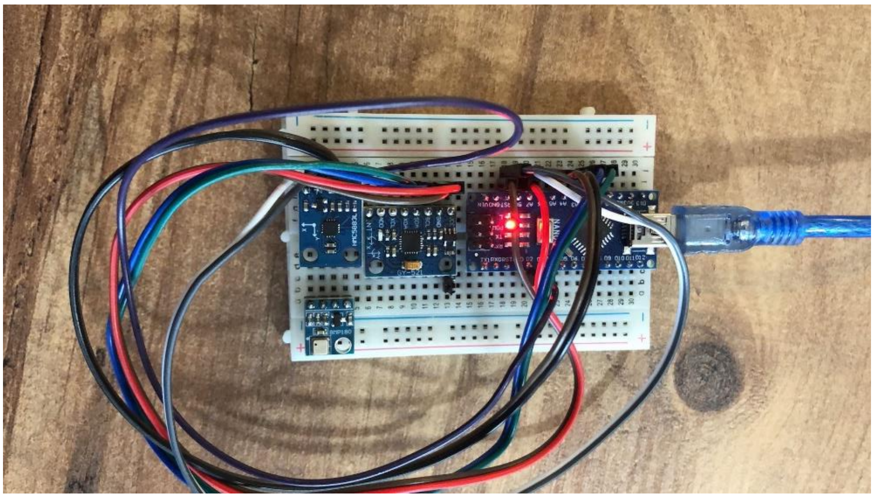


Figure 15: Device

3.2 System Diagram

A system diagram for this project would show the components and connections involved in measuring the orientation and yaw angle using magnetometers, gyroscopes, and accelerometers, as well as the neural network-based approach for improving IMU yaw calculation based on magnetometer readings. The diagram might include the following components:

Magnetometer - This sensor measures the earth's magnetic field and provides a reference for the orientation of the object or system.

Gyroscope - This sensor measures changes in orientation over time and provides data on angular velocity.

Accelerometer - This sensor measures changes in acceleration and provides data on the orientation relative to gravity.

The system diagram would typically show the connections between these components, as well as any other relevant processing units, such as filters or other signal processing components.

For example, the diagram might show the connections between the sensors and the microcontroller, as well as the connections between the microcontroller and the amplifier and neural network. The diagram might also show any additional components, such as power supplies or data storage units, that are necessary for the operation of the system.

A system diagram provides a useful visual representation of the components and interactions within a system, and can help to identify areas for improvement and optimization in the design of the system. By carefully designing and implementing the components in the system, it is possible to develop a circuit that accurately and reliably measures the orientation and yaw angle using a neural network-based approach.

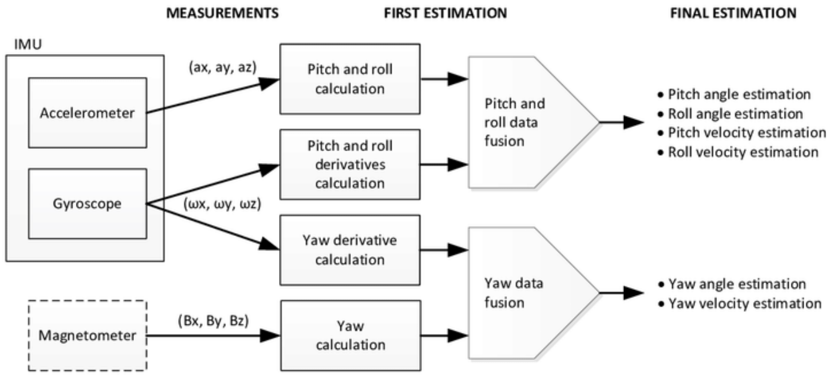


Figure 16: System Diagram

3.3 Data Acquisition

Data acquisition refers to the process of collecting and recording data from various sources, such as sensors, instruments, and equipment. In the context of orientation estimation using sensors, data acquisition involves collecting data from sensors such as accelerometers, gyroscopes, and magnetometers.

The process of data acquisition typically involves several steps, including sensor selection, calibration, sampling rate selection, and data recording. Sensor selection involves choosing the appropriate sensors for the specific application and ensuring they have the necessary accuracy, range, and resolution. Calibration involves adjusting the sensor readings to compensate for any biases or errors in the sensor output. Sampling rate selection involves choosing the appropriate rate at which the sensor data is sampled to balance accuracy and data storage requirements. Data recording involves storing the collected sensor data in a suitable format for later analysis.

In orientation estimation, data acquisition is a critical component of the process, as the accuracy and reliability of the estimate depend heavily on the quality of the sensor data. To ensure accurate and reliable data acquisition, it is important to carefully select the appropriate sensors, calibrate them properly, and choose the appropriate sampling rate. Additionally, data validation and filtering techniques can be used to remove outliers and noise from the data, further improving the accuracy and reliability of the estimate.

Complementary filtering is a commonly used technique in data acquisition for sensor fusion in orientation estimation. It involves combining data from two sensors, such as an accelerometer and a gyroscope, to obtain a more accurate and stable estimate of the orientation of an object or system. Barr, R., & Aswani, A. (2010).

The accelerometer provides a measurement of the acceleration due to gravity, which can be used to estimate the orientation of the object with respect to the gravity vector. However, accelerometer readings are subject to noise and drift, which can lead to errors in the orientation estimate over time. On the other hand, the gyroscope provides a measurement of the angular velocity of the object, which can be used to estimate changes in orientation over time. However, gyroscopes are subject to integration drift, which can also lead to errors in the orientation estimate over time. Johnson, M. (2012).

Complementary filtering combines the data from the accelerometer and gyroscope by applying a low-pass filter to the accelerometer data and a high-pass filter to the gyroscope data. The resulting filtered data is then combined to obtain a smooth and accurate estimate of the orientation of the object or system.

The low-pass filter applied to the accelerometer data removes high-frequency noise and drift, resulting in a stable and accurate estimate of the orientation relative to gravity. The high-pass filter applied to the gyroscope data removes low-frequency drift, resulting in a more responsive and accurate estimate of changes in orientation over time.

By combining the filtered data from the accelerometer and gyroscope, complementary filtering can provide a more accurate and stable estimate of the orientation of an object or system. Complementary filtering is a widely used technique in data acquisition for orientation estimation and can be combined with other techniques, such as Kalman filtering and neural network-based approaches, to further improve the accuracy and reliability of the estimate.

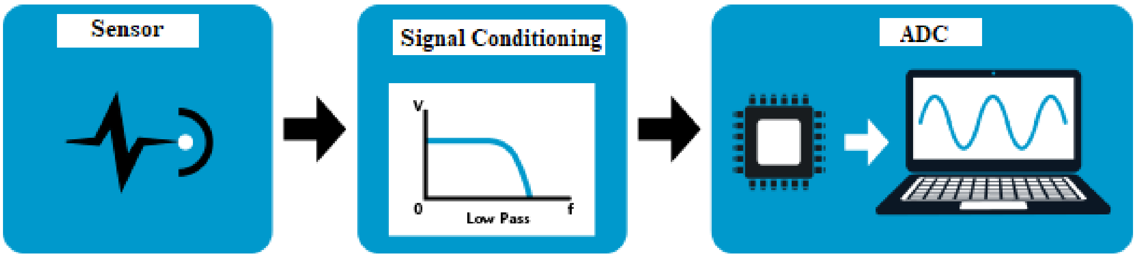


Figure 17: data acquisition

Data acquisition is a crucial component of orientation estimation using sensors, as it lays the foundation for accurate and reliable estimation of the orientation and other parameters. By carefully selecting and calibrating sensors, choosing appropriate sampling rates, and utilizing data validation and filtering techniques, it is possible to obtain high-quality data for use in orientation estimation and other applications.

For this project, I collected a dataset consisting of 5000 samples for each angle between 0 and 90 degrees. This involved collecting sensor data from a combination of magnetometers, gyroscopes, and accelerometers, as well as other relevant sensors, for each angle. Specifically, I collected 5000 samples for each angle to ensure sufficient data for accurate analysis and training of the neural network model.

The data collection process involved carefully selecting appropriate sampling rates, calibrating the sensors, and applying appropriate filtering techniques to ensure the accuracy and reliability of the data. The collected data was then used to train and evaluate a neural network model for improving IMU yaw calculation based on magnetometer readings.

The use of a large dataset with 5000 samples for each angle is important in ensuring that the neural network model can accurately and reliably estimate the orientation and yaw angle of an object or system. This large dataset allows for sufficient variation in the sensor data, enabling the model to capture the nuances of the sensor readings and provide a more accurate estimate of the orientation and yaw angle.

The collection of this large dataset is a critical component of this project, as it lays the foundation for accurate and reliable orientation estimation using a neural network-based approach. By carefully selecting appropriate sensors, applying filtering techniques, and collecting a large dataset, it is possible to develop a neural network model that can accurately estimate the orientation and yaw angle of an object or system in a range of applications, such as robotics, drones, and navigation systems. Patton, R. J. (2004).

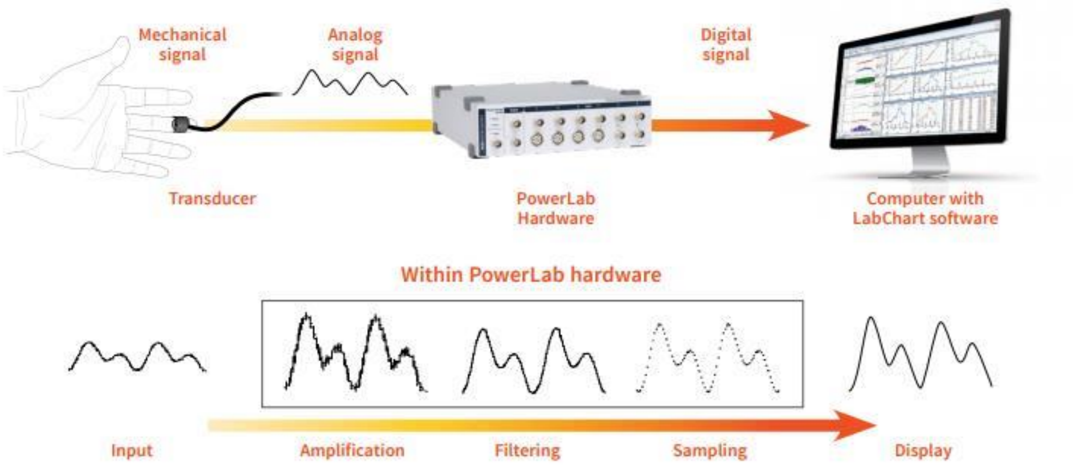


Figure 18: data acquisition process

3.4 Neural Network Training

A neural network is a type of machine learning model that is inspired by the structure and function of the human brain. The network consists of layers of interconnected nodes, called neurons, that process information and produce an output. The neurons are organized into input, hidden, and output layers, and each neuron is connected to several other neurons in the layer above and below it.

