Investigate feasibility of utilising a neural-networked set of inertial measurement units to compensate for variations in motion of a COTS RC vehicle in a dryland agricultural context.

A thesis

submitted in partial fulfilment

of the requirements for the Degree of

Master of Applied Science

at

Lincoln University

by

Brett Malcolm Davidson

Lincoln University

2023

Abstract of a thesis submitted in partial fulfilment of the

requirements for the Degree of Master of Applied Science.

Abstract

Investigate feasibility of utilising a neural-networked set of Inertial measurement units to compensate for variations in motion of a COTS RC vehicle in a dryland agricultural context.

by

Brett Malcolm Davidson

Image stabilisation is desired for efficient identification of objects in the path of a self-driving vehicle. The gyroscope and accelerometer of an inertial measurement unit (IMU) can be used to derive the movement of a vehicle, which can then be used by a rotation matrix to compensate for this movement but a gyroscope has inherent “drift” errors, and while the accelerometer of an IMU is more accurate, it has a slower response time which reduces the detection rate.   
Kalman filters are often used to fuse the gyroscope and accelerometer data to reduce the effects of drift, noise and other gaussian-based errors but these are computationally intensive for the sort of lightweight processor that a radio-controlled car could be expected to power. A complementary filter is a simpler and less processor-intensive solution.   
  
This project compares complementary and Kalman filter operation with a neural network when performing sensor fusion and investigates if using a neural network of multiple IMUs reduces errors and enhances performance compared to a single IMU in the context of stabilising a video feed of a consumer-level camera on a moving commercial off-the-shelf radio control vehicle.

**Keywords:** IMU, image stabilisation, neural network, translational sensor drift.

Acknowledgements

I would like to thank my supervisors, Stuart Charters and Mos Sharifi, for their expertise and guidance throughout this project. I would also like to acknowledge the support of Shane Dye of Ara Institute of Canterbury for his assistance in helping me understand the mathematical concepts in this project.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is  
my own except where explicitly stated otherwise in the text, and that this work has not  
been submitted for any other degree or professional qualification except as specified.

*(Brett Davidson)*

Table of Contents

[Abstract 1](#_Toc146552323)

[Acknowledgements 2](#_Toc146552324)

[Declaration 2](#_Toc146552325)

[Table of Contents 3](#_Toc146552326)

[List of Tables 6](#_Toc146552327)

[List of Code 6](#_Toc146552328)

[List of Equations 6](#_Toc146552329)

[List of Figures 7](#_Toc146552330)

[Chapter 1 Introduction 9](#_Toc146552331)

[1.1 Motivation 9](#_Toc146552332)

[1.1.1 NZ Farm information and desired data 9](#_Toc146552333)

[1.1.2 Data Gathering Methods 10](#_Toc146552334)

[1.1.3 Utilising a rover as a data mule 18](#_Toc146552335)

[1.1.4 Obstacle Avoidance 19](#_Toc146552336)

[1.1.5 Image stabilisation 27](#_Toc146552337)

[1.2 Research Objective 39](#_Toc146552338)

[1.3 Thesis Outline 39](#_Toc146552339)

[Chapter 2 Literature Review 40](#_Toc146552340)

[Chapter 3 Method 43](#_Toc146552341)

[3.1 System Design 43](#_Toc146552342)

[3.2 Method Employed 43](#_Toc146552343)

[3.3 Equipment utilised 44](#_Toc146552344)

[3.4 Initial baseline configuration 46](#_Toc146552345)

[Chapter 4 Results 51](#_Toc146552346)

[4.1 Initial data capture and calibration 51](#_Toc146552347)

[4.2 Filter Performance on a stationary vehicle 54](#_Toc146552348)

[4.3 Filter performance on a rolling vehicle. 56](#_Toc146552349)

[4.4 Filter performance on a pitching vehicle 61](#_Toc146552350)

[4.5 Filter Performance on a varied roll and pitch vehicle 63](#_Toc146552351)

[4.6 Neural network using a single IMU. 67](#_Toc146552352)

[4.6.1 Neural network of a varied-motion vehicle: Three Front IMUs 78](#_Toc146552353)

[4.6.2 Neural network of a varied-motion vehicle: All IMUs 78](#_Toc146552354)

[4.6.3 Evaluating differing orientations of IMUs 78](#_Toc146552355)

[Chapter 5 Discussion and Analysis 80](#_Toc146552356)

[5.1 Experimental Design 80](#_Toc146552357)

[5.2 Filter results. 81](#_Toc146552358)

[Chapter 6 Conclusion and Future Works 82](#_Toc146552359)

[Chapter 7 Appendices 84](#_Toc146552360)

[7.1 Appendix 1. Code 84](#_Toc146552361)

[7.2 Appendix 2. Calibration results. 84](#_Toc146552362)

[7.3 Appendix 3. Electrical Characteristics of IMU-29048 IMU 86](#_Toc146552363)

[Chapter 8. References 87](#_Toc146552364)

List of Tables

Table 1.1 Backhaul (long range) communication protocols compared. 14

Table 1.2 Short range communication protocols compared 15

Table 1.3 Communication protocol options available on a range of microcomputers 17

Table 1.4 Single obstacle-avoiding Sensor Summary 24

Table 1.5 Simple representation of a single video image. 26

Table 3.1 Angles of movement per motion test 48

Table 4.1 First three lines of data from the Camera IMU 50

Table 4.2 Initial Camera IMU Gyroscope Offsets 51

Table 4.3 Accelerometer Slope and Offset Table 51

Table 4.4 Sample of Euler angle results using tool arm sensors while at rest. 52

Table 4.5 Joint angles of Robot at rest 52

Table 4.6 Madgwick filter results from a stationary vehicle. 53

Table 4.7 Initial Neural Network training results with default layer size of 10 67

Table 4.8 Training results of initial neural network with a layer size of 20. 68

Table 4.9 Training Results 69

Table 4.10 Neural Network training progress after optimised alignment 74

Table 4.11 Training results of Neural Network after optimised alignment. 74

Table 4.12 Training progress on a 5 layer Neural network 76

Table 4.13 Training results on a 5-layer Neural Network 76

Using the inputs of all three front IMUs, the 10-layer neural network above trained with the Levenberg-Marquardt method performed as outlined in Tables 4.14 and 4.15. Table 4.14 LM Training progress on 3 IMUs with 10-layer NN 77

Table 4.15 LM Training results on 3 IMUs with 10-layer NN 77

Table 4.16 LM Training progress on 3 IMUs with 5-layer NN 77

Table 4.17 LM Training results on 3 IMUs with 5-layer NN 77

Table 4.18 Bayesian Training progress on 3 IMUs with 10-layer NN 78

Table 4.19 Bayesian Training results on 3 IMUs with 10-layer NN 78

Table 4.20 Bayesian Training progress on 3 IMUs with 5-layer NN 78

Table 4.21 Bayesian Training results on 3 IMUs with 5-layer NN 78

Table 4.22 Scaled Conjugate Training progress on 3 IMUs with 10-layer NN 78

Table 4.23 Scaled Conjugate Training results on 3 IMUs with 10-layer NN 79

Table 4.24 Scaled Conjugate Training progress on 3 IMUs with 5-layer NN 79

Table 4.25 Scaled Conjugate Training results on 3 IMUs with 5-layer NN 79

Table 4.26 LM Training progress on 5 IMUs with 10-layer NN 79

Table 4.27 LM Training results on 5 IMUs with 10-layer NN 80

Table 4.28 Bayesian Training progress on 5 IMUs with 10-layer NN 80

Table 4.29 Bayesian Training results on 5 IMUs with 10-layer NN 80

Table 4.30 Scaled Conjugate Training progress on 5 IMUs with 10-layer NN 80

Table 4.31 Scaled Conjugate Training results on 5 IMUs with 10-layer NN 80

Table 4.32 Scaled Conjugate Training progress on 5 IMUs with 5-layer NN 81

Table 4.33 Scaled Conjugate Training results on 5 IMUs with 5-layer NN 81

Table 7.1 IMU Error Coefficients produced from initial calibration (rounded to 8 decimal places) 87

List of Code

All code can be found at the project github location, https://github.com/BratNZ/Thesis

List of Equations

Equation 1.1 Acceleration at rest where g=9.81m/s2. 37

Equation 1.2 Acceleration on a tilted frame 37

Equation 1.3 The mathematical relationship between level and tilted frames. 37

Equation 1.4 Mathematical equations for pitch and roll 37

List of Figures

Figure 1.1 Differentiating Cat-M1 and NB-IoT technologies. Image from (*Standards for the IoT*, 2016) 12

Figure 1.2 Obstacle detecting Sensor Types 20

Figure 1.3 Diagram of roll, pitch and yaw angles. Image from (Fang et al., 2013) 30

Figure 1.4 Fisheye Lens capture example. Image from (BHPhotoVideo.com, n.d.) 33

Figure 1.5 Neuron construction (unit part of the diagram). Image from (Samarasinghe, 2006). 34

Figure 1.6 Example functionality of neural networks for scientific data modelling: (a) fitting models to data or regression, (b) complex classification tasks, (c) discovering clusters in data, and (d) time-series forecasting. Image from (Samarasinghe, 2006) 36

Figure 1.7 Some neural network topologies: (a) single-layer perceptron, (b) linear neuron, (c) multilayer perceptron, (d) competitive networks, (e) self-organising feature maps and (f) recurrent networks. Image from (Samarasinghe, 2006) 37

Figure 3.1 UR5 Robotic Arm Base position showing mounted baseboard 46

Figure 3.2 Initial Design of Neural Network using nnstart 50

Figure 4.1 Madgwick filter results on stationary vehicle (Brown=Roll,Yellow=Pitch,Blue=Yaw) 54

Figure 4.2 Kalman Filter results from a stationary vehicle 55

Figure 4.3 Magnified view of Kalman filter results from a stationary vehicle. 56

Figure 4.4 Madgwick results from a stationary vehicle with a default gain of 0.1 57

Figure 4.5 Madgwick Filter on a rolling vehicle with a gain of 1 57

Figure 4.6 Madgwick filter on a rolling vehicle with a gain of 10 58

Figure 4.7 Madgwick filter on a rolling vehicle with a gain of 15 59

Figure 4.8 Filter and Robot arm data for a rolling vehicle using Camera IMU values. 60

Figure 4.9 Averages of Filter Roll values against robot arm movements 61

Figure 4.10 Filter and robot arm data for a pitching vehicle using camera IMU values 62

Figure 4.11 Averages of Filter Roll values against pitching robot arm movements 62

Figure 4.12 Filter and robot arm varied-motion data using a Madgwick sample rate of 8.33 63

Figure 4.13 Averages of Filter values with a sample rate of 4998. 64

Figure 4.14 Madgwick and Kalman values with a sample rate of 100 65

Figure 4.15 Averages of Madgwick and Kalman values with a sample rate of 100 65

Figure 4.16 Madgwick and Kalman values with a sample rate of 50 66

Figure 4.17 Madgwick and Kalman values with a sample rate of 500 66

Figure 4.18 Madgwick filter beta value set to 0.5 with a sample rate of 100 66

Figure 4.19 Madgwick filter beta value set to 5.0 with a sample rate of 100 67

Figure 4.20 Neural network Model training summary 67

Figure 4.21 Initial Neural Network Performance plot with default layer size of 10 68

Figure 4.22 Initial Neural Network Error Histogram plot with default layer size of 10 68

Figure 4.23 Initial Neural Network Regression plots with default layer size of 10 69

Figure 4.24 Initial Neural Network Performance plot with 20 layers 70

Figure 4.25 Initial Neural Network Error Histogram plot with 20 layers 70

Figure 4.26 Initial Neural Network Regression plot with 20 layers 71

Figure 4.27 Initial Neural Network Training State plot with 20 layers 71

Figure 4.28 Plot IMU and robot values from t=1000 to t=1300 from a dataset of 4999 values 72

Figure 4.29 Filter and Robot angles after new offset values – looks OK 73

Figure 4.30 Filter and Arm results after new offset values - samples from 100-300. 73

Figure 4.31 Filter and Arm results after new offset values - samples from 1000-1300. 74

Figure 4.32 Filter and Arm results after new offset values - samples from 3000-3300. 74

Figure 4.33 Filter and Arm results after new offset values - samples from 4500-4999. 74

Figure 4.34 Training state plot after alignment optimisation 75

Figure 4.35 Performance plot after alignment optimisation 76

Figure 4.36 Error Histogram plot after alignment optimisation 76

Figure 4.37 Regression plot after alignment optimisation 77

Figure 7.1 IMU Calibration graphs for Central Camera IMU 84

Figure 7.2 IMU Calibration graphs for Front Left IMU 85

Figure 7.3 IMU Calibration graphs for Front Right IMU 85

Figure 7.4 IMU Calibration graphs for Rear Left IMU 85

Figure 7.5 Figure IMU Calibration graphs for Right Rear IMU 86

# Introduction

## Motivation

This chapter outlines the requirements around obtaining animal information on a New Zealand farm in a timely and cost-effective manner and a broad discussion of possible solutions for readers not familiar with New Zealand animal data management. The chapter discusses the types of data often recorded and methods of delivering that data back to the farmer. Where an unattended ground vehicle (UGV or rover) is required, various obstacle detection methods are discussed (including choices of sensors) and an overview of camera-based avoidance techniques is presented.  
The issue of image stabilisation when using camera-based systems and common techniques for mitigating this is presented and the research objective is outlined.

### NZ Farm information and desired data

To maintain animal welfare and to obtain the best possible profit, beef cattle farmers in rural New Zealand need to acquire accurate and timely data on the state of their livestock (Swain et al., 2013) . In New Zealand, 80% of beef production is exported and international importers wish to purchase pasture-fed healthy beef (Beef and Lamb NZ, 2017).  
The total area of New Zealand comprises 26.8 million hectares with approximately three quarters of this area above 200 metres in altitude. In 2017, 8.765 million hectares were used for beef and sheep farming spread across 23,403 holdings, averaging 374.5 hectares per farm. 92% of these commercial sheep and beef farms were owner-operated and most farmers farm both sheep and beef cattle as these are complementary for pasture management. In 2020, these farms hosted 26 million sheep and 3.9 million beef cattle, averaging 1,110 sheep and 166 beef cattle per farm. The average figures are somewhat skewed as approximately 45% of the total stock are on farms holding more than 500 cattle (Beef & Lamb NZ, 2021).

Examples of the type of data collected on a particular animal might include animal movement (to determine most visited feeding and watering areas and to track cattle movement in case of disease), weight, as this is the main measurement used to determine price, along with animal health and fat percentage. Monitoring of farm and pasture conditions (weather, pasture quality, water quality, feed and water levels, images, etc) is also desirable as these have an impact on cattle quality (Swain et al., 2013). Due to the higher value of cattle versus sheep, sensors are normally only implemented on cattle (Beef and Lamb NZ, 2017). Optionally, data on the activity of each animal (often obtained by accelerometers around the animal’s neck) could be used to determine if an animal is in distress compared to resting or sleeping (Derek W. Bailey, Mark G. Trotter, Colt W. Knight, 2018).  
  
To track animal movement, a Global Positioning System (GPS) sensor with data logging can be attached to the animal (Y Ropert-Coudert, 2005). To gather weight statistics, the animal must walk over some form of weight scale or via estimation (hoisting an animal to weigh it is impractical) (Wangchuk et al., 2017). Gathering weighbridge data where a farmer is not present to ensure the animal stays still (to ensure an accurate measurement) may introduce errors but this project will not explore these further.

The main problem with gathering data from an animal that moves is efficiently transmitting that data back to the farmer. In the following sections various data gathering methods are discussed and the rationale for using a rover for data capture is outlined. Besides battery life and route planning, the next major issue of using a rover on alpine terrain is object avoidance, and various sensor types are presented and the reason for using a camera as the main sensor (with assistance from others) is presented. One issue effecting the use of a camera is image stabilisation, especially when a rover could be travelling at high speeds. Various image stabilisation methodologies are presented and the utilisation of IMUs is outlined.

### Data Gathering Methods

There are three main methods available to collect data from a remote location.   
Radio Frequency (RF), a data mule (autonomous device travels to sensors to collect data) and a hybrid solution of these.

#### **1.1.2.1 Radio** frequency data collection.

Possible radio frequency (RF) implementations include satellite (De Sanctis et al., 2016) , cellular service (Gaddam & Rai, 2018) , dedicated radio links such as WiMAX (Andreev et al., 2015), LongRange Wireless Area Network (LoRaWAN) (Adelantado et al., 2017; P. S. Cheong et al., 2017; Haxhibeqiri et al., 2018b; Lavric & Popa, 2018),Wifi-Halo (Tian et al., 2021), Bluetooth (Gomez et al., 2012; Team, 2019), ZigBee (Baronti, P Pillai, P Chook, V.W.C Chessa, S Gotta, A Hu, 2007; Gheorghiu & Iordache, 2018) or similar (listed in order of greatest range) (Al-Sarawi et al., 2017).  
These systems transmit the signals from the sensors to either a central location (this design is called “hub and spoke”) or in a form of ““ where sensors transmit via other sensors until they are in range of the farmer (Sethi & Sarangi, 2017) .  
Beef cattle are generally farmed in hill country so a direct line of sight from a sensor on the animal to a central radio (or satellite) point is not always possible (Spark NZ, 2023; Starlink, 2022).   
Animals tend to travel in groups so a mesh configuration is not practical and would be cost-prohibitive over the distances involved (Ramseyer et al., 2009).

RF techniques can be split into two types; long range (backhaul) or short range.

##### Long range or Back haul

Long range radio frequency systems include satellite, cellular service, dedicated radio links and Long-Range Wireless Area Network (LoRaWAN). The strengths and weaknesses of these systems in an alpine farmland context is discussed in this section.  
  
Satellite systems need a clear line of sight to a minimum of 4 satellites greater than 15 degrees above the horizontal plane for GPS location but data communication can be achieved with one satellite. While GPS is a free service, satellite data is expensive, location-restrained, power-hungry and generally slow although companies such as Iridium are improving their 1.4kbs links to 512kbs and StarLink has recently expanded its operations in New Zealand with promises of greater than 50Mb/s upload speeds. Starlink, as of writing, costs $370 to install and $160 per month and has extensive coverage (Starlink, 2023). The main advantage of using satellite communications is that the architecture can be flattened to just sensors and a back-end infrastructure, which aids simplicity.   
Disadvantages are that satellite data transmission is expensive as each animal would need a satellite transmitter and data plan. Confidentiality of data would also need to be considered.   
The use of GPS is free and transmitters are ubiquitous and are low-cost so satellite transmission should be reserved for GPS location data only.

Cellular systems.

Cellular technologies (4G, 5G). Public cellular technology support in the high country is sparse although recent OneNZ announcements are supplementing cellular support with the use of Starlink satellites, to be released in 2025 (OneNZ, 2023a).   
  
Current fourth generation (4G) coverage is possible in rural high country but there are pockets of no coverage from commercial providers (OneNZ, 2023b; Spark, 2021). Coverage would need to be evaluated and confirmed on each site.  
Purchasing a subscriber identity module (SIM) card for each animal would quickly become expensive, even without considering potential damage from animal activity, but like GPS data, does reduce the architecture down to sensor devices and the cloud.  
It is possible to utilise generic consumer SIM technology to perform connectivity but the customer would be paying for voice capability where that is not required.  
  
Fifth generation (5G) cellular technologies (Cat-M1, LTE-M, Narrowband-IoT) chipsets are cheaper than 4G chips but due to limited demand, costs are still somewhat high. This will most likely change as demand increases.   
Cat-M1 operates at 1.4 MHz bandwidth and this wider bandwidth allows Cat-M1 to achieve very good data rates (up to 1 Mbps – generally 200-400kbs) with low latency and device positioning capabilities. Cat-M1 supports voice calls and connected mode mobility. Cat-M1 generally has a similar range as 4G (10-15km) since it extends the same LTE-based cellular technology underpinning these.   
LTE-M is a competitive system to Cat-M1 with similar features.  
Narrowband-Internet of Things (NbIoT) sacrifices throughput (250kbs maximum upload with a payload size of up to 1600 bytes) for extended range and improved power usage.   
All 5G technologies mentioned here can support more than 55,000 simultaneous clients.   
Differentiating Cat-M1, NB-IoT and GSM-IoT.
Source: 

Figure 1.1 Differentiating Cat-M1 and NB-IoT technologies. Image from (*Standards for the IoT*, 2016)

Dedicated link RF options

WiMAX (WorldwIde operability over MicrowAve) is a technology using Microwave links.  
WiMAX does not require line of sight to operate and has further range than conventional Wifi (30-50km maximum limit for line of sight). Power usage is very good compared to cellular networks (Deruyck et al., 2010). There are two main types of WiMAX; fixed and mobile and these are very different in infrastructure. There are few public WiMAX deployments in NZ and none in rural spaces so the farmer would need to invest in WiMAX infrastructure and deploy stations in the appropriate places to get coverage to and from the remote areas. WiMAX appears to be losing the battle for spectrum to cellular providers and a “recent” (2017) spectrum analysis report indicates there is little intention for WiMAX to be implemented in a large scale in NZ (New Zealand IoT Alliance, 2019). As such, for futureproofing rationales, WiMAX Is not recommended for use.   
  
SigFox is a proprietary system where devices connect to base stations which connect to each other via point-point links (maximum range of 50km) back to the SigFox cloud. Each sensor can report up to 12 bytes in a message (26 bytes in a frame), with a maximum of 140 messages per day. Transmission speed is either 100 or 600 bits per second (SigFox, 2023b).  
Coverage of some areas in New Zealand is suggested at Sigfox’s commercial website (SigFox, 2023a) however both the proprietary nature of SigFox and it’s low transmission rates, make it unappealing as a solution, and SigFox is not recommended for use.  
  
LoRa is a competitor to SigFox. This is a low power communications protocol designed to send small packets of data at regular intervals with low-power usage. It is a proprietary protocol owned by Semtech. LoRaWAN is an open source point to multipoint messaging protocol built to utilise the Lora communications protocol (*What Is LoRaWAN® | LoRa Alliance®*, n.d.).  
LoRaWAN has very low power requirements and a practical range of up to 3km in an urban environment (Augustin et al., 2016) and up to 15km in an rural environment (Haxhibeqiri et al., 2018a) but has a duty cycle limit (1% duty equals 36 seconds of communications time per hour for each device). It prefers line-of-sight communication to perform well (Haxhibeqiri et al., 2018a). Transmission speed varies from 300bps to 37.5kbps with a maximum payload of 246 bytes.   
The theoretical maximum nodes in a LoRa system are 10,000 but LoRaWAN has scaling issues of a maximum of 8,000 devices (1000 nodes per channel with 8 channels) at a collision rate of 95% (Lavric & Popa, 2018). More stations are required in order to alleviate this, which is possible as LoraWAN operates in a star of stars fashion where every station receives the transmissions of all devices around it.