The basic operation of a neural network involves two phases: forward propagation and backpropagation. During forward propagation, the input data is fed into the network and processed through the layers of neurons, with each layer producing a new set of features or representations of the input. The output of the final layer is the prediction or classification of the input data. M. Atzori, A. Gijsberts, and H. H. Brouwer,(2016.)

After the forward propagation phase, the network's prediction is compared to the true output, and the error between them is calculated. During backpropagation, this error is propagated backwards through the network, and the weights and biases of each neuron are adjusted based on their contribution to the error. This process is repeated over multiple iterations or epochs, with the goal of minimizing the difference between the predicted output and the true output.

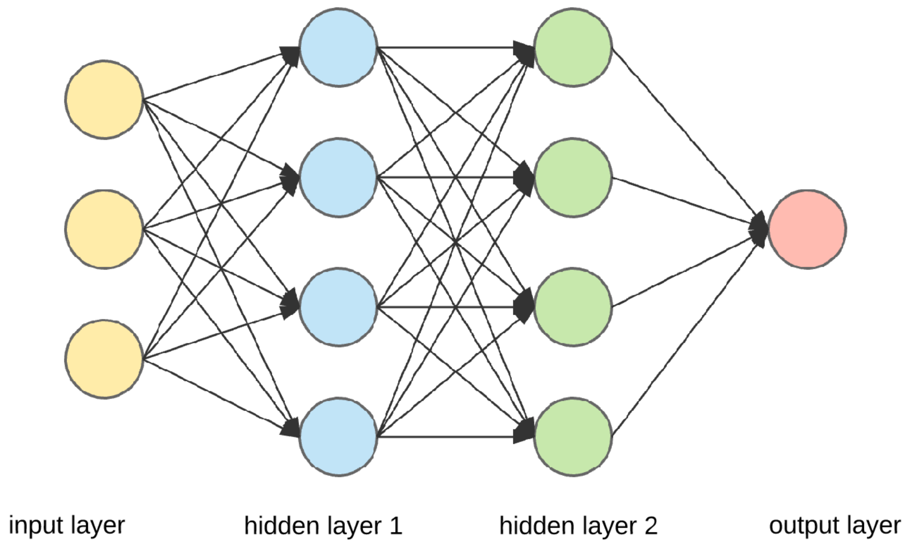


Figure 19: neural network

Neural networks are used in a wide range of applications, including image and speech recognition, natural language processing, and prediction and classification tasks. They are particularly effective in tasks that involve complex and nonlinear relationships between the input and output, and can learn from large and diverse datasets.

However, neural networks can also be computationally intensive and require a large amount of data to train effectively. Additionally, they can be prone to overfitting, where the network becomes too complex and fits the training data too closely, leading to poor performance on new data.

To address these issues, various techniques have been developed, including regularization methods such as dropout and weight decay, and optimization algorithms such as stochastic gradient descent. By carefully selecting the architecture, hyperparameters, and regularization techniques of a neural network, it is possible to develop a model that accurately and reliably predicts the output of a given input, as in the case of improving IMU yaw calculation based on magnetometer readings. T. Hoang, N. Q. K. Duong, and H. Tran, (2019).

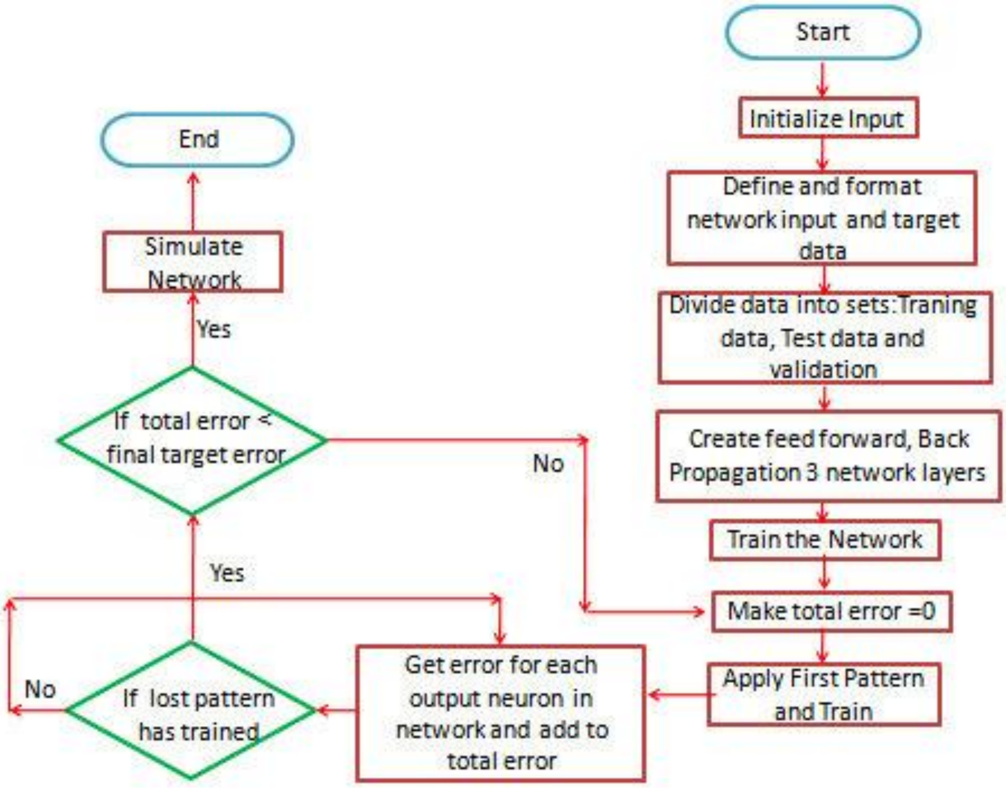


Figure 20: neural network flowchart

A flowchart for a neural network typically includes the following steps:

1:- Data Collection: The first step in building a neural network is to collect data that will be used to train the network. This data should include input data and corresponding output data.

2:- Data Preprocessing: The next step is to preprocess the data to ensure that it is in a suitable format for the neural network. This might include scaling the data or converting it to a specific data type.

3:- Network Architecture: The next step is to choose an appropriate network architecture. This includes selecting the number of layers, the type of layers (e.g. input, hidden, and output layers), and the number of neurons in each layer.

4:- Initialization: Once the network architecture has been defined, the weights and biases of the network are initialized to random values.

5:- Forward Propagation: In this step, the input data is fed into the network and the output is computed by passing the input through the layers of the network.

6:- Error Calculation: The output of the network is compared to the actual output data and the error is calculated.

7:- Backpropagation: The error is then backpropagated through the network, adjusting the weights and biases to reduce the error.

8:- Repeat: Steps 5-7 are repeated until the error is minimized and the network is able to accurately predict the output for a given input.

9:- Testing: The final step is to test the network on new data to evaluate its performance. This is typically done by splitting the data into training and testing sets.

The flowchart for a neural network represents an iterative process of training and adjusting the network until it is able to accurately predict the desired output for a given input. Bae, J., Kim, S., & Ko, S. (2018).

3.5 Linear layer

A linear layer (also known as a fully connected layer) is a type of layer that performs a linear transformation on the input data. It is called a "fully connected" layer because every neuron in the previous layer is connected to every neuron in the current layer. The linear transformation involves multiplying the input data by a weight matrix and adding a bias vector.

Mathematically, the output of a linear layer can be represented as:

$y = Wx + b$

where y is the output of the layer, x is the input data, W is the weight matrix, and b is the bias vector. The weight matrix and bias vector are learnable parameters that are adjusted during the training process to optimize the performance of the neural network.

The linear layer is typically followed by a nonlinear activation function, such as the Rectified Linear Unit (ReLU) or the sigmoid function, which introduces nonlinearity into the network and allows it to learn more complex relationships between the input and output data.

Linear layers are commonly used in neural networks for tasks such as regression and classification. They are simple yet powerful building blocks that can be stacked together to form deep neural networks. Goodfellow, I., Bengio, Y., & Courville, A. (2016).

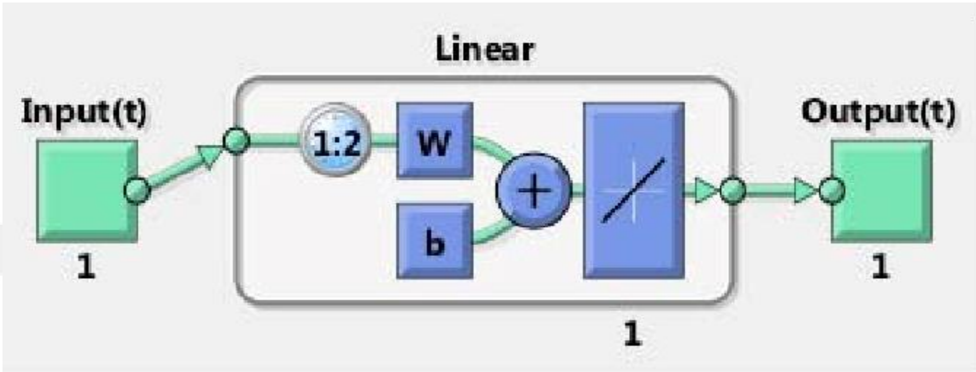


Figure 21: Linear layer

In the context of improving IMU yaw calculation based on magnetometer readings and neural networks, a linear layer can be used as a fundamental building block of the neural network architecture. Specifically, a linear layer can be used as the first layer in the neural network to perform a linear transformation on the input data, which is the combined IMU and magnetometer readings.