In NZ, Spark concentrate LoRaWAN in the main urban centres (Spark NZ, 2023), so in many sites, the farmer would need to invest in LoRaWan infrastructure and deploy stations in the appropriate places to get coverage to and from the back-end stations.   
There are no fees at present for operating LoRaWan systems but a license is required. (*Short Range Devices GURL | Radio Spectrum Management New Zealand*, n.d.)  
Each LoRaWan edge station requires a TCP/IP-based backhaul protocol to deliver content to the cloud. This could be possible within an area covered by one of the 4G or 5G satellite protocols at an additional cost and complexity.

Dedicated radio links are a possible solution but would require specific designs for each station. Annual licensing costs also apply (*Fixed Link Licence | Radio Spectrum Management New Zealand*, n.d.) Depending on the distances it might be more cost-effective to implement direct links but this technology implementation would be site-specific and costly, and will not be evaluated in this review.

Table 1.1 Backhaul (long range) communication protocols compared.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Protocol | 4G | Cat-M1/LTE-M | NbIoT | LoraWan |
| Standard |  | 3GPP LTE CatNB1 | 3GPP LTE CatNB2 | IEEE 802.15.4g |
| Frequency | Licensed | Licenced | Licensed | 915 MHz |
| Range | 10 km | 10 km | 12 km | 5 km |
| Transfer Rate | 2 Mbps | 300-400 kbs | 20-127 kbs | 50 kbs |
| Power usage | 250 mW | 220 mW | 200 mW | 125 mW |
| # clients | 55,000 | 55,000 | 55,000 | 10,000 |
| Availability | Very high | Very high | Very high | Limited |
| Major Limitations | Range | Data rate and capacity | Range and data rate | Data rate, capacity and range |
| Recommended Applications | Where Video/Voice are required | Where Images/Voice are required | Where Images are required | For small datasets generated hourly. |
| Farm Use | Not ideal | Ideal for large datasets if available | Ideal for medium sized datasets if available | Ideal for small datasets if available |

##### Short Range RF options

ZigBee is based on the IEEE standard 802.15.4 (Baronti, P Pillai, P Chook, V.W.C Chessa, S Gotta, A Hu, 2007) for small personal LAN communication with a range from 10-100m (for the standard variant) or a practical range of 3.9km (for the long range Zigbee variant - 15km is advertised) (Jawad et al., 2017). Maximum nodes are 65,000 which should be ample for even a large farm. Range limitations can be overcome by using repeaters. ZigBee requires line of sight communication which is a limiting factor but has low power requirements. Transmission rate varies from 20 to 250kbs. Zigbee is designed to support multi-hop networks. There are three types of devices in a Zigbee network; FFD (Fully functional device which can also act like a router), a RFD (Reduced Functionality device) and a single coordinator. Routing traffic is in-band with the data.  
Zigbee is a potential solution provided that the distance between stations does not exceed the line-of-sight (LOS) communication range. The main advantage of ZigBee is that it can use bursts which enables the rapid transmission of data however it is a short range protocol and an intermediary device would be needed to communicate data back to the farmer. Zigbee is a possible solution for communication between cattle and to a very local transmission site but other devices will be needed to transmit data back to the farmer.  
  
Dash7 (*DASH7 Alliance – An Open Specification*, n.d.) arose from military RFID use and primarily utilises the 433MHz band which gives it multi-km range. To improve performance some network cards support the 868 and 915MHz unlicensed ranges as well. Bandwidth is either 25kHz or 200kHz which gives transfer rates of 9.6kbit/s, 55.55kbit/s or 166.7kbit/s. Max packet size is 256 bytes. (Arsalan et al., 2018). There are no retail providers of Dash7 devices in NZ so this technology is not recommended.

Wifi HaLow is based on the IEEE 802.11ah standard (Shanmuga Sundaram, 2016) and consumes less power than standard wifi with a longer range; up to 750m using the “unlicensed” frequencies from 915-928MHz. Being a superset of Wifi, Wifi Halow supports IP based communication and is designed to support a lot of simultaneous clients (up to 8191 as 13 bits are used for an ID) in a star-shaped network with data transmission rates from 150kbs to 347Mbs. As such, Wifi HaLow is only possible for cattle monitoring. Up to 6960 stations can be simultaneously connected over 1km while transmitting 100 bytes of data every 60 seconds without packet loss when tuning TIM (12 groups) and RAW (2 slots) parameters. In terms of power efficiency, 802.11ah enables 500 IoT devices to turn on their radios 3% of the time when transmitting every 60 seconds (Šljivo et al., 2018). Wifi-HaLow is included here despite there being no retail supply of network cards in New Zealand as it has some distinct advantages that may make it worthwhile to source parts from overseas. One of these advantages is the multihop relay support integrated with the technology.

Bluetooth V5 was released as an enhancement for IoT devices. The theoretical range (240m@125kbps) is 2 times that of Bluetooth Low Energy (mentioned below) but requires significantly more rf output power (from 10mW to 100mW) (Team, 2019). The Bluetooth V5 protocol stack supports IP based communication natively which is an advantage in designing a system. There are four data transfer rates of 2Mbps, 1Mbps, 500kbps and 125kbps, with increased range at each reduction of rate.  
For both power usage and connection speed advantages, Bluetooth is very attractive. Bluetooth 5 also includes mesh networking, making this a significant challenger to ZigBee.  
  
RFID is short range only (less than 2m) but has extremely low power usage (Duroc, 2022; Jia et al., 2012; Landaluce et al., 2020). RFID tags are required for cattle identification in NZ (NZ Government, 2018) so these tags will be present, regardless of any other technologies chosen (Williams et al., 2019).  
  
LoRaWAN can be used as a short haul system but the other systems are more cost-effective, albeit with reduced range.

Table 1.2 Short range communication protocols compared

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Parameter | RFID | ZigBee Default | ZigBee Long Range | BluetoothLow Energy | BluetoothVersion5 | LoraWan | Wifi HaLow |
| Standard | RFID | IEEE 802.15.4 | IEEE 802.15.4 | IEEE 802.15.4 | IEEE 802.15.4 | IEEE 802.15.4g | IEEE 802.11ah |
| Frequency | 433Mhz | 2.4 GHz | 915 MHz | 2.4 GHz | 2.4 GHz | 915 MHz | 915-930 MHz |
| Range | 2m | 60 m | 14km / 6.5km | 10m | 750m | 5km | 1km |
| Data rate | n/a | 250 kbs | 10 / 200 kbs | 1 Mbps | 125kbps-2 Mbps | 50 kbps | 150 kbs (1km range) – 80Mbps |
| Power use |  | 36.9  mW | <250mW | 10 mW | 10 mW | 125 mW | 11mW |
| Network size |  | 65,000 | 65,000 | App-defined | App-defined | 10,000 | 8,191 per AP |
| Network topologies |  | P2p, Tree, Star, Mesh | P2p, Tree, Star, Mesh | Star | Star | Star of stars | Star and Relays |
| Native IP |  | No | No | NIC-Specific | NIC-Specific | No | Yes |
| Limitations |  | Line of sight | Power usage | Short range | No mesh support | Data rate, capacity | Limited devices |
| Farm use ideals |  | Not ideal | For small datasets only | Not ideal | Only useful for animal to collector site use | Not ideal | Ideal but imported cards may be expensive and not supported in NZ |

The choice of short range RF solution for an alpine farm will depend on the amount of data transmitted and the number of animals. Wifi HaLow is an interesting option but it’s cost and lack of support (at present) makes recommending Bluetooth V5 an easy choice.

Table 1.3 Communication protocol options available on a range of microcomputers

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Comms option | Pi V4 | Rock Pro64 | Jetson Nano | Tinker EdgeT | Arduino MegaR3 | Microbit | ESP32 |
| GPS | M | M | M | M | M | O | O |
| 802.11ah | O | O | O | O | O | O | O |
| BT 5 | I | M | M | M | M | X | O |
| BT BLE | M | I | M | I | M | X | I |
| ZigBee Std | M | M | M | M | M | O | O |
| ZigBee LR | M | M | M | M | M | O | O |
| 433MHz | M | M | M | M | M | O | O |
| LoraWan | M | M | M | M | M | M | M |
| NbIOT | M | M | M | O | M | O | O |
| Cat-M1 | M | M | M | O | M | O | O |
| LTE-M | M | M | M | O | M | O | O |

Key: I=Inbuilt, M=Module available in NZ,O=Module available from Overseas and X=Not Suitable  
  
If reliable frequent communication is not possible extra storage may need to be provisioned on the sensor device to act as a data buffer, raising the costs and lowering the battery life of each sensor unit.

#### 1.1.2.2 Data Mule

To avoid the monthly costs and possible infrastructure investments in RF backhaul technology, an autonomous travelling device (rather than the actual farmer) could visit the herd, download the data and upload this data on return. This concept is called a “data mule” as the device is purely a carrier for data in a similar manner as the (humorous) ethernet over avian carrier protocol (Waitzman, 1990). At the time of writing, this approach would incur significant range limitations due to current battery technology and requires efficient pathfinding requirements of the device, along with the requirement that the device distinguish between animals and other obstacles. The device would need to navigate close enough to each animal to read the passively powered RFID sensor information without being trampled and/or startling the animal and obtain data from the other sensors on the animal. Most research on data mules in a rural context has concentrated the use of unattended aerial vehicles (UAV or “drone”) as these significantly reduce the issues involved in efficiently traversing terrain and avoiding obstacles between the various sensor locations. This approach is not feasible at present in New Zealand as the current civil aviation laws in the country prohibit autonomous operation and require an operator to have line-of-sight visibility of a drone at all times (Aviation Authority of New Zealand, 2015).

An unmanned ground vehicle (UGV or “rover”) could be utilised to visit each animal however these face issues of more difficult route planning and obstacle avoidance concerns. As such, using a rover data mule to visit individual animals is not addressed in this project.  
  
1.1.2.3 Hybrid of RF Short-Range and Data Mule solution

Beef cattle tend to revisit feed and water lots (Johnstone-Wallace & Kennedy, 1944; Martina et al., 2015) so a hybrid of the RF short-range option and a rover-based data mule could be implemented where cattle visit a solar-powered data-aggregation feed/water site and transmit their data by one of the short range RF options (Bluetooth version 5 for example) (Jawad et al., 2017) and then, a rover data mule can be used to deliver the data from the aggregation point to the farmer at regular intervals. An advantage of this method is that photographs of animals can be taken when they visit the site to enable a visual inspection of animal health and scales to measure animal weight could also be implemented. Each data-collection site’s short-range RF infrastructure will need to be repeated for every feed/water site where observations are required. The data will need to be stored for as long as it takes for a communications link to transfer the data or for a data mule to visit the site. It is envisaged that a maximum period of at least once every 24 hours would provide timely data.

### Utilising a rover as a data mule

The main difficulties in using a rover as a data mule are ensuring reliability, effective range/battery life, efficient sensor data capture/storage, efficient route planning and obstacle avoidance (Borges De Sousa & Andrade Gonçalves, 2010; Chemhengcharoen et al., 2019; Manderson & Hunt, 2013; Petterson, 2020).   
Reliability (Nguyen-Huu & Titus, 2009), Battery life (Hall, 2021; Zogopoulos, 2021) and Data storage methodologies and systems are improving all the time so the current limitations of these will not be addressed as it is assumed they will be alleviated in the near future.   
Route planning and obstacle avoidance are somewhat related as an efficient route planning system must also address any potential obstacles.  
The route planning problem is generally treated as an NP-Neighbourhood solution (Ab Wahab et al., 2015; Koenig et al., 2004; Sugihara & Gupta, 2011) however the presence of potential obstacles requires that the route-planning system adjust a planned route dynamically to account for a new localised route to avoid the obstacle and continue (Masehian & Katebi, 2014).   
To avoid obstacles the rover must first be able to detect them, and the following section outlines possible detection sensors and their limitations.

### Obstacle Avoidance

As a rover traverses the terrain, it is essential that it avoids objects that could cause damage to itself and/or vegetation. Animal (including human) object avoidance is paramount. Despite considerable research into rover obstacle avoidance

#### Obstacle Types

Obstacles can be defined as either static (fixed location) or dynamic (mobile).   
Many obstacle-avoidance systems treat all objects as dynamic objects, a safe default, but this approach requires significant processing power and prevents a rover from passing these objects closer than it might otherwise, reducing rover range.   
Where systems do distinguish between static and dynamic obstacles, this is often determined when these are scanned (“is the object moving right now?”) which offers little processing benefit over the purely dynamic approach. Static objects can be predefined before the rover ventures on its run when an aerial map is loaded to form an initial route, but a map may not identify all objects.  
Classifying obstacles via a static map loaded at initialisation on site and classifying obstacles and updating this map as the robot passes each object would provide the best efficiencies.  
  
To avoid an object we need to know what types of obstacle are expected and how to identify them.  
The nature of rural obstacles will depend on the topology of the farm, but in New Zealand, will likely comprise of:  
1. Various vegetation types of various lengths, some of which may be considerably taller than the rover, along with trees and shrubs at various stages of growth.  
2. Various soil types, ranging from very sandy to extremely clay-heavy, which retain water differently, so the rover must be able to distinguish between dry or boggy conditions and ideally differentiate between deep and low-lying pools of water.  
3. Rocks and stones, which are numerous on NZ farms and the rover must decide to either climb or circumnavigate these obstacles.  
4. Various weather conditions - sun, snow, rain, ice and fog.   
5. The terrain gradient which needs to be traversible or avoided.   
6. Rabbit warren holes and/or other large depressions which need to be avoided.  
7. Farm fencing, where the rover must either stay within the defined boundary and/or detect and utilise methods to pass through fences, gates, cattle stops, etc.  
8. Humans, animals and any farm equipment, which must be avoided, wherever they are.‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬

#### Typical Obstacle-detection sensors

Sensor classification falls broadly under two functional axes: proprioceptive/exteroceptive and passive/active (Siegwart et al., 2011).  
Proprioceptive sensors detect the rover’s state such as position, orientation and speed via velocity sensors (W. H. Zhu & Lamarche, 2007), inclinometers/gyroscopes (Dai et al., 1996) , position sensors (Chao et al., 2013), heading sensors or accelerometers (Beliveau et al., 1999), etc.  
Exteroceptive sensors collect information from the environment around the rover via such sensors as time-of-flight (Foix et al., 2011), lidar (Yan et al., 2015), laser (Suh, 2019), sonar (Choit et al., 2005), microwave radar (C. Li et al., 2017), and cameras (Bernini et al., 2014).  
The other axis delineates between passive and active sensors.  
A passive sensor such as a CCD camera (Fossum & Hondongwa, 2014) or thermal camera (Akagawa, 1996) receives environmental information energy whereas an active sensor generates and emits energy and measures the response. Passive sensors use less power but often require more processing time to process resulting datasets.  
Active sensors tend to require less processing time and have further range but require power to emit the signals and can be influenced by other sources of similar energy.  
  
Diagram

Description automatically generated

Figure 1.2 Obstacle detecting Sensor Types

Radio detection and ranging (RADAR) is an active system often utilised on full-size vehicles as it has a long range and can cope with dusty conditions. The system works by emitting radio signals and timing how long it takes for the signal to return after being bounced off an obstacle. Rain and snow can cause attenuation, but in a large vehicle, power may be boosted to compensate for this. The data returned is of high resolution (normally a point) but categorisation of an object is not possible without further processing.

Forward looking ultra-wideband (UWB) RADAR can penetrate grasses (Wong et al., 2003) to determine rocks hidden behind tall grass, etc. but developing a data model for the reflected nature of holes such as rabbit warrens may be indistinguishable from background noise and may not measure far enough ahead of the rover for the rover to be able to avoid the obstacle. This is an aspect of object avoidance that does not appear to be well-researched‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬ besides Wong’s work but is a project more suited to an RF engineer to design the required antennae array and is out of scope for this project.  
  
Sonic navigation and ranging (SONAR) (typically an active ultrasonic sensor) works like RADAR but the systems emits sound rather than radio waves. SONAR suffers from environmental constraints such as wind, temperature and humidity which effect the measurement accuracy (Mohammed et al., 2020). Ultrasonic sensors are also limited to the speed of sound (the signal attenuates heavily in air) and thus have limitations on their range and sensing rate. The wavelength is approximately 4mm and so can miss some narrow surfaces such as a narrow bush branch or a fence wire, etc. As SONAR uses reflection, like RADAR, any surfaces at oblique angles (soft edges) to the rover may not reflect back (Mubarak, 2013) and reflections from multiple objects may also confuse a sensor. Vegetation tends to confuse a sonar system as well, particularly if the grass is blowing in a wind. The wide beam angle reduces directional resolution (overlapping ultrasonic sensors have been used to reduce this limit). (Kleeman & Kuc, 2008). SONAR’s main advantages are price and that, being sound based, it is immune to visibility issues such as dust, fog and night.  
  
Light detection and ranging (LIDAR) has been used effectively in many rover deployments and is the method of choice for many household vacuum cleaner robots (Mohammed et al., 2020). LIDAR is an active light (light amplification by stimulated emission of radiation or “laser”) reflection system and is less affected by background solar radiation or night but is affected by dust, fog, rain, ice and snow (Formsma et al., 2010). 360degree LIDARs are a relatively cheap form of object detection, enabling edge detection of an object, but only scan in a single dimension, missing objects higher or lower than the narrow scan angle. Scanning vertically while a motor rotates the lidar unit provides a complete viewpoint but reduces scanning time, limiting predictive ability. Wind effects on grass may also confuse the sensor. Maximum range is about 80m with an accuracy of +/- 50mm over a range of 20m (Crane et al., 2006) but consumer-level LIDAR units tend to have a range of 12m (Adafruit, 2022).  
  
Far-infra-red (FIR) passive camera sensors have long range and are immune to most environmental factors such as rain, snow, etc. and provide improved resolution at night. They suffer from high cost and moderately high computer processing power required to analyse the visual data, along with low resolution (compared to a conventional camera) and only produce grey-scale images. Distinguishing objects in cold environments may be difficult (Mohammed et al., 2020). Matthies and Rankin (L. Matthies & Rankin, 2003) found that thermal signatures are effective at determining depressions and other holes in the ground, although their approach worked best at night and would need to take solar radiation into account for daylight use. Other Infra-red systems (typically active systems) are more effected by environmental conditions, especially solar radiation from the sun.

FIR passive camera systems can be used as rangefinders (Druzhkov & Kustikova, 2016; Gao et al., 2016; S. Huang et al., 2020; Mohammed et al., 2020; Moravec, n.d.; Rajavarshini et al., 2021; Schäfer et al., 2005). Backlighting and lens flare are the main issues with using cameras besides their limitation of daylight-only use, without additional lighting.  
  
Monocular cameras use the size and shape difference between successive image frames to determine depth and can be confused by areas of sharp contrast, especially shade in a bright sun (Michels, 2005).   
Stereo cameras can have better noise immunity and these are easier to derive range information from (via triangulation from the two cameras) but processing two sets of images requires more computing power.  
Schäfer, etc. implemented a “depth discontinuity” method to determine obstacles above and below ground level using a stereo camera, recognising that traditional ground level references do not apply in a typical outdoor topology (Schäfer et al., 2005). The weakness of this approach is that the camera needed to be angled towards the ground, and thus there is a contradictory relationship between angle, the size of the obstacle that can be determined, and rover speed, which was not explored in their work. It also means that the camera may have limited forward vision, depending on the angle, which may require additional sensors to address.  
Sharma and Shah used image processing to determine if animals were present with an 82.5% accuracy (S. U. Sharma & Shah, 2017) however this required significant processing resources (Corei5) to produce 10 fps images.

Detection of holes in the ground (rabbit warrens or natural depressions) has been briefly explored (Ghaffari et al., 2004; Kusuma Arbawa et al., 2021; L. Matthies & Rankin, 2003; J. Wang et al., 2016) by using thermal signatures (infra-red) which works well at night provided that ground cover is not too dense but has limited effectiveness on a sunny day (L. Matthies & Rankin, 2003). Matthies and Rankin propose modelling solar illumination for their approach to be usable in daylight conditions.  
  
Determining and adapting to ground conditions has primarily been limited to detecting traction loss on the driven wheels and compensating by pulsing torque in a similar manner to anti-lock braking. The methodology is well developed and is the approach taken by full-size vehicles.   
Determining ground conditions before the rover reaches them has had less research. Khan and Ahmed (Khan & Ahmed, 2021) used a CNN (convolutional neural network) to detect snow on road images and Kawai (Kawai et al., 2014) used a car-mounted webcam to distinguish road conditions at night via colour differences but a road is a known surface and rural topologies vary considerably. Wading sensors as utilised by Tran (Tran et al., 2015) only face downwards and can’t predict the depth of water ahead of a vehicle.  
  
Compensating for ground elevation and slope has been treated extensively in both research and general industry, with a gyroscope being the most-used sensor used to detect the current gradient the vehicle is travelling on. Predicting upcoming gradients has had less research. A stereo camera techniques such as Schäfer’s (Schäfer et al., 2005) holds some promise for holes or negative gradients but their technique is not as applicable to positive gradients.  
  
The standard post and wire fence used in NZ farming is difficult to detect as the diameter of the wire is less than the detection scan resolution of many sensors. An ultrasonic sensor requires an object of at least 4mm in width, for instance. In this regard a LIDAR sensor ought to be better but very small objects may be missed.   
  
If the fence needs to be traversed (a common occurence on a large farm), determining a suitable place to cross is somewhat difficult, especially where long grasses may cover a tunnel and/or a gap in the bottom wire.‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬ ‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬‬  
  
Wind effects acting on objects can make object identification difficult. Object identification when facing into the sun or other lighting challenges are also likely to be issues in a real-world environment.

#### Sensor fusion approaches

As a rover should operate in all weathers and conditions, multiple sensors are likely to be required to compensate for the deficiencies in each sensor type. Multiple sensor calibration has been investigated thoroughly at a basic level of integrating two sensors (Rodriguez F. et al., 2008; Zhou et al., 2018) but integration of more than two sensors appears to be lightly touched in research.  
  
Manduchi (Manduchi et al., 2005) implemented a combination of a colour-stereo camera to categorise grass, bark, soil and rocks, holes, etc. along with analysis of a single-axis lidar to detect rocks partially hidden by grass. Manduchi found soil/dry grass categorisations was difficult with pure colour matching and they suggested additional techniques such as visual texture mapping or multispectral thermal analysis. (Castano et al., 2014; Gilmore et al., n.d.)