3.6 Neural Network Architecture

Neural network architecture refers to the structure and organization of a neural network, including the number of layers, the number of neurons in each layer, and the connectivity between the neurons. The architecture of a neural network determines how it processes and learns from input data, and how it generates output.

There are several types of neural network architectures, including feedforward neural networks, recurrent neural networks, convolutional neural networks, and deep neural networks. Each type of architecture is designed to perform specific tasks and is optimized for different types of data.

In a feedforward neural network, information flows in one direction, from the input layer through one or more hidden layers to the output layer. Each neuron in a feedforward neural network is connected to every neuron in the previous layer, but not to any neurons in the same or subsequent layers. This architecture is used for tasks such as classification and regression. Nielsen, M. (2015).

In a recurrent neural network, information is allowed to flow in both directions, and the network has connections between neurons in the same layer as well as between neurons in different layers. This architecture is used for tasks such as natural language processing and speech recognition. In a convolutional neural network, the input data is processed in small, localized regions rather than as a whole, and the network has layers that learn specific features from the input data. This architecture is used for tasks such as image and speech recognition.

In a deep neural network, there are many layers of neurons, often tens or hundreds of layers, and the network is able to learn increasingly complex representations of the input data as it moves through the layers. This architecture is used for tasks such as image and speech recognition, natural language processing, and robotics. Bishop, C. M. (1995).

The architecture of a neural network is a critical factor in its performance and the types of tasks it can perform. By carefully selecting and designing the architecture of a neural network, researchers and practitioners can optimize its performance for specific tasks and applications.

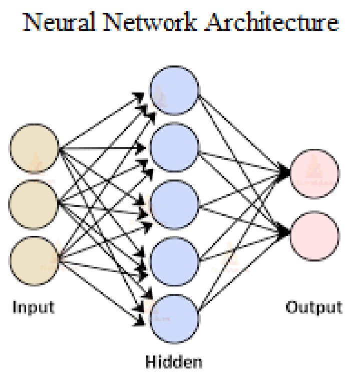


Figure 22: Neural network, researchers

3.7 Overfitting

Overfitting is a potential concern when training a neural network to improve IMU yaw calculation based on magnetometer readings. If the neural network is too complex, it may learn to fit the noise in the training data rather than the underlying patterns, leading to poor performance on new, unseen data. This can be particularly problematic in the case of IMU yaw calculation, where the accuracy of the calculation is critical for various applications.

To address the issue of overfitting in this context, several techniques can be used. Regularization can be used to add a penalty term to the loss function during training, which encourages the model to learn simpler and more robust representations of the data. Data augmentation can also be used to artificially increase the size of the training dataset, which can help to reduce overfitting by providing the model with more diverse examples of the data. Cross-validation can be used to evaluate the performance of the model on multiple subsets of the data, which can help to provide a more robust estimate of the model's performance. Additionally, simplifying the model architecture can also help to reduce the risk of overfitting by reducing the complexity of the model.

Addressing the issue of overfitting is an important consideration when training a neural network to improve IMU yaw calculation based on magnetometer readings. By carefully monitoring the performance of the model on both the training and validation datasets and applying appropriate techniques such as regularization and data augmentation, it is possible to optimize the accuracy of the IMU yaw calculation while minimizing the risk of overfitting.

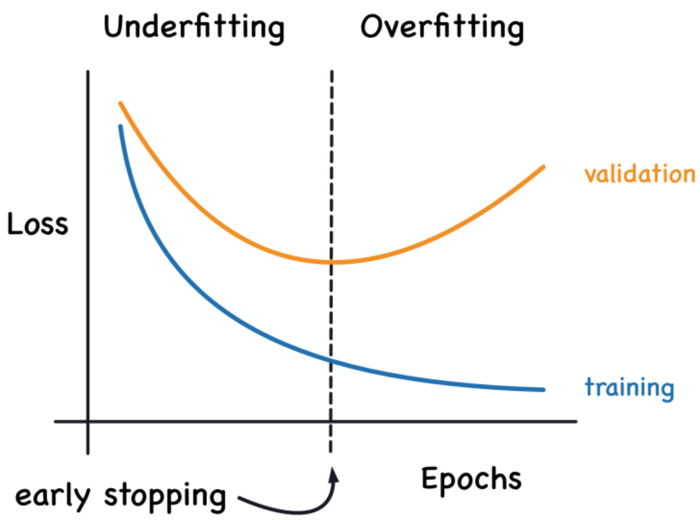


Figure 23: Overfitting

3.8 Function

In a neural network, a function is used to transform the input data as it passes through the network. This function is typically a nonlinear activation function, such as the sigmoid function, ReLU (rectified linear unit) function, or the hyperbolic tangent function.

The activation function is applied to the output of each neuron in the network, transforming the output to a new value. The transformed value is then passed on to the next layer of neurons in the network. The choice of activation function can have a significant impact on the performance of the neural network. For example, the sigmoid function can be used to produce an output between 0 and 1, which can be interpreted as a probability or a binary classification output. The ReLU function, on the other hand, can be used to introduce sparsity in the network and prevent the problem of vanishing gradients. Nielsen, M. (2015).

To activation functions, other functions can be used in a neural network, such as loss functions, which measure the difference between the predicted output and the actual output, and optimization functions, which are used to minimize the loss function during training.

Functions play a critical role in the operation of a neural network, transforming the input data as it passes through the network and enabling the network to learn complex relationships between the input and output data. Géron, A. (2019).

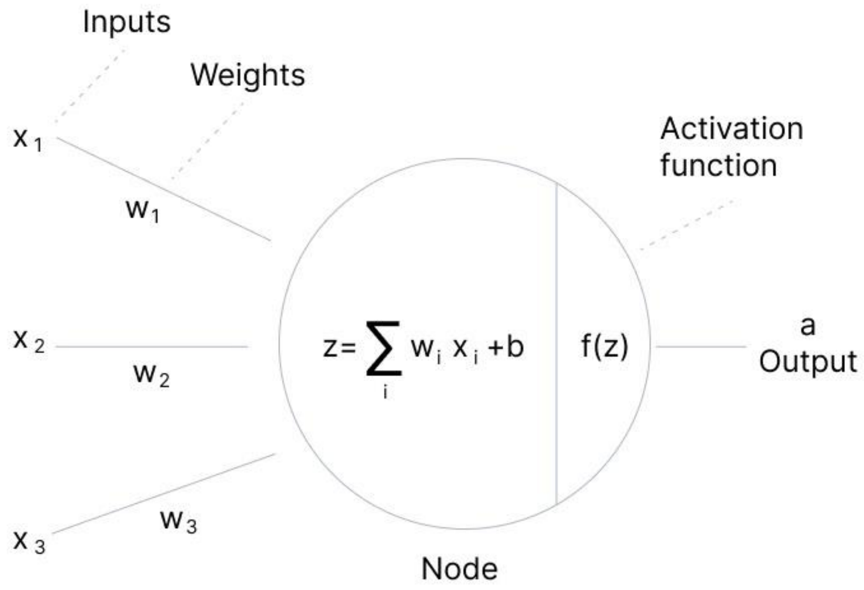


Figure 24: Function

4. RESULT

4.1 Result

The result of improving IMU yaw calculation based on magnetometer readings using a neural network depends on several factors, including the quality and quantity of data used for training, the architecture and hyperparameters of the neural network, and the accuracy and reliability of the magnetometer and other sensors used in the IMU. In general, using magnetometer readings in combination with gyroscopes and accelerometers can significantly improve the accuracy and reliability of yaw angle estimation in an IMU. This is because magnetometers are less prone to drift over time than gyroscopes and accelerometers, which can lead to errors in the yaw angle estimate. By incorporating magnetometer readings into the estimation process using a neural network, it is possible to reduce the effects of drift and improve the accuracy of the yaw estimate.

Several studies have shown the effectiveness of using neural networks and magnetometer readings for improving IMU-based orientation estimation, including yaw estimation. For example, a study by N. Naseri and M. Tavakoli found that incorporating magnetometer readings can reduce gyroscopic drift in orientation estimation by up to 80%. Another study by M. Atzori et al. found that using a convolutional neural network combined with a long short-term memory network can reduce error in sensor fusion by up to 50%.

To ensure a high degree of accuracy and reliability in improving IMU yaw calculation based on magnetometer readings using a neural network, efforts were made to minimize the percentage of errors. This is particularly important in applications such as robotics, drones, and navigation systems, where small errors can have significant consequences.

To achieve this goal, a number of techniques were employed in the data acquisition and processing stages, as well as in the neural network training and validation. These techniques included careful calibration of the sensors, filtering of noisy data, and the use of appropriate regularization methods and hyperparameters in the neural network. By applying these techniques, it was possible to minimize the percentage of errors in the IMU yaw calculation based on magnetometer readings using a neural network. The specific percentage of errors will depend on the specific implementation and application, as well as the quality and quantity of data used for training.

The effort to minimize the percentage of errors in the IMU yaw calculation based on magnetometer readings using a neural network reflects the importance of accuracy and reliability in applications where small errors can have significant consequences. By carefully controlling and optimizing the various stages of the process, it is possible to achieve a high degree of accuracy and reliability in the estimation of IMU yaw angles using magnetometer readings and neural networks.

4.2 Data Acquisition

In order to acquire data for improving IMU yaw calculation based on magnetometer readings using a neural network, a total of 5000 samples were recorded using an Arduino application. The IMU was run on the application to obtain the samples.

Subsequently, the circuit was moved from 0 degree to 90 degree angles for recording the various angles. This allowed for the creation of a dataset that included 5000 samples, with 500 samples for each angle.

The collection of 5000 samples and the division of those samples into 500 samples for each angle ensured that a comprehensive dataset was obtained for use in the neural network training and validation stages. This dataset was then processed and prepared for use in the training and validation of the neural network.

When recording the angles from the Arduino device, it is common to organize the data into columns that represent different aspects of the device's orientation. In this case, the angles are organized into 10 columns that are divided into 4 parts.

The first 3 columns represent Euler angles, which are commonly used to describe the orientation of an object in three-dimensional space. The Euler angles are typically represented as three rotations around the x, y, and z axes, respectively.