Table 1.4 Single obstacle-avoiding Sensor Summary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Sensor | Radar | Ultrasonic | Lidar | IR | FIR | Stereo Camera | Single Camera |
| Type | Active | Active | Active | Passive | Active | Passive | Passive |
| Cost | High | Low | Med | Med | Med | High | Med |
| Range (m) | <250 | <10 | <50 | <12 | <50 | <10 | <100 |
| Precision | Low | Low range | V. High | V. High | V. High | Med | Low |
| Resolution | Low | Low | Precise | V. High | V.High | Med | Med |
| Reliability | High | Low | High | High | High | Med | Med |
| Power | High | Med-High | Med-High | Low | Med | Low | Low |
| Processing | Fast | Speed of sound limits | Fast | Med | Fast | Fast CPU needed | Fast CPU needed |
| Rain influences | No but reduces range | Yes | Yes | Yes | No but reduces range | Yes | Yes |
| Dust influences | No | No | Yes | Yes | No but reduces range | Yes | Yes |
| Fog influences | No | No | Yes | Reduces range | Reduces range | Yes | Yes |
| Sound influences | No | If High pitch | No | No | No | No | No |
| Light influences | No | No | No | Direct sun | Direct sun | Lens flare | Lens flare |
| Temp influences | No | Yes | No | Can reduce contrast | Can reduce contrast | No | No |
| Other influences |  | Echoes Wind | Snow | Snow |  | Wind | Wind |
| Classifies object | No | No | In single plane | Yes | Yes | Yes | Yes |
| Light required | No | No | No | No | No | Yes | Yes |
| Single point Reliability | No | No | - | - | - | - | Static objects only |
| Research | (Blanche et al., n.d.; J. Huang et al., 2001; Mohammed et al., 2020; Norouzian et al., 2019; Schneider & Wenger, 2003; Yamauchi, 2008) | (De Simone et al., 2018b; Jiménez et al., 2014; Kapoor et al., 2018; Kleeman & Kuc, 2008; Mohammed et al., 2020; Rosique et al., 2019; Shing et al., 2008; Sulaimon Alli et al., 2018) | (Byeon & Yoon, 2020; Goodin et al., 2019; Lebakula et al., 2021; Lu et al., 2020; Mohammed et al., 2020; Wallace et al., 2020; J. Wang et al., 2016; Yamauchi, 2006, 2008; Zhong et al., 2020) | (L. Matthies & Rankin, 2003; Ren et al., 2020; Sulaimon Alli et al., 2018) | (Dwork et al., 2006; L. Matthies & Rankin, 2003) | (M. K. Cheong et al., 2016; Gao et al., 2016; Hautì et al., 2006; Karuppuswamy, 2000; Lecun et al., n.d.; Lwowski et al., 2014; Manduchi et al., 2005; Mannar et al., 2018; Michels, 2005; Nguyen Viet & Marshall, n.d.; Noori et al., n.d.; Odeh & Faqeh, 2009; Schäfer et al., 2005; P. S. Sharma & Chitaliya, 2007; Simmons et al., n.d.; Sun et al., n.d.; van Hecke et al., 2018; S. Wang et al., 2021; Yamauchi, 2008; Z. Zhang, 2012) | (Chaudhary et al., 2019; Hautì et al., 2006; Hoffman et al., 1999; Hoffmann et al., 2004; Karuppuswamy, 2000; Lecun et al., n.d.; Lwowski et al., 2014; Manduchi et al., 2005; Mannar et al., 2018; Michels, 2005; Nguyen Viet & Marshall, n.d.; Sun et al., n.d.; van Hecke et al., 2018; S. Wang et al., 2021; J. Zhang et al., 2017; Z. Zhang, 2012) |

A very high resolution camera is the suitable solution, supplemented with a non-visual sensor such as SONAR or RADAR, however this will come with its own benefits and limitations.  
  
Camera-based solutions to recognise obstacles

Cameras are often used in industry to perform obstacle detection as they provide many advantages.   
To provide obstacle detection a video camera is used as differences between successive image frames can be analysed to determine if a recognised object is getting closer or not.  
  
A single video image is a matrix of pixels (a single piece of information in the image), with each pixel having a combination of three luminance values, each measuring the luminance value of a particular colour. Pixels are normally manufactured in Red, Green and Blue variants and these systems will be discussed in this report. An individual image can be represented as a matrix of values as per Table 1.5. The resolution of an image depends on how many pixels are in the image and the size of the number storing the luminance values.

Table 1.5 Simple representation of a single video image.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Horizontal Pixel  1 | Horizontal Pixel  2..(N-1) | Horizontal Pixel  N |
| Vertical Pixel 1 | Red, Green and Blue luminance values | Red, Green and Blue luminance values | Red, Green and Blue luminance values |
| Vertical Pixel 2..(N-1) | Red, Green and Blue luminance values | Red, Green and Blue luminance values | Red, Green and Blue luminance values |
| Vertical Pixel N | Red, Green and Blue luminance values | Red, Green and Blue luminance values | Red, Green and Blue luminance values |

Obstacle identification is generally based on edge detection of an image – determining objects by recognising the boundaries of the object. "Canny" (extension of "Sobel") Edge Detection (Ganesan & Sajiv, 2018) and "Fuzzy" edge detection (Haq et al., 2015) (for images with a high noise floor) are the standard industry-utilised methods of edge detection.  
  
The Matched Filter extension to the Canny approach (Ofir et al., n.d.) is interesting as the algorithm is designed for soft (curved) edges (more likely to be present in a natural environment) but the curved edges evaluated were letters and numbers in a 2D image rather than a receding three-dimensional image plane. Run times for their optimised algorithm (using sampling) were 0.6 seconds using C++ on an I7 with 16Gb of Ram with an image of 129x129 pixels). For a 257x257 image, run time was 5 seconds which is too slow for real-time use on a rover.  
  
Utilising wavelet analysis is a very interesting approach, is almost perfect in resolution and would work with a receding plane BUT the authors state the process is not useful in noisy environments (Damlamian & Jaffard, 2019) and the processing requirements of a “Datacube MV200”, 68040 CPU and a “Sparcstation” workstation, as implemented, are too computationally (and economically) expensive for a low cost battery-powered rover.  
  
Utilising a multi-spectral camera system enables detection of soil and vegetation (Larry Matthies et al., 1995) but would involve power and computation usage beyond the rover’s capabilities unless filters could be dynamically applied to a camera.

The main problem of using any form of edge detection algorithm is the quality of the image.  
An ideal image has no noise, and perfectly describes the scene it represents. In the real world, images are of a limited pixel and luminance value count and the lens used to capture the light introduces distortions of scale in the image. The image may consist of extremes of light that the imaging system may not perfectly capture (commonly called over and under exposure). Rain, snow or other environmental conditions may also reduce image quality. These are all well-known problems and many different solutions exist to reduce their impact on image quality.  
  
One external issue impacting quality that is particularly relevant to a rover is image stabilisation.

### Image stabilisation

As a video stream is a successive stream of images, the motion of objects around a vehicle can be determined by locating an object (often called a “feature”) in an image frame, and then comparing its location in subsequent frames. (If a feature is static, the same process can be used to determine the velocity and direction of the vehicle). For this approach to work, initial feature detection is required and as the vehicle passes this feature, new feature/s will need to be acquired and used for comparison. The vehicle’s own movement (including any chassis vibrations) will also need to be accounted for.   
  
What makes this approach more problematic is that, in rural New Zealand, the topology may not be smooth, and so our vehicle (and therefore it’s camera) will be susceptible to jostling while travelling, making feature detection difficult. At an excessive level of movement, the feature/s may not be present from frame to frame.   
  
Image stabilisation works by compensating for the movement of a vehicle by recognising that movement and applying a counter-movement in the opposite direction. As such, some image resolution is lost as the resulting usable image is of smaller dimensions than the original and the system will not work if the movement results in an entire frame being out of position compared to the previous frame.  
  
There are two main types of image stabilisation systems; Optical Image Stabilisation (OIS) and Digital Image Stabilisation (DIS) or a hybrid of these two approaches. We will concentrate on the first two methods.  
  
1.1.5.1 Optical image stabilisation (OIS)  
  
One solution to the problem of image stabilisation is to use optical image stabilisation (OIS) where an control system is placed within the camera lens (this being the ideal place as it acts as a force multiplier compared to the sensor location) and/or to the camera sensor. This technique is designed for countering the vibrations caused by hands shaking whilst holding a camera. If the system senses movement to the left (normally via a hall-effect sensor), it will move the lens or sensor to the right to compensate. This approach works well for small movements but limits on the mechanical range of the servometers (servos) used prevent this technique from compensating large movements and there may be time lags due to inertia of the detection and mechanical components. Another issue with this technique is that the cost to implement OIS in a camera system is high.   
  
An extension to this technique is to use a gimbal – an external system mounted to the camera system comprising of a gyroscope that is controlled by larger servos. This system copes with larger movements than an in-camera system but still suffers from mechanical range and inertia issues and requires a hysteresis system to prevent oscillations if a vehicle is moving fast with large displacements. This method may also prevent the camera system from looking ahead down a path if the vehicle is travelling on an incline or decline.

1.1.5.2 Digital Image stabilisation (DIS)  
  
Digital image stabilisation uses computers to perform the image stabilisation.  
Three main methods exist. Feature Mapping and Sensor Fusion or a hybrid of these methods.

##### DIS Method One. Feature Mapping

The first method is to use the same feature-mapping solutions used to determine obstacles but to select a static feature that can be tracked from frame to frame. One problem with this approach is that, as the vehicle is moving, the change in perspective will lead to distortion as shown at (Android Authority, n.d.-b). A potentially greater problem is that the purpose of image stabilisation is to enable feature detection, yet feature detection is required to stabilise the image, so the selection of which features to track is critical to the success of this method.  
  
Feature tracking is also computationally expensive and complex. Feature mapping has limitations on image size and the amount of displacement as increasing both increases processing requirements. Current research has focused on efficiency improvements such as the SIFT (Battiato et al., 2007; Chao et al., 2013) and SURF (Ljubičić et al., 2021; Shene et al., 2016) methods, along with other approaches such as affine-transform matrices (Mai et al., 2012; Mingkhwan & Khawsuk, 2017; Schwertfeger et al., 2011; Shen et al., 2009; Thillainayagi & Senthil Kumar, 2017) which compares separate video frames, particle filters (J. Zhu et al., 2016), linear and curve filters (L. Wang et al., 2012) and iFMI spectral registration (Schwertfeger et al., 2011). Hsu (Hsu et al., 2005) concentrated on hand-held camera shake using an inverse triangle technique while Morimoto (Morimoto & Chellappa, 1998) compared 4 DIS algorithms and found that simpler models performed better than more complex algorithms, due to them being less sensitive to tracking errors. Ljubičić et al. (Ljubičić et al., 2021) outlines freely-available software applications for applying digital image stabilisation. Once the direction and velocity of the camera movement has been determined, mathematical rotational and/or translational matrix operations are used to apply this vector in the opposite direction to counter the effects of the movement.

##### DIS Method 2. Sensor Fusion

The second method is to apply sensor fusion where camera movement can be compensated by an external sensor such as a gyroscope and/or accelerometer. This approach is similar to the OIS method except that the calculated movement is compensated for by translating the image in the opposite direction and velocity of the movement using software. The same matrix mathematical translations used with feature tracking are then applied. The most common sensor utilised with this approach is using an inertial measurement unit (IMU), a sensor combining a gyroscope and accelerometer together to give 6 degrees of freedom detection (6DOF). Adding a magnetometer gives 9DOF, adding an additional barometer/temperature gauge outputs 10DOF and adding a GPS sensor to that gives 11DOF.

For an inertial management unit (IMU) to compensate for unwanted movement in video streams, a vector of the resulting movement needs to be created so that the pixels in each image frame can be rotated/translated by the amount of unwanted movement.  
  
The gyroscope on an IMU outputs data in angular velocity which is a measure of rotation measured in angles of movement per second and is expressed in revolutions, radians or degrees per second per each axis of the three dimensions, x, y and z. An accelerometer measures acceleration in the same three dimensions and outputs data in m/s2.   
  
For a vector to be created, the rotation measured by the gyroscope needs to be converted into angles of roll (how much the vehicle tilts from side to side or rotation around the x axis), pitch (how much the vehicle rises and falls or rotation around the y axis) and yaw (how much the vehicle moves horizontally to the left or right or rotation around the z axis). These angles are commonly called euler angles. Note that the convention of x,y and z axes is not strictly defined, and one may place the x, y and z -axes in any direction that is convenient, providing the angled relationships between them remains fixed, but here, we will use the terms demonstrated in Figure 1.8.

Diagram of a truck with the same body and the same body

Description automatically generated with medium confidence

Figure 1.3 Diagram of roll, pitch and yaw angles. Image from (Fang et al., 2013)

The mathematical process to convert between gyroscope angles to euler angles incorporates integration which introduces a cumulative error, the size of which depends on the number of samples. Along with this problem is the issue of gyroscope drift, where the gyroscope values deviate over time due to temperature variances, gyroscope tolerances and external disturbances such as the jostling of a moving vehicle. The results of these two errors can be seen in Section 4.   
The general process of using an IMU accelerometer to measure euler angles is expressed in the following mathematical equations.  
At rest, gravity will be the only acceleration acting on the IMU, and assuming it is mounted so the z axis points straight down, the acceleration matrix will be:

Equation 1.1 Acceleration at rest where g=9.81m/s2.

The acceleration in a tilted frame will be

Equation 1.2 Acceleration on a tilted frame

where ai refers to the acceleration values for each axis.   
Euler angles are generally used (in particular the 3-2-1 set) referred to as yaw, pitch and roll.  
Using a direction cosine matrix, the relationship between the tilted frame and level frame is:

Equation 1.3 The mathematical relationship between level and tilted frames.

Solving for pitch and roll, we get:

Equation 1.4 Mathematical equations for pitch and roll

Using an accelerometer in this manner is subject to high levels of noise, as amongst other errors, any movement of the vehicle will corrupt the measured value of gravitational acceleration.  
  
Magnetometers can give an accurate measurement of the gravitational field but are influenced by other sources of magnetism, and in this case, as the camera and front IMUs are mounted close to an electric motor on a steel chassis, the magnetometer readings are not utilised.

To measure orientation using an accelerometer, an initial orientation must be known and then measurement values are integrated over time as acceleration is differentiation of speed. Integration of any errors will lead to increasing error components in the calculated value.

Once the roll and pitch values are known, a rotation matrix can apply these angles to our video frame data to transform the moving frame data to a purely horizontal reference as described by Odelga (Odelga et al., 2017).

This approach has been explored previously. Karpenko used the gyroscope on an iPhone to stabilise videos by iteratively calculating the differences between frames rather than directly using immediate orientation values (Karpenko et al., 2011). Jiang used an IMU to adjust both OIS (in strict x=0 and y=0 directions) and DIS stabilisation algorithms for the z-axis (Jiang et al., 2010) but did not provide real-world results.  
Stegagno et al used integrated IMU and Dense visual odometry pose estimation using a red-green-blue-depth (RGB-D) camera (requiring indoor operation) and pan-scanned the drone to increase field of view for a haptic-tele-operational drone. They proposed using a stereo camera for outdoor operation. Their experiment constrained the drone to 3 degrees of freedom, where roll will be present in this project, requiring 6DOF.  
Wiriyaprasat and Ruchanurucks (Wiriyaprasat & Ruchanurucks, 2015) used a full attitude and heading reference system (AHRS) unit (this has an onboard processor to provide attitude and heading rather than outputting raw gyroscope and accelerometer values as an IMU does) with Iterative Least Squared Error (ILSE) methods to modify a homography matrix to accommodate more rotational differences than a normal homography matrix. Calibration of the sensor to the camera was required as these were not installed in the same plane. A constant was required to change the pitch angle which they could not explain.  
Odelga’s team used both gyroscopes and accelerometers of the IMU on a drone and used the accelerometer values to compensate for the gyroscope drift, using a complementary filter. They also implemented fish-eye RGB-D lenses to get an improved field of view and compensated for the distortion introduced by this lens (Odelga et al., 2017).   
All the approaches listed above only considered single IMU units.

##### DIS Method 3. Hybrid of Feature mapping with sensor fusion.

The third method combines feature detection with IMU data to improve accuracy and speed of a feature detection system like Ryu and Auysakul (Auysakul et al., 2018; Ryu et al., 2010) who implemented an IMU to assist with a KLT tracker algorithm. Auysakul later used the IMU to drive servo motors as an OIS technique (Auysakul et al., 2019). The same issues with a feature system still apply with this method but accuracy and speed are increased. Chang et al (Chang et al., 2016) extended the use of an IMU by utilising a TLD (Tracking, Learning, Detection) multi-layer classification algorithm coupled with mechanical servos to compensate for the camera movement using OIS techniques. The main benefit of the TLD algorithm is the fuzzy logic controller algorithm dynamically updates the feature points. Utilising servos is only applicable if the motive movement and jostling are slow enough for the servos to compensate, and the servos have sufficient range of motion, both of which did not seem practical when implemented on a rover. As discussed earlier, feature detection is a computationally expensive process for the typically limited processing resources available on a rover.  
  
The common issue with all stabilisation systems is that a portion of the image will need to be cropped to act as a buffer for varying movement, the effects of which can be seen at (Android Authority, n.d.-a). If these effects are to be avoided, further cropping of the image is required, reducing quality. A direct approach to reduce the impacts of large jostling movements is to utilise a wide point of view lens such as a “fisheye” so a feature will stay in the frame even with large movements however this introduces significant lens distortion which will need correction in software.

A road with yellow paint on it

Description automatically generated

Figure 1.4 Fisheye Lens capture example. Image from (BHPhotoVideo.com, n.d.)

The fish-eye lens solution is a good solution for a fast-moving rover and is recommended for use in a production system but will not be evaluated in this experiment for simplicities sake.

Utilising multiple IMUs to compensate for errors in a single IMU has been investigated (Madgwick et al., 2013) but Madgwick only utilised accelerometers in the array.

It is possible that a Neural Network could perform this form of regression task of both single and multiple IMUs.  
The neural network or artificial neural network (ANN) or multilayer perceptron (MLP) has come back into fashion in recent years due to its use in many machine learning applications. Originally designed to mimic the biological brain, these supervised machine learning models can predict an output from given inputs and are optimised for classification (discrete outputs – typically a category of some sort) and/or regression (usually continuous outputs) tasks.   
  
A neural network is a collection of interconnected neurons that incrementally learn from data to capture both linear and non-linear trends to provide predictions for new situations containing even noisy and partial information.  
  
The basic unit of a neural network is the neuron. A neuron applies a weighted average function across input data and then applies a nonlinear “input/output” function as shown in Figure 2.2

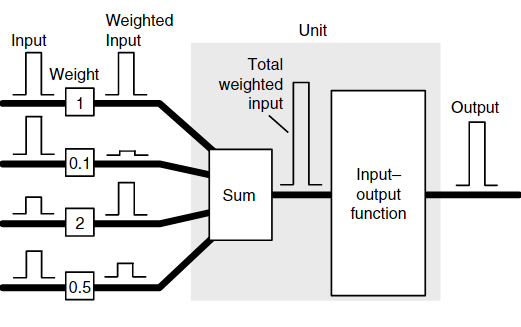


Figure 1.5 Neuron construction (unit part of the diagram). Image from (Samarasinghe, 2006).

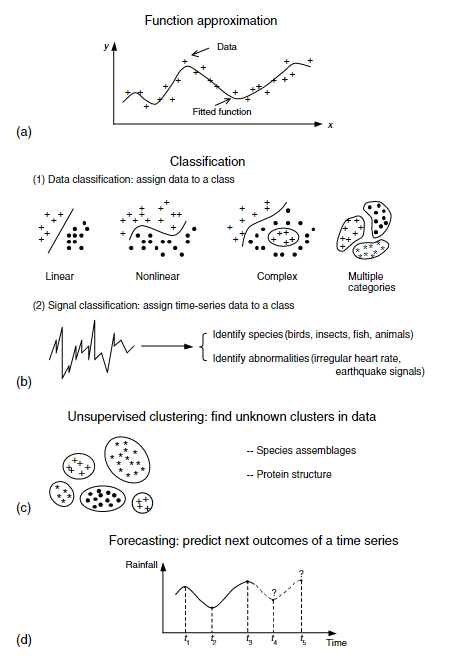
One of the inputs is usually given a value of 1 and called a bias input. This input is used to account for output effects that are not accounted for by the other inputs.  
  
Training data is used to train a network and must consist of input values and associated correct output values. Training involves iteratively randomly changing the value of the weights until the system learns to perform the task properly (the system’s output matches the training data output).  
The differences in output between the training data and the neural network’s outputs during training (or error) is determined by either simple subtraction for simple networks or a least-squared-error approach for multilayer networks. As the least-squared-error function is parabolic, the network calculates the error gradient in a downward direction along the curve to determine an optimal solution. A process called backpropagation is used to determine the error contribution of each weight and therefore how much each weight should change in the next iteration of the learning process.  
   
The overall function of a neural network is determined by the network structure, the connection strengths between neurons (called a weight) and what activation functions are used at each neuron.  
  
Some examples of the type of functions and their use is show in Figure 2.3   


Figure 1.6 Example functionality of neural networks for scientific data modelling: (a) fitting models to data or regression, (b) complex classification tasks, (c) discovering clusters in data, and (d) time-series forecasting. Image from (Samarasinghe, 2006)

Examples of the types of network topologies and their indicative uses are shown in Figure 2.4

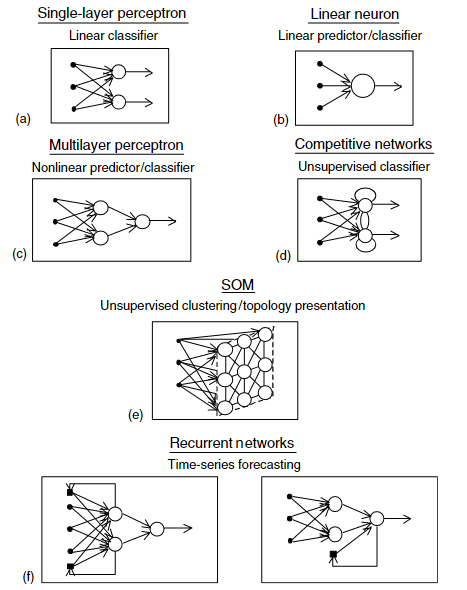


Figure 1.7 Some neural network topologies: (a) single-layer perceptron, (b) linear neuron, (c) multilayer perceptron, (d) competitive networks, (e) self-organising feature maps and (f) recurrent networks. Image from (Samarasinghe, 2006)

Multilayer Neural Networks have an input layer, an output layer and, commonly, one or more “hidden” layers between the input and output layers. These tend to be configured as supervised learning systems where valid responses are included in the training data. The weights of the neurons are adjusted until the difference between the correct output and the neural network output reach an acceptable level, as determined by utilising a least square error method along with determining the negative gradient of the error function.

Other types of neural network such as convolutional neural networks (CNN) exist which are optimised to create feature maps of data which can be used to determine shape and texture of data.  
A CNN neural network identifies features very well and is ideal for the classification of obstacles. For image stabilisation, a CNN network implementation was not deemed to be necessary, and a multilayer perceptron approach will be initially selected.   
A Recurrent Neural Network (RNN) is a network designed for recognising patterns over time. It is a different design than conventional neural networks where weighted inputs are applied to a function which then passes the function result to the next stage, in that outputs from some of the neurons are fed back into the inputs of the same neurons as a feedback mechanism. These networks provide a localised “memory” feature in the feedback-equipped neurons which makes RNN networks excel at relational problem solving tasks such as language processing but seemed excessive for a relatively simple regression operation.