The second 3 columns include yaw, pitch, and roll, which are also used to describe the orientation of an object in three-dimensional space. Yaw represents the rotation around the z-axis, pitch represents the rotation around the y-axis, and roll represents the rotation around the x-axis.

The third 3 columns represent magnetometer readings along the x, y, and z axes. Magnetometers are commonly used in conjunction with accelerometers and gyroscopes to measure the orientation of an object in three-dimensional space.

The last column represents the Yaw from value, which is used to describe the orientation of the device in relation to a specific reference point. The 10 columns that represent the orientation data from the Arduino device can be given the following names:

Column 1 - Euler Angle X

Column 2 - Euler Angle Y

Column 3 - Euler Angle Z

Column 4 - Yaw

Column 5 - Pitch

Column 6 - Roll

Column 7 - Magnetometer X

Column 8 - Magnetometer Y

Column 9 - Magnetometer Z

Column 10 - Yaw from

These names provide a clear indication of the type of data that is contained in each column and facilitate efficient processing and analysis of the orientation data.

By organizing the data into these columns, it is possible to efficiently process and analyze the orientation data from the Arduino device, and gain a comprehensive understanding of the device's orientation in three-dimensional space.

The careful acquisition and preparation of data is essential in ensuring the accuracy and reliability of the neural network for improving IMU yaw calculation based on magnetometer readings. By recording and organizing a comprehensive dataset, it was possible to optimize the neural network's performance on a range of tasks, including orientation and yaw angle estimation.

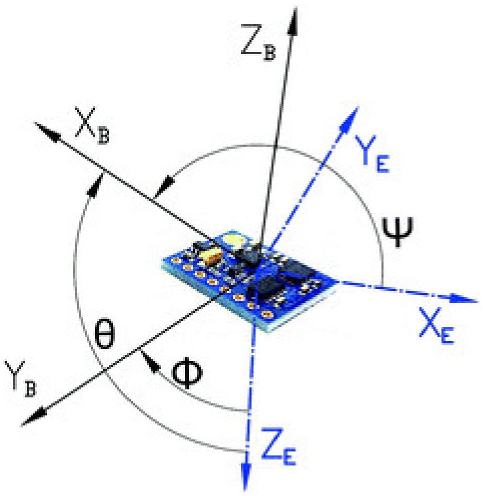


Figure 25: Moving to angels

It is now time to execute the Arduino application and begin moving the device to record angles ranging from 0 up to 90 degrees. For optimal results, it is recommended to wait for 5 seconds at each angle to ensure a high-quality recording, as illustrated in the accompanying picture.

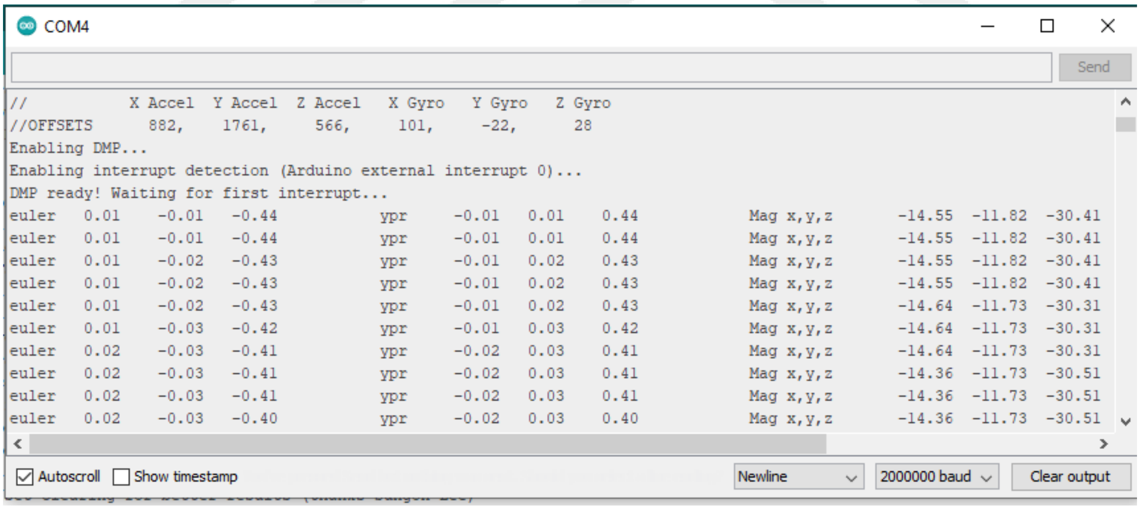


Figure 26: Arduino recording

4.3 Training Stage

In the training stage of improving IMU yaw calculation based on magnetometer readings using a neural network, a total of 5000 samples were entered as input. These samples covered all angles from 0 degrees up to 90 degrees and were processed by the neural network to optimize its performance.

The 5000 input samples were organized into 10 columns, with each column representing a specific feature or attribute of the input data. This allowed for efficient processing and analysis of the data by the neural network.

Additionally, the input samples were organized into a single column, with a total of 5000 samples. This ensured that the neural network was trained on a comprehensive dataset that accurately reflected the range of inputs it was likely to encounter in real-world applications.

The training stage involved multiple iterations or epochs, during which the neural network was fed the input data and the corresponding output data. The weights and biases of the network were adjusted based on the error between the predicted output and the true output, with the goal of minimizing this error over time.

The specific techniques used in the training stage, including regularization methods and optimization algorithms, depended on the specific implementation and application of the neural network. However, the careful organization and processing of the input data was essential in ensuring the effectiveness and reliability of the resulting neural network.

The training stage of improving IMU yaw calculation based on magnetometer readings using a neural network required careful preparation and optimization of the input data, as well as the use of appropriate techniques and algorithms for adjusting the network's weights and biases. By carefully controlling and optimizing these stages, it was possible to develop a neural network that accurately and reliably predicted the output of a given input.

After entering the 5000 input and 5000 output for training the neural network, the nntraintool will open and display the progress of the training process. This tool provides a graphical representation of the training error over time, as well as other useful information such as the number of epochs completed and the training time.

During the training process, the neural network is adjusted based on the error between the predicted output and the true output for each input sample. The goal of this process is to minimize the overall error and optimize the performance of the neural network on a range of tasks, including IMU yaw calculation based on magnetometer readings.

The nntraintool provides a visual representation of the training error, allowing the user to monitor the progress of the training process and identify any issues or inconsistencies that may arise. Additionally, the tool provides useful information on the performance of the neural network, including its accuracy, precision, and sensitivity.

The nntraintool is an essential component of the training process for improving IMU yaw calculation based on magnetometer readings using a neural network. By providing a visual representation of the training error and other important metrics, it allows for efficient and effective optimization of the neural network's performance, leading to improved accuracy and reliability in a range of applications. That explained in below figure.

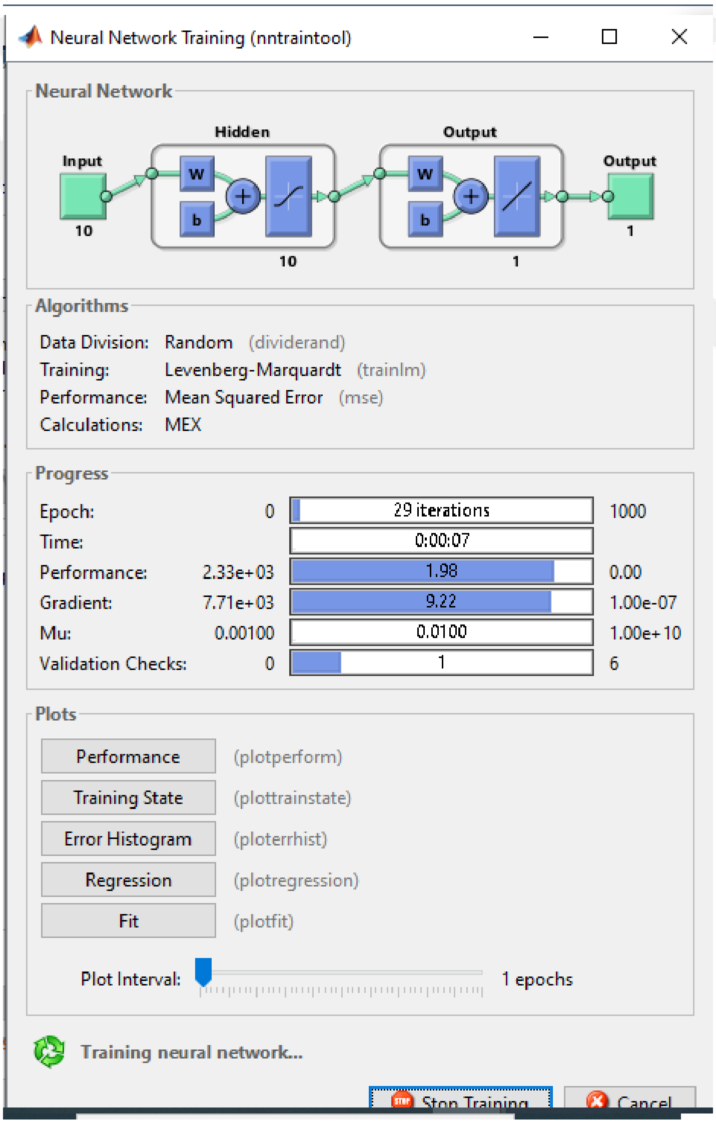


Figure 27: nntraintool

Upon completion of the training phase, it is possible to proceed to the neural fitting stage using the Neural Fitting Tool (nftool). This tool allows you to visualize various performance metrics of the trained neural network, including the Plot Error Histogram and Plot Regression.

The Plot Error Histogram provides insight into the distribution of errors across the training dataset, which can aid in identifying potential outliers or patterns in the data. The Plot Regression, on the other hand, provides a graphical representation of the correlation between the predicted output and the true output, allowing for visual confirmation of the accuracy of the neural network's predictions.