The types of neuron input/output functions are varied and depend on the purpose of that neuron in the neural network.   
Common functions are a threshold function where the output is mapped to either 0 or 1 for use in classifying data.  
Competitive learning models various neurons become active and the ones that receives the largest input values “win” over the others and are weighted more. Over time the model can be said to become more sensitive to types of input data that they respond best to, and over time, various neurons become specialists at particular input types. This type of model is designed for competitive situations and/or self-organising feature maps and is not applicable to this project.  
  
Supervised learning uses a sample of the input/output data as training data and uses the following three basic techniques:  
1) forced learning, which is somewhat similar to the competitive learning model  
2) Reinforcement learning, where feedback is applied to improve the responses over time.  
3) Supervised learning, where the actual error differences between input and output are used to change weight values, based on an error gradient.   
  
An epoch is defined as one pass of all training data input patterns into the network. Many epochs may be (usually are) needed across the input data until the weight optimisation process is complete.  
  
Accuracy of a neural network is defined as the percentage of patterns correctly classified for each classification over the entire dataset. Once a suitable level of accuracy has been achieved the neural network is ready for live data. There are two main parameters of the network that are examined to ascertain accuracy. MSE is the mean square sum of any differences that appear between the input and the output of a neural network, compared to the expected training results. The MSE gives a measure of how accurately the network is able to reproduce a single set of output values from the input values. The Pearson Coefficient, R, is used in a neural network to measure how well the neural network understands the entire problem and reflects the accuracy of the trained algorithm across all inputs and outputs.  
  
As can be seen, neural networks are not magical and will not solve all problems. They require the practitioner to understand the datasets involved to design and configure a neural network system that will correctly classify/regress incoming data.

## Research Objective

The primary objective of this project is to determine whether multiple inertial measurement units in conjunction with a neural network can improve image stabilisation of a camera on an RC vehicle, compared with a single inertial measurement unit.  
A secondary objective is to determine the least number of inertial measurement units required to provide a significant measurable improvement.

## Thesis Outline

Chapter 1 (this chapter) provided an overview of the motivation, objective and results of the project. Chapter 2 discusses prior research that is relevant to this project, which involves the concept of a data mule, obstacle avoidance approaches, image stabilisation and neural networks. Chapter 3 describes the methods used in the project, including the approach taken and the resources and tools required. Chapter 4 presents the results of the project and these are discussed and analysed in Chapter 5. Chapter 6 provides conclusions and suggests paths for further research. Chapter 7 provides a place for Appendices and Chapter 8 contains a list of references.

# Literature Review

This chapter reviews current literature on the application of inertial measurement units (IMUs to image stabilisation problems. It outlines approaches that have been used and their benefits and drawbacks. The chapter concludes by identifying research gaps and identifying the research question(s) for this project.

As mentioned in the introduction, an IMU can be used to provide accurate positioning however using acceleration and magnetic data can introduce noise and mathematical processing artifacts (integration) and gyroscope data has inherent drift issues. Various methods have been proposed to counter these concerns.   
  
One mitigating factor is to obtain a gyroscope with a low zero-rate offset value so this integration error is minimised. Gyroscope selection is critical. (La Rosa et al., n.d.)  
  
  
Compensating for gyroscope errors with accelerometer (with a gravity vector) and magnetometer (with magnetic north) readings has been explored as the latter do not depend on prior state knowledge, as a gyroscope does.   
The most common algorithms for performing this fusion are complementary filters (Odelga et al., 2017; Yoo et al., 2011), Kalman filters and optimisation filters (Yean et al., 2018).

Complementary filters are computationally efficient and easy to implement but are subject to noisy and biased data and are susceptible to gyroscopic drift around the z axis and are thus not reliable or robust enough for regular use over time (Madgwick, 2010).   
Kalman filters provide very accurate orientation (Higgins, 1975; Marins et al., 2001; Sabatini, 2006) in the presence of large noise values such as persistent acceleration however are computationally expensive and complex (Yuan et al., 2019).   
Optimisation filters estimate a vector representing the sensor output at the current orientation and attempt to minimise the difference between predicted and measured results. [12]. Optimisation filters have acceptable accuracy with lower computational expense than Kalman filters but can suffer from unpredictable convergence (Fan et al., 2017; Mahony et al., 2005; Yean et al., 2018).   
To counter gyroscope drift and bias errors, accelerometer arrays have been proposed (Madgwick et al., 2013) but these lack real world testing and implementation.  
  
A software approach proposed by Odelga et al (Odelga et al., 2017) feeds both gyroscope and accelerometer data from a drone into a complementary filter, with a constant gain value, determined empirically to reduce the drift error without ignoring vehicle acceleration. The use of a complementary filter rather than the more widely implemented Kalman filter reduces processing requirements. Using this approach with a very wide-angle (“fish-eye”) lens permitted a wide viewpoint, allowing for compensation of very large movement displacements.   
Odelga found that the IMU solution worked well and utilising “fish-eye” cameras with large fields of view reduced the issue of a feature disappearing between subsequent images, but introduced barrel distortion which needed to be compensated for.   
  
In 2011 Madgwick et al introduced a gradient-descent algorithm commonly called Madgwick’s algorithm (Madgwick et al., 2011) which is more computationally efficient than Kalman filters and is currently widely used in industry (Fan et al., 2017; Yean et al., 2018) but requires two sequential minimisation steps (first on the magnetometer and then on the accelerometer) which can lead to slow convergence, the calculations of roll and pitch are not decoupled from each other which can lead to unpredictable orientation errors and a single adjustable parameter made it difficult to combine accelerometer and gyroscope values (Fan et al., 2017).  
  
In 2020 Madgwick et al extended a complementary filter in quaternion form (Euston et al., 2008) based on Mahoney’s work (Mahony et al., 2008) to increase robustness against noise while maintaining low computation cost and predictable convergence efficiency (Madgwick et al., 2020) and made this algorithm open source. (X-IO, 2008). It decouples roll and pitch data and varies the gain factor from a high value at initialisation to a lower running value to improve initialisation time.  
There are two variants of this algorithm, one in which magnetometer information is combined (IECF) and the other with only accelerometer and gyroscope values included (IECF6). Fan found that when magnetic disturbances were present, the IECF6 algorithm performed better. Fan also outlined that including magnetometer data was important to provide a single point of reference when using multiple IMU units (Fan et al., 2017). Yean’s approach of using a complementary filter to counter gyroscope drift, combined with a Kalman filter / gradient descent algorithm worked well only for slow and controlled ranges of motion, and so, is not suited for a bouncing vehicle.  
  
Madgwicks IECF6 algorithm will be used as the control factor in this project. This filter algorithm uses quaternion representations of angles internally to avoid the singularities caused by angles approaching π/2 radians. A single beta weighted-gain value acts as a fusion factor in a complementary filter, feeding accelerometer data into the gyroscope data to compensate for gyroscope drift. A gradient-descent algorithm enables performance at low sampling rates but, internally, the lowest sampling rate is recommended to be 10 Hz. The sampling rate of the experimental equipment averages at 8.3 Hz so it is expected that the Madgwick filter may not perform as well as the Kalman.  
  
The standard Matlab Kalman-based imufilter will be used as a comparison with the Madgwick results.   
  
Multiple IMU solutions have been proposed before (see above) but none have suggested using a neural network to dynamically compensate for magnetometer and other errors across multiple IMUs. The concept of a neural network has been covered in the introduction.  
Using a trained back-propagating supervised neural network with multiple IMUs should effectively compensate for drift and other errors present in a single IMU implementation and that is the model implemented in this project. Due to the computation requirements of a neural network, it makes most sense to apply the neural network directly to sensor readings, rather than complementing a filter algorithm, as the neural network should be able to determine the relationships.  
  
In Chapter 3, the experiment design and methods are outlined, in Chapter 4, the results from these experiments will be detailed and discussed, with analysis of these in Chapter 5 and conclusions reached in Chapter 6.

# Method

In this chapter, the project methodology is outlined. The theory behind the system design is presented first with the implementation details following. All angles used are expressed in radians.

## System Design

As the concept of applying a rotation vector to an image has been extensively covered by the afore-mentioned works, this experiment concentrates on accurately and efficiently deriving euler angles from the imu.  
  
The overall method employed is to capture IMU data at various states of motion and then analyse this data in Matlab to ascertain if using a neural network with both a single IMU and multiple IMUs offer any improvement in video stability processing compared to a single IMU using the Madgwick IECF6 and Matlab Kalman-based imufilter algorithms.

A base control system is established with a Universal Robots UR5 robotic arm, programmed for a set sequence of movements. This provides the reference point of true angles of rotation for comparison.  
  
The Matlab-based Madgwick IECF6 algorithm is applied to the IMU data. The gyroscope is set to a range of 0dps-250rps and the accelerometer is set to a range of 0g to 2g. An initial Beta (ratio of accelerometer to gyro data used) is set to the recommended value of 0.5. The accelerometer values are normalised to the gyroscope values internally within the algorithm.   
  
A Matlab-based Kalman imufilter is applied to the same data set with default settings.  
  
Finally, a simple 10-layer multilevel perceptron neural network will be used to compute euler angles.  
Different neural network topologies may need to be considered to increase accuracy.

The three different systems will be compared on speed of processing, processing CPU requirements and overall accuracy.   
  
Once a neural network model has been selected, a different set of robot movements shall be run and the three systems compared against the new previously unseen data.

## Method Employed

The contents of all scripts can be found in the github location mentioned in Appendix 1.  
  
The IMU’s gyroscopes and accelerometers will be calibrated using static measurements at various positions of 1.5808 radians on all axes, using recorded gravity as a control measurement. (ie Tilting the board by 90degrees in all directions). The robot data arm is to be calibrated with a 1.5m long spirit level.   
A python script (imudata.py) is then used on the Raspberry Pi to capture IMU data and a separate python script (GetRobotData.py) is run on a network-attached laptop to capture robot arm data.   
These two datasets are to be manually synchronised.   
Madgwick and Kalman filters (and, where appropriate, a Neural Network) are to be applied to the synchronised data.   
The results of these filters is to be compared and evaluated to determine the best performing system.   
As the neural network will have 30 inputs and the Madgwick and Kalman filters only support input data from a single IMU, the Madgwick and Kalman filter results will be averaged, to more closely align the resolution of the data. The filters (and Neural network model) are to be applied to a predetermined set of movements.  
Details of each movement pattern are given in Table 3.1 but are categorised as follows:

* Stationary vehicle (Madgwick and Kalman filters only)
* Rolling motion of vehicle (Madgwick and Kalman filters only)
* Pitching motion of vehicle. (Madgwick and Kalman filters only)
* Varied Combination of both pitching and rolling movements

Once trained, both filters and the neural network model will be applied to a new combination of varied movements that are different to the trained dataset.