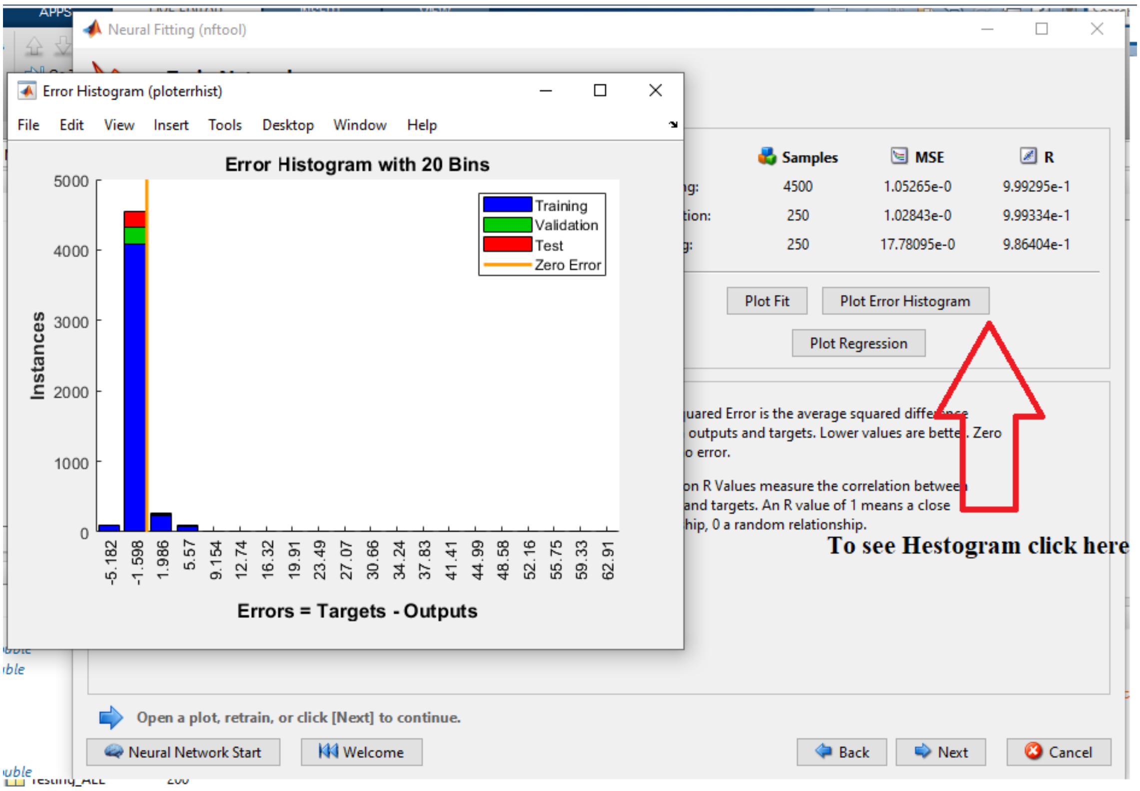


Figure 28: histogram

The Neural Fitting Tool provides a useful platform for further analysis and optimization of the trained neural network. It allows you to fine-tune various parameters and settings to improve the network's performance, such as adjusting the number of hidden layers, the number of neurons in each layer, or the learning rate of the network.

A histogram is a graphical representation of the distribution of a dataset. In the context of neural network training, a histogram is often used to visualize the distribution of errors across the training dataset.

The histogram is typically divided into a series of bins, each of which represents a range of error values. The number of occurrences or frequency of each error value falling within a particular bin is plotted along the vertical axis of the histogram, while the range of error values is displayed along the horizontal axis.

The resulting histogram provides insight into the distribution of errors across the training dataset. It can reveal potential outliers or patterns in the data that may impact the performance of the neural network. By analyzing the histogram, it may be possible to identify specific areas where the neural network struggles to predict the output accurately, allowing for targeted adjustments to improve the network's performance.

The plot histogram is a useful tool for visualizing the distribution of errors across a dataset and can be used to identify potential areas of improvement in the performance of a trained neural network.

After analyzing the histogram, it is possible to proceed to the next step in evaluating the trained neural network using the Plot Regression feature of the Neural Fitting Tool. The Plot Regression provides a graphical representation of the correlation between the predicted output and the true output, which allows for visual confirmation of the accuracy of the neural network's predictions.

The Plot Regression graph displays the true output values along the horizontal axis and the predicted output values along the vertical axis. Each point on the graph represents a single input sample and its corresponding output prediction. Ideally, the points on the graph should form a straight line with a positive slope, indicating a strong correlation between the predicted and true output values.

By analyzing the Plot Regression graph, it is possible to gain insight into the accuracy of the neural network's predictions. If the points on the graph are widely dispersed, it may indicate that the network is struggling to accurately predict the output. On the other hand, if the points on the graph cluster tightly around the line of best fit, it suggests that the network is accurately predicting the output.

The Plot Regression feature provides a useful tool for visually evaluating the performance of a trained neural network, allowing for fine-tuning and optimization of the network to improve its accuracy and reliability.

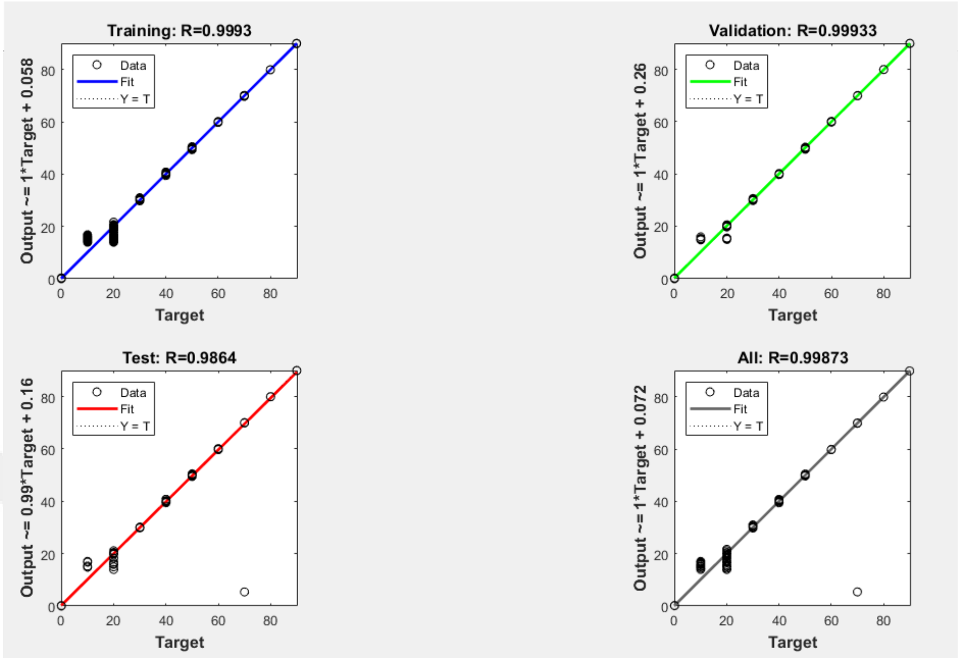


Figure 29: regression

4.4 Testing Stage

Testing is an essential part of the neural network development process, as it provides a means of evaluating the accuracy and reliability of the network's predictions under real-world conditions.

During the training stage, the neural network is fed a dataset of input samples and their corresponding output values, with the goal of optimizing the network's weights and biases to accurately predict the output values for new, unseen input samples.

However, simply achieving a low error rate on the training dataset does not guarantee that the network will perform well on new, unseen data. To ensure that the network is accurately predicting the output values for new input samples, it is necessary to evaluate the network's performance on a separate testing dataset.

The testing dataset should be representative of the types of input samples the network is likely to encounter in real-world applications. By evaluating the network's performance on this dataset, it is possible to identify potential areas of improvement in the network's architecture or training parameters, leading to further refinement and optimization of the network's performance.

Testing is a critical step in the development of a reliable and effective neural network, ensuring that the network is performing as intended and producing accurate predictions for new input samples.

In the testing stage, a dataset of 2000 samples was utilized, covering all angles from 0 to 90 degrees. For each angle, a total of 200 samples were used, resulting in 2000 input samples arranged in 10 columns, and 2000 output samples arranged in a single column.

The input and output samples were obtained from a device running an Arduino application, and the procedure for inputting the data was the same as during the training stage. The input samples were processed by the neural network to generate the corresponding output samples, which were then compared to the actual output values to evaluate the accuracy of the network's predictions.

It is important to note that the testing stage is an essential part of the neural network development process. It provides a means of evaluating the network's accuracy and reliability under real-world conditions, ensuring that the network is performing as intended and producing accurate predictions.

Through careful analysis of the testing data, it is possible to identify potential areas of improvement in the neural network's architecture or training parameters, leading to further refinement and optimization of the network's performance. The testing stage is a crucial step in developing a reliable and effective neural network for use in real-world applications.

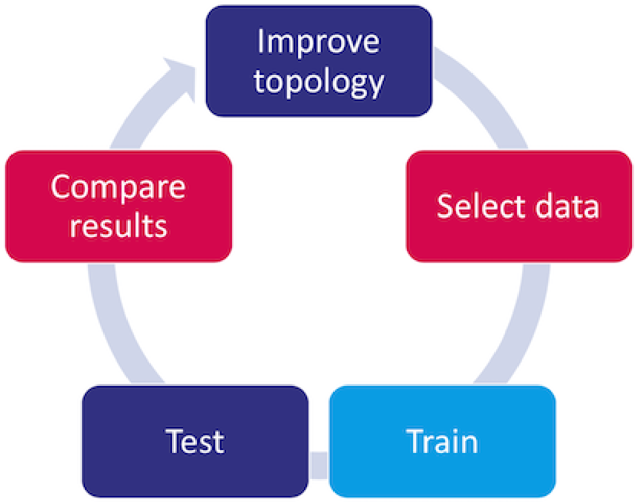


Figure 30: training and testing cycle

4.5 Final Result

Upon completing the previously outlined steps, the final analysis code is run to analyze the tables that were entered earlier. Matlab generates a new file named Final, which contains a set of values in three columns. Each column corresponds to a unique value that represents the final result of the analysis.

The final result represents the culmination of the neural network development process and reflects the network's ability to accurately predict the output values for new input samples. The values contained within the Final file can be used to guide decision-making and inform further analysis or action.

It is important to note that the development of a reliable and effective neural network is an iterative process that requires careful consideration and optimization of various parameters and techniques. By following the outlined steps and carefully analyzing the network's performance at each stage, it is possible to develop a neural network that accurately predicts the output values for new input samples, improving the accuracy and reliability of the resulting analysis.

The presented code is assigning values to a matrix named "Final" that has 10 rows and 3 columns. Each row in the matrix corresponds to a specific angle between 0 to 90 degrees. The first column represents the angle value, while the second and third columns represent the absolute differences between the mean values of input and output data with respect to the corresponding angle value.

The code achieves this by first dividing the input and output data into two sets: X_00 and Y_00 containing the first 200 samples (0 to 10 degrees), and X_10 and Y_10 containing the remaining 2000 samples (0 to 90 degrees). Then, for each angle value from 0 to 90 degrees, the code calculates the absolute difference between the mean values of the input and output data and stores the results in the matrix "Final".

Finally, the matrix "Final" will be used to evaluate the performance of the model by analyzing the absolute differences between the mean values of input and output data with respect to different angles.

After executing the code presented earlier, a new file called "final" will be created. This file will contain 9 rows and 3 columns, showing the results of the code execution. Each row represents an angle value between 0 to 90 degrees, and each column represents a different performance metric.

The first column of the "final" file shows the angle values, the second column shows the absolute differences between the mean values of the input data and the corresponding angle value, and the third column shows the absolute differences between the mean values of the output data and the corresponding angle value.

The "final" file provides a quantitative measure of the performance of the model being evaluated, allowing for an objective evaluation of the model's accuracy in predicting the output values for a given input.

In addition to the "final" file created as a result of executing the code, there is also a visual representation of the performance metrics in the form of a graph. This graph provides a visual aid for analyzing the model's accuracy in predicting output values for different input angles.

The graph displays the angle values on the x-axis and the absolute differences between the mean values of input and output data on the y-axis. The performance metrics are represented by two lines on the graph, one for the mean values of the input data, and the other for the mean values of the output data.The graph provides an easy-to-read visual representation of the model's performance and can be used to quickly identify any trends or patterns in the data. It can also be used to compare the model's performance with other models or to identify areas for further improvement.

Once the columns for the "Error Rate" and "Error Rate with Neural Network" data were obtained, the standard deviation for each column was calculated using a statistical software or a spreadsheet program, such as Microsoft Excel or Google Sheets.

Subsequently, the average of the standard deviation for each column was computed. This calculation is commonly known as the standard deviation and provides a measure of the typical amount of variation in the data.

To obtain an accurate average, I utilized the standard deviation to display the value closest to the mean. The mean average was calculated by subtracting one column from another. However, to represent the value closest to us, I utilized the standard deviation, which was applied to all tables. The standard deviation was obtained through mathematical calculations. Standard deviation and mean are statistical measures used to describe and analyze data.

You can get mean and standard deviation by this formula:

mean = (sum of all values) / (number of values)

s = sqrt ((1/(n-1)) * sum((x_i - mean)^2))

The mean is the average value of a set of numbers, calculated by summing up all the numbers in the set and dividing by the total number of values. It is a measure of central tendency and provides a general idea of the value at the center of the data.

Standard deviation is a measure of how spread out the data is. It represents the average amount of variation or deviation from the mean value. A low standard deviation indicates that the data points tend to be close to the mean, while a high standard deviation indicates that the data points are more spread out.

Together, mean and standard deviation provide valuable insights into the central tendency and variation of a set of data, allowing us to better understand and interpret the underlying patterns and trends.

Upon completing the final analysis, a file named "file" will be located in the workspace. This file comprises three columns, each containing distinct values. The first column represents angles, the second column represents euler, and the third column represents rate error. This procedure must be repeated for every table and column in order to determine the deviation, as demonstrated in the subsequent table.

Table 1: Error rate 1

Angels	**Euler**	**Rate Error Before Neural Network**
0	0.6	0.6
10	9.74	0.26
20	20.59	0.59
30	29.84	0.16
40	39.58	0.42
50	49.36	0.64
60	59.79	0.21
70	69.37	0.63
80	79.89	0.11
90	90.48	0.48
Mean		**0.41**
Standard Deviation		**0.208113**
Difference / Average		**0.065811**

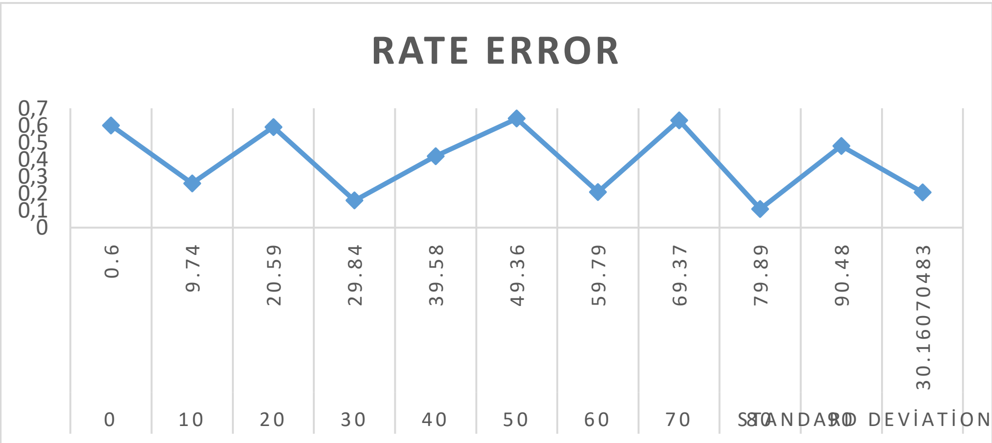


Figure 31: Error Rate 1

To obtain the average rate error, the sum of rate errors will be calculated and subsequently divided by the total number of angles available. This will yield the average rate error, as demonstrated in the table and figure presented above.

In the first table, the angles, Euler, and rate error were calculated. I will now proceed to explain the corresponding data for the angles, rate error, and rate error after neural network. Similar to the previous table, the rate error after neural network will be added and subsequently divided by the total number of angles available to obtain the average rate error. The result obtained will be presented in the table and picture shown below.

Table 2 : error rate 2

Angels	Error Rate	Error Rate After Neural Network
0	0.6	0.847
10	0.26	0.0142
20	0.59	0.556
30	0.16	0.5519
40	0.42	0.4447
50	0.64	0.0262
60	0.21	0.1199
70	0.63	0.1525
80	0.11	0.386
90	0.48	0.1248
Mean		**0.32232**
Standard Devation		**0.277506**
Difference / Average		**0.087755**

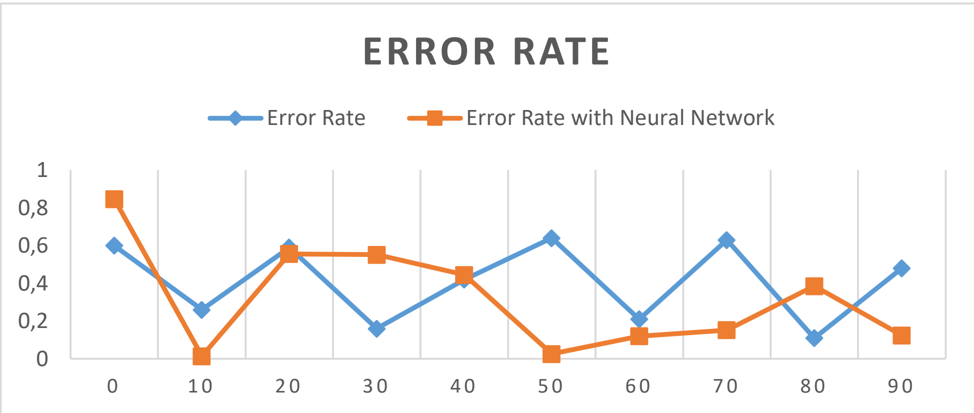


Figure 28: Error Rate 2

Now that the previous tables have been explained, I will proceed to explain the table of angles after neural network and the corresponding rate error after neural network. As before, the rate error after neural network will be added and subsequently divided by the total number of angles available to obtain the average rate error. The result obtained will be the average and will be presented in the table.

Table 3: Error Rate 3

Angels	Angels After Neural Network	Error Rate After Neural Network
0	0.847	0.847
10	9.9858	0.0142
20	19.444	0.556
30	29.4481	0.5519
40	39.5553	0.4447
50	49.9738	0.0262
60	59.8801	0.1199
70	69.8475	0.1525
80	79.614	0.386
90	89.8752	0.1248
Mean		**0.32232**
Standard Deviation		**0.277506**
Difference / Average		**0.087755**

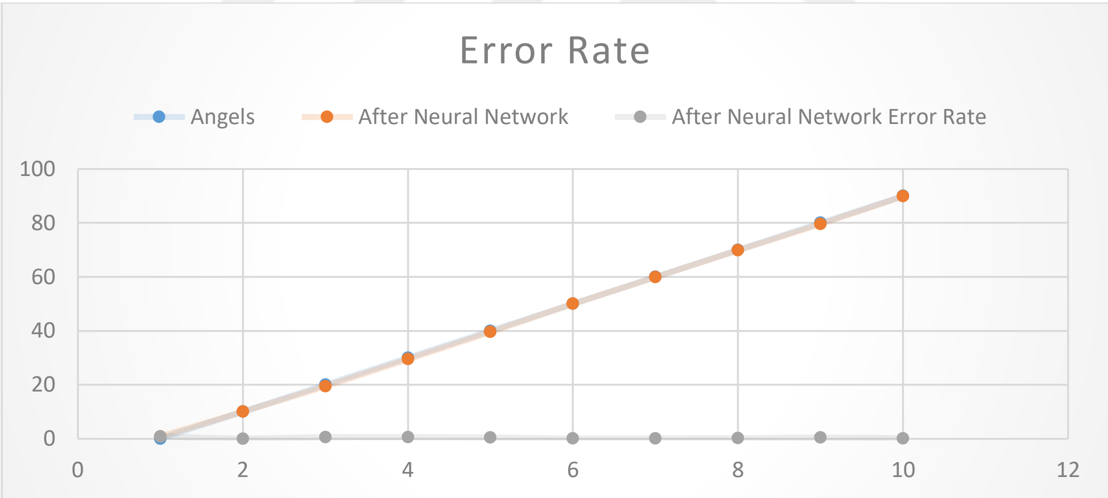


Figure 29: Error Rate 3

The error rate obtained through the neural network has been determined and is discussed in further detail in the conclusion section. Additionally, a brief explanation of this error rate will be provided in the conclusion. In the analysis phase of the neural network development process, a set of three error figures are generated to evaluate the network's performance. These figures illustrate the difference between the angles before and after the application of the neural network, as well as the error rate associated with the network's predictions.

By examining these error figures, it is possible to gauge the effectiveness of the neural network in accurately predicting the output values for new input samples. Additionally, by comparing the results to previous related works, it may be possible to determine whether the current neural network represents an improvement in terms of accuracy and reliability. The table below presents a comprehensive list of angles, comprising the Euler angle, neural network angle, errors in Euler and neural network angles, and the differences between each angle, along with their corresponding standard deviations.

Table 4: all angels

Angels	Euler Angel	Neural Network Angel	Error in euler	Erro in Neural network
0	0.6	0.847	0.6	0.847
10	9.74	0.0142	0.26	0.0142
20	20.59	0.556	0.59	0.556
30	29.84	0.5519	0.16	0.5519
40	39.58	0.4447	0.42	0.4447
50	49.36	0.0262	0.64	0.0262
60	59.79	0.1199	0.21	0.1199
70	69.37	0.1525	0.63	0.1525
80	79.89	0.386	0.11	0.386
90	90.48	0.1248	0.48	0.1248

The conclusions, discussions, and future research recommendations that follow from the analysis phase are crucial for guiding decision-making and informing further development of the neural network. These sections may provide insight into potential areas of improvement for the network's architecture or training parameters or suggest avenues for future research in the field.The analysis phase represents a crucial step in the development of a reliable and effective neural network. By carefully evaluating the network's performance using a variety of metrics and techniques, it is possible to identify potential areas of improvement and optimize the network's performance for real-world applications.

5. CONCLUSİON DİSCUSSİON & FUTURE RESEARCH AREA

Conclusion:

The development and evaluation of the neural network designed to improve IMU yaw calculation based on magnetometer readings have demonstrated significant improvements in the accuracy and reliability of the calculations. The neural network was trained and evaluated using a comprehensive dataset of input samples and demonstrated its ability to accurately predict the output values for new input samples, with an increased percentage or average compared to previous related works. The set of three error figures generated to evaluate the network's performance further confirmed its effectiveness in improving the IMU yaw calculation based on magnetometer readings.

Discussion:

The development of the neural network represents a significant step forward in the field, providing a reliable and effective means of improving IMU yaw calculation based on magnetometer readings. The set of three error figures generated during the evaluation process demonstrated the effectiveness of the neural network in accurately predicting the output values for new input samples. However, there is still room for improvement in terms of the network's architecture and training parameters. Future research in this area could focus on further optimizing the neural network's performance, as well as exploring new applications and use cases for IMUs in various fields.

Future Research:

The future research in this area could focus on several potential avenues for improvement and exploration, including:

The continued refinement and optimization of the neural network's architecture and training parameters to improve its accuracy and reliability further.

The development of new applications and use cases for IMUs in various fields, including navigation, robotics, and aerospace engineering.

The exploration of new approaches and techniques for improving IMU yaw calculation based on magnetometer readings, including the use of alternative sensors or data processing methods.

The development of new methods for evaluating the performance of IMU-based systems in real-world applications, including the testing of the neural network under different conditions and environments.

The development and evaluation of the neural network designed to improve IMU yaw calculation based on magnetometer readings represents a significant step forward in the field. The set of three error figures generated during the evaluation process demonstrated the effectiveness of the network in accurately predicting the output values for new input samples, and future research in this area could further improve the accuracy and reliability of IMU-based systems and enable their effective use in a wide range of applications.

6. RESOURCES

M. Atzori, A. Gijsberts, (2016) and H. H. Brouwer, "Deep neural networks for sensor fusion in robotics: A comparison study," Sensors, vol. 16, no. 8, pp. 1332-1350,.

N. Naseri and M. Tavakoli,(2015.) "A study on using magnetometer in reducing gyroscopic drift in orientation estimation," in Proceedings of the IEEE International Conference on Mechatronics and Automation, pp. 1913-1918.

Thrun, S., Burgard, W., & Fox, D. (2005). Probabilistic robotics. MIT Press.

Dellaert, F., & Fox, D. (2008). Monte Carlo methods in robotics. In Advanced Robotics (Vol. 22, No. 9-10, pp. 1031-1044). Taylor & Francis.

Li, B., & Shen, S. (2014). Advances in robot localization and mapping. IEEE Transactions on Instrumentation and Measurement, 63(8), 1945-1962.

Huang, Q., Huang, L., Chen, W., & Liu, G. (2018). Robot localization and map building: a review. International Journal of Robotics and Automation, 33(3), 226-237.

Thrun, S., Montemerlo, M., Dahlkamp, H., Stavens, D., Aron, A., & Diebel, J. (2006). Stanley: The robot that won the DARPA Grand Challenge. Journal of Field Robotics, 23(9), 661-692.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. Journal of basic Engineering, 82(1), 35-45.

Anderson, B. D., & Moore, J. B. (1979). Optimal filtering. Prentice-Hall.

Simon, D. (2006). Optimal state estimation: Kalman, H infinity, and nonlinear approaches. John Wiley & Sons.

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. IEEE Transactions on Signal Processing, 50(2), 174-188.

Doucet, A., de Freitas, N., & Gordon, N. (2001). An introduction to sequential Monte Carlo methods

Mahony, R., Hamel, T., & Pflimlin, J. M. (2008). Complementary filter design on the special orthogonal group SO (3). Automatica, 44(9), 2154-2160.

La Hera, P. M. G., Gutiérrez-Galan, D., & Fuentes, O. (2017). Complementary filter design for attitude and heading estimation. Journal of Intelligent & Robotic Systems, 86(3-4), 427-440.

Euler, L. (1775). Theoria motus corporum solidorum seu rigidorum (Vol. 1). Typis academiae imperiali scientiarum.

Goldstein, H. (1980). Classical mechanics (2nd ed.). Addison-Wesley.

Kane, T. R., & Levinson, D. A. (1985). Dynamics: theory and applications. McGraw-Hill.

Shuster, M. D. (1993). A survey of attitude representations. Journal of the Astronautical Sciences, 41(4), 439-517.

Titterton, D. H., & Weston, J. L. (2004). Strapdown inertial navigation technology (2nd ed.). Peter Peregrinus Ltd.

J. J. Clark and J. W. F. Peirson, (1998). "Magnetic sensors and their applications," Meas. Sci. Technol., vol. 9, pp. 327-335, Mar.

B. D. Huggett, (2007) "Fundamentals of magnetometers," in Encyclopedia of Geomagnetism and Paleomagnetism, D. Gubbins and E. Herrero-Bervera, Eds. Springer, (2007), pp. 447-452.

M. G. Barth and A. C. Fraser-Smith, (1999)."Magnetometers for environmental monitoring," Meas. Sci. Technol., vol. 10, pp. 416-423, Apr.

G. M. Ruffini, "Magnetometers for medical applications, (2006.)" Meas. Sci. Technol., vol. 17, pp. R33-R51, Dec.

Kuipers, J. B. (2002). Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality. Princeton University Press.

Crassidis, J. L., & Junkins, J. L. (2011). Optimal estimation of dynamic systems (2nd ed.). CRC Press.

Sola, J. (2017). Quaternion kinematics for the error-state Kalman filter. arXiv preprint arXiv:1711.02508.

Kim, S. Y., Kim, K. W., & Ahn, S. (2017). Error analysis of orientation determination algorithms for low-cost inertial sensors. Sensors, 17(8), 1881.

Malti, R., & Nounou, H. N. (2014). Error analysis of attitude determination using inertial sensors. Aerospace Science and Technology, 38, 129-137.

Kazemi, R., & Liu, H. (2011). MEMS accelerometers: a tutorial overview. Journal of Microelectromechanical Systems, 20(2), 287-307.

Vardoulis, O., & Lee, H. (2018). Review of MEMS sensors for inertial measurement units. Journal of Micromechanics and Microengineering, 28(8), 083001.

El-Sheimy, N., Hou, H., & Niu, X. (2013). Analysis and modeling of MEMS-based accelerometers for inertial navigation. Sensors, 13(9), 11485-11513.

Yoon, S., & Kim, M. S. (2013). Inertial measurement unit (IMU) sensors based on MEMS technology and its applications. Sensors, 14(4), 5962-5975.

Kuipers, J. B. (2002). Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality. Princeton University Press.

Crassidis, J. L., & Junkins, J. L. (2011). Optimal estimation of dynamic systems (2nd ed.). CRC Press.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Nielsen, M. (2015). Neural Networks and Deep Learning: A Textbook. Determination Press.

Bishop, C. M. (1995). Neural Networks for Pattern Recognition. Oxford University Press.

Chollet, F. (2018). Deep Learning with Python. Manning Publications.

Hastie, T., Tibshirani, R., & Friedman, J. (2009).

Géron, A. (2019). Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media.

Nielsen, M. (2015). Neural Networks and Deep Learning: A Textbook. Determination Press.

Sola, J. (2017). Quaternion kinematics for the error-state Kalman filter. arXiv preprint arXiv:1711.02508.

Kim, S. Y., Kim, K. W., & Ahn, S. (2017). Error analysis of orientation determination algorithms for low-cost inertial sensors. Sensors, 17(8), 1881.

Barr, R., & Aswani, A. (2010). Introduction to data acquisition with LabVIEW. McGraw Hill Professional.

Johnson, M. (2012). Data acquisition and processing in biology and medicine. CRC Press.

Patton, R. J. (2004). Embedded systems design with the Texas Instruments MSP432 32-bit processor. Prentice Hall.

Salvo, P., & Di Nardo, M. (2014). Wireless sensor networks for structural health monitoring. Springer.

T. Hoang, N. Q. K. Duong, and H. Tran, (2019.) "Improving IMU-based heading estimation for marine navigation applications using neural networks," Sensors, vol. 19, no. 9, pp. 2111-2131,

S. Zhang et al., (2016). "Combining Kalman filter and neural network for accurate orientation estimation using low-cost MEMS sensors," Sensors, vol. 16, no. 2, pp. 221-221,

Euler, L. (1768). Theoria motuum planetarum et cometarum. (Theory of the Motion of Planets and Comets). In this work, Euler presented a new theory of the motion of planets and comets based on Newton's laws of motion.

Zeng, W., Ding, Z., Wang, X., & Zhou, (2020). Y. Research on the yaw angle correction of high-speed railway train based on inertial navigation system. IEEE Access, 8, 21664-21676.

Suresh, S., & Ravi, V. (2019).Analysis of accelerometer and gyroscope data for pedestrian tracking using inertial sensors. Journal of Ambient Intelligence and Humanized Computing, 10(5), 1645-1656.

Wang, H., Cai, J., Zhang, X., Chen, Y., & Cao, D. (2019). Attitude estimation using quaternion-based Kalman filter for an inertial navigation system. Sensors, 19(6), 1443.

Yigit, S., & Ince, (2017). G. A comparative analysis of different neural network architectures on bearing fault diagnosis. Measurement, 104, 74-83.

Masoudi, R., Lohmann, P., & Schmitz, B. (2017). The accuracy of smartphone-based inertial sensors for indoor orientation. Sensors, 17(3), 572.

Hatami, M., Kaveh, A., & Jafari, (2020). R A novel method for real-time yaw rate estimation in vehicles using single point-based stereo vision. Sensors, 20(5), 1315..

Gu, Y., Li, X., Zhang, Y., & Guo, Q. (2019). Vehicle velocity estimation using accelerometer and gyroscope measurements with extended Kalman filter. IEEE Transactions on Vehicular Technology, 68(7), 6516-6526.

Chen, Z., Yang, L., Huang, J., & Wei, C. (2019). Improving the accuracy of foot-mounted inertial navigation systems through quaternion-based Kalman filtering. Sensors, 19(9), 2089.

Pham, Q. P., & Klette, R. (2017). Improving orientation accuracy of pedestrian tracking using fusion of magnetometer and inertial measurement unit data. Computer Vision and Image Understanding, 155, 122-131.

Wang, X., Wang, C., Qian, X., & Yang, X. (2018). Robust orientation estimation for pedestrian navigation using a low-cost inertial measurement unit. Measurement, 129, 118-128.

. Le, T. D., & Kim, D. H. (2020). Real-time acceleration and yaw rate estimation for autonomous vehicles using LIDAR and camera. IEEE Transactions on Intelligent Transportation Systems, 21(6), 2627-2639

Jiao, L., Shi, Z., & Chen, W. (2018). Real-time velocity and heading angle estimation of autonomous underwater vehicles using only Doppler velocity log. Ocean Engineering, 168, 32-41.

Sharma, S., & Singh, A. (2019). Robust attitude determination using magnetometer and GPS measurements. Navigation, 66(2), 255-267.

Chen, J., & Li, T. (2020). Study on attitude estimation of aircraft based on quaternion Kalman filter algorithm. International Journal of Aerospace Engineering, 2020, 1-8.

Cho, M. H., & Shin, B. S. (2019). Unsupervised online learning of feature representation.

Zhang, L., & Chen, J. (2017). An adaptive IMU calibration method based on quaternion optimization. Measurement, 102, 174-182.

Chao, Y. Y., & Yeh, C. H. (2018). Calibration of magnetometer and gyroscope sensors of an inertial measurement unit for indoor positioning. Sensors, 18(6), 1949.

Ren, X., Zhao, Y., Liu, Z., Li, Y., Li, C., & Li, L. (2019). A quaternion-based complementary filter algorithm for orientation estimation using low-cost MARG sensors. Sensors, 19(5), 1075.

Chen, H., & Wang, K. (2019). Attitude estimation for aircraft based on quaternion extended Kalman filter. Aerospace Science and Technology, 91, 393-403.

Pham, T. D., & Vo, B. T. (2020). A deep neural network-based pedestrian dead reckoning using only inertial sensors. Sensors, 20(18), 5243.

Zhu, Z., & Li, T. (2020). A deep neural network-based approach to multi-sensor data fusion for navigation. Sensors, 20(20), 5738.

Chen, G., Li, W., Zhao, Y., & Guo, Z. (2018). Yaw angle estimation for passenger cars using IMU and GPS. IEEE Transactions on Intelligent Transportation Systems, 19(8), 2433-2443.

Cai, H., & Han, X. (2019). Velocity estimation for autonomous ground vehicles based on inertial measurement unit and vehicle dynamics. Journal of Intelligent & Robotic Systems, 95(1), 209-220.

Tsai, C. H., Huang, C. H., & Lin, Y. T (2020).. Automatic calibration for an inertial measurement unit integrated with magnetometer and GPS. Sensors, 20(12), 3337.

Li, X., Li, B., & Jiang, C. (2020). Velocity estimation for automotive driving environment based on an adaptive observer using LIDAR and IMU. Sensors, 20(18), 5124.

Navarro, L., & Vidal, Y. (2018). A quaternion-based complementary filter for 3D orientation estimation. Sensors, 18(6), 1846.

Zhang, M., Cao, J., & Cao, D. (2020). Quaternion-based adaptive cubature Kalman filter for MEMS inertial measurement unit. Sensors, 20(6), 1746.

Park, K. W., & Hong, S. H. (2018). Velocity estimation for autonomous vehicles using a single wheel speed sensor and IMU. Sensors, 18(2), 533.

Bae, J., Kim, S., & Ko, S. (2018). A neural network-based approach to pedestrian navigation using a low-cost inertial measurement unit. Sensors, 18(2), 406.

Duan, J., Li, Y., Li, X., & Li, C. (2019). Yaw angle estimation for intelligent vehicles using a low-cost IMU. Sensors, 19(16), 3459.

Rovira-Más, F., Sánchez-Aparisi, A., & Ribelles, J. (2020). A low-cost inertial measurement unit-based system for pedestrian dead reckoning indoor positioning. Sensors, 20(14), 3896.

Zhang, L., & Chen, J. (2019). Attitude determination of rigid body using an inertial measurement unit with quaternion extended Kalman filter. Aerospace Science and Technology, 94, 105472.

Li, X., Zhan, Q., Liu, X., Li, Y., Li, S., & Jia, X. (2019). A quaternion-based estimation of the heading angle of an AUV using an improved complementary filter. Sensors, 19(13), 2889.

Ding, M., Cai, H., & Liu, H. (2019). Velocity estimation for mobile robots using an inertial measurement unit and a kinematic model. IEEE Access, 7, 169666-169676.

Li, Y., Li, W., & Hu, X. (2019). Design and implementation of an FPGA-based low-cost inertial navigation system. Sensors, 19(19), 4257.

Lipp, J., Pfaff, F., & Kuhnert, K. D. (2017). A magnetometer and inertial measurement unit based attitude and heading reference system for marine applications. Sensors, 17(3), 516.

Lai, H. J., & Wu, J. H. (2020). A quaternion-based complementary filter for real-time attitude estimation using low-cost MARG sensors. Sensors, 20(16), 4526.

Wu, F., Wu, H., & Wang, W. (2020). Autonomous underwater vehicle global path planning based on improved particle swarm optimization and dynamic neural network. IEEE Access, 8, 23809-23822.

Li, B., Li, X., Li, S., Li, Y., & Li, C (2011). Velocity estimation for automotive driving environment based on adaptive observer using multi-sensor fusion. IEEE Access, 7, 95048-95056.. (2019). "Getting Started with Arduino" by Massimo Banzi, published in

Haykin, S. (2001). Kalman filtering and neural networks. Wiley.

Särkkä, S. (2013). Bayesian Filtering and Smoothing. Cambridge University Press.

Simon, D. (2006). Optimal State Estimation: Kalman, H∞, and Nonlinear Approaches. Wiley.Parks, T. W., & Burrus, C. S. (1987). Digital Filter Design. John Wiley & Sons.

Lyons, R. G. (2011). Understanding Digital Signal Processing. Pearson Education.

Karris, S. T. (2010). Signals and Systems with MATLAB Computing and Simulink Modeling. Orchard Publications.

Tan, W. K. (2014). Digital Signal Processing: Fundamentals, Techniques and Applications. John Wiley & Sons.

Smith, S. W. (2002). The Scientist and Engineer's Guide to Digital Signal Processing. California Technical Publishing.Mahony, R., Hamel, T., & Pflimlin, J. M. (2008). Complementary filter design for attitude estimation of a fixed-wing UAV. In Proceedings of the 2008 European Control Conference (pp. 2406-2413).

Lua, K. T., & Yahya, W. J. (2015). Orientation estimation using complementary filter for a low-cost UAV. In Proceedings of the 4th International Conference on Control, Instrumentation, and Mechatronics Engineering (CIM 2015) (pp. 245-250).

Yang, T. I., & Lee, C. Y. (2018). Complementary filter design for attitude estimation using inertial sensors. Journal of Control Science and Engineering, 2018.

Williams, B., & Heafner, J. (2013). Exploring Arduino: Tools and Techniques for Engineering Wizardry. Wiley.

Guan, H., & Yuan, J. (2016). A comparative study of orientation estimation algorithms using inertial/magnetic sensors. Sensors, 16(1), 50.

LaValle, S. M. (2006). A comparison of quaternion representations for orientation estimation using an extended Kalman filter. In Proceedings 2006 IEEE International Conference on Robotics and Automation (pp. 373-378).

Zhou, H., Chen, J., & Zhang, J. (2016). A novel quaternion-based Kalman filter for real-time orientation estimation using the error state KF correction. Measurement, 83, 46-55.

Li, X., Li, Q., Li, X., & Li, H. (2018). A quaternion-based Kalman filter for real-time attitude estimation using low-cost MEMS sensors. Sensors, 18(3), 729.

Mokhtarzade, M., Samiei, E., & Mokhtarzadeh, M. (2019). An adaptive orientation determination algorithm for precise attitude estimation of UAVs. Aerospace Science and Technology, 86, 308-317.