## Equipment utilised

The data capture aspect of this project is implemented on a Raspberry Pi4b with 4Gb of RAM (Raspberry, n.d.), using Raspbian version 11 (Bullseye) operating system, running on a generic 16Gb MicroSD card, using the ext4 filesystem (mounted with no atime).  
The Pi is configured with ssh and I2C options enabled and synchronised to a reliable time source for accurate logging.  
Python 3.9.2 is used for data capture from the Sparkfun IMU-20948 IMU sensors (Sparkfun, n.d.) and Matlab 2023a is implemented on an HP Z230 workstation (quad-core Xeon E3-1270v3@3.50GHz CPU with 32Gb of DDR3 RAM and 1Tb Samsung 860QV0 SSD) running Windows 10 Pro 22H2 for data analyis. A Tamiya “Bruiser” 1/10 scale radio-controlled vehicle is used as the field-testing vehicle as the footprint of the model is smaller than the 1/5 or other scale model that would most likely be implemented and should more readily react to changes in topology than a larger vehicle (Tamiya, 2012). The unit is to be assembled as it comes as an unassembled kitset. A generic 2-channel RC radio system is used to drive the vehicle, but due to the limitation of two channels, this requires manual pre-selection of the gear when operating.  
A signboard (aluminium with a plastic layer on each side) board is mounted above the vehicle to ensure rigidity and that all IMUs are mounted in the same horizontal plane, simplifying calculations.  
The aluminium layer in the signboard acts as a paramagnet and so, with an electric motor mounted below the main sensors, the algorithm employed will only utilise gyroscopic and acceleration values from the IMU.  
  
Additional python modules to be installed are sparkfun\_qwiic, sparkfun-qwiic-tca9548a (to drive the multiplexer), board (to simplify addressing), adafruit-circuitpython-icm20x (to communicate with the ICM-20948 IMUs). Change the dtparam=i2c\_arm=on line in /boot/config.txt to read *dtparam=i2c\_arm=on,arm\_baudrate=1000000* and reboot as this will enable a 1Mbps baud rate, giving an effective sampling rate of the IMU units of up to 500kb/s.  
Install the opencv-python package (to drive the NoIR camera) from source via pip3 install git+https://github.com/opencv/opencv-python. (This avoids a known wheel dependency issue).  
Finally scipy (apt install python3-scipiy) and thenumpy and matplotlib pip libraries should be installed to aid in calibration.

## Initial baseline configuration

Initially a set of set squares was used to calibrate the IMUs which worked well (details in results baseline section) but a Neural Network requires training data that includes accurate results for each data point so a robot arm was sought.   
Lincoln University has one but the weight limit was insufficient to support the testing baseboard.   
Canterbury University has a Universal Robots UR5 robotic arm (Robots, n.d.) and this was used for both calibration and positioning to ensure that all methods employed could be verified against known angles.  
Robot arm programs were developed to move the robot to and from known angles to verify IMU positioning and both IMU and robot arm positions were captured and analysed.  
  
An initial baseline is developed to provide a control. The data for all IMUs is captured even though the initial baseline configuration utilises only the camera IMU data to mitigate against artifacts brought in by measuring the IMUs differently when the Neural Network is applied.  
  
The hardware is to be assembled and tested and the required python libraries installed.  
  
The UR5 robot arm needs to be calibrated first. This was undertaken with a 1.5 meter long spirit level. The decision to use a robot arm meant that the baseboard was screwed directly to the robot arm as mounting the actual testing vehicle would have been significantly more challenging to avoid the size of the vehicle limiting the range of movements the robot arm can travel.



Figure 3.1 UR5 Robotic Arm Base position showing mounted baseboard

Initial IMU calibration is to be undertaken to determine offset values to be applied to the measurements. The calibrate.py file in github contains the python code used to perform calibration. The calibration process uses the gravity readings from the IMU accelerometers to provide data for calibration. The calibration factors are to be stored and entered into the imudata.py program which reads in data from the IMUs and applies the calibration factors before writing data to a csv file.  
  
IMU data was obtained by using a laptop to ssh to the Raspberry Pi unit via an ethernet cable connected to a small 1Gb unmanaged switch. The switch enabled both the Raspberry Pi and the Robot arm Ethernet interface to connect simultaneously to the Laptop recording the data.  
The Laptop used was an HP Elitebook 850 G8 laptop with 16 Gb of RAM and an i7-1165G7 Quad-core CPU running at 2.8GHz. The operating system used was Windows 10 Pro 22H2., build 10945.3448 running the Windows Experience Feature Pack 1000.19044.1000.0.  
  
There was some difficulty in installing the ut\_rtde robot arm software for Python (wheel dependency issues) on the Windows laptop so an Ubuntu Services for Linux container was run on the Laptop and this container was used to run the GetRobotData.py program (Appendix 1) which connects to the robot arm via TCP/IP.  
  
Yaw movement data is not derived as the IMUs would realistically require magnetometer data to calculate this information and the aluminium baseboard (and steel construction of the vehicle chassis) would likely introduce too much variability to the measurements for a magnetometer to be useful. In production, fusing this data with a GPS unit would also allow location data to be obtained.  
  
The container and GetRobotData.py script to capture robot arm data should be started first, then start the imudata.py IMU data capture script (via an ssh connection to the Raspberry Pi) and finally the robot arm program should be started to move the robot arm. This gives a small window of time for the robot and IMU sensors to record stationary data before any movement occurs, making the manual alignment process simpler.  
  
A predefined path is programmed on the robot arm for each set of movements outlined in Table 3.1 and the IMUs and robot arm position data are polled during the arm movement.   
  
The capturing of all data is managed over TCP/IP. The laptop uses an ssh connection to the raspberry pi and a python program captures data directly to the SD card.  
While this is running, a a python script inside a Windows Services for Linux Ubuntu container gathers data from the robot listening on 192.168.1.100 via the ut\_rtde python module.  
  
The captured data files are copied from the Ubuntu Container and Raspberry Pi to a data directory and data is edited to remove any Python artifacts, if present. (ie brackets, etc). Outlying data is to be kept to verify real world performance.  
  
The captured IMU data during these movements is entered into Matlab and processed using the Matlab commands listed in “Matlabs commands-<movement\_type>.txt” in the Github repository.  
This will provide Matlab variables and plotting results that can then be used to align the IMU and robot arm data. The Matlab command file needs to be edited for the correct filenames and for imu and robot data alignment.  
  
Alignment of the IMU and Robot arm data is a manual process.  
The data sample number where camera IMU movement starts (obtained by examining the associated Matlab camera IMU array) and the data sample number where the robot arm shows initial movement (via examining the robot entry array) should be entered into the Matlab commands script. An offset value should also be selected which is the number of robot arm sample lines to skip and is a factor of the robot sample rate compared to the IMU sample rate This offset value should be approximately 258 although this needs verifying by trying different values.  
To verify the offset value, replot the non-averaged plot sequence found near the end of the Matlab commands file with all values of data points, rather than the more limited range used to highlight values. The offset value is value that gives the best fit of aligning the calculated Kalman and Madgwick results with the robot movements across the entire dataset.

Once the data is aligned then the Matlab command script should be re-run with the values derived and the resulting plots examined.

The script provides a section at the top of the file to specify captured data filenames and the amount of data sample in the sequence.  
Another section (approximately line 150) is where the data sample numbers of initial movement of the camera IMU and robot arm and the robot line offset value are entered.

Table 3.1 Angles of movement per motion test

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Move | Pos1 Roll : Pitch | Pos2 Roll  :  Pitch | Pos  3 Roll  : Pitch | Pos 4 Roll  : Pitch | Pos 5 Roll :Pitch | Pos 6 Roll : Pitch | Pos 7 Roll : Pitch | Pos 8 Roll : Pitch | Pos  9 Roll :Pitch | Pos  10 Roll :Pitch |
| Roll | 0 : 0 | -.7854: 0 | -1.047: 0 | Loop  x 5 | LoopA 1.047 : 0 | Loop B -1.047 : 0 |  |  |  |  |
| Pitch | 0 : 0 | 0 : .5236 | 0 :0.7854 | 0 : 1.047 | 0 : -.5236 | 0 : -.7854 | 0 : -.1047 |  |  |  |
| Mixed1 | 0  : 0 | .2618 : .5236 | .5236 : .7854 | .7854 : .5236 | .7854 : .7854 | .5236 :  0 | .7854 : .5236 | 0  :  0 | -2.094 : .5236 | 0 : 0 |
| Live (TBC) | 0 : 0 | 60 : -30 | 30 : -45 | -60 : 30 | 60 : -30 | Loop x 5 | Loop  A 30 : -30 | Loop B -15 : 30 |  |  |

The script performs the following functions:  
Loads the csv data from the files and converts these to Matlab arrays.  
Aligns all camera data to use the smallest number of entries of the sets of data.  
Performs the alignment of robot arm data to camera IMU data based on the values selected.  
For each set of IMU data:  
 Calculate roll and pitch via the Madgwick method 10 times to enable processing time capture.   
 (Matlab timing accuracy is not defined for measurements below 0.1 seconds).  
 Calculate roll and pitch via the Kalman method  
Obtain average values of roll and pitch across all IMUs.  
Plots the results of the camera IMU results and the averaged results.  
  
The Madgwick filter selected is the freely available Matlab implementation provided by the author of the filter. The “Initial release” version of this code was used (dated 28/09/2011). (Madgwick, 2009)  
  
The Kalman filter selected is an internal Matlab “imufilter” with default settings unless specified otherwise. The version of Matlab used shall be the 64bit version 2023a (build 9.14.0.2206163).  
  
Best practise for neural network design is to start with a simple model and alter this if it proves insufficient for the task. The Matlab graphical neural network builder, “nnstart” will be used to develop the initial neural network using the “Fitting” model design (this best suits the regression task we are attempting). This will create a 6-input, 2-output 10-layer neural network using sigmoid functions for the inner layers and a final linear function for the output layer, shown in Figure 3.2.  
By default, the system will use a random division of data into training, validation and testing sets of 70%, 15% and 15%, respectively. The training algorithm used will be Levenberg-Marquardt with the mean-squared-error performance method.

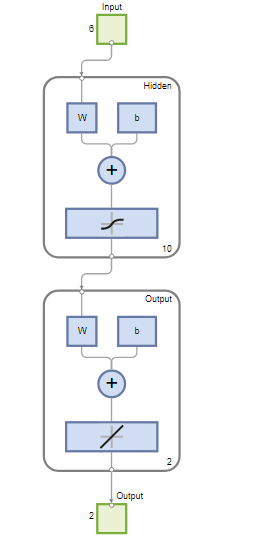


Figure 3.2 Initial Design of Neural Network using nnstart

Initial neural network operation should only analyse the camera IMU input. Training of the data should result in low mean square error (MSE) and high Pearson correlation coefficient (R) values.  
The model may need to be altered to accomplish good results with minimal computing resources.  
  
Once the best performing model has been selected, the live training data set (containing data not seen by the nn up until this point) should be used to validate real world performance and a comparison made against the Madgwick and Kalman filters.  
  
Tests on the front 3 (three) IMU results and across all 5 (five) IMUs should then be undertaken (this may require a redesign of the model to achieve satisfactory results) to explore if multiple IMU data offers any improvements to the fitted data compared to the extra computing resources required to process this additional data.  
  
In Chapter 4, the results from these experiments will be detailed and discussed, with analysis of these in Chapter 5 and conclusions reached in Chapter 6.

# Results

## Initial data capture and calibration

A sample of the raw IMU data taken from the stationary vehicle is shown below in Figure 4.1. All IMU data was captured but only the Camera IMU data is displayed in this figure. The large value of the z (vertical) axis is a measure of gravitational acceleration. This was measuring an average of 10.22617 m/s2 instead of the expected 9.80665m/s2. The maximum variation between sea level to the top of Mt Everest is approximately 0.02m/s2 so the discrepancy does not relate to altitude. (These measurements are taken at approximately 19m above sea level). Varying the mounting positions of the sensors slightly by tightening the mounting nuts changed these values by significant amounts, demonstrating that sensor positioning is of critical importance for accurate results. Once the sensors were mounted firmly on the chassis board, results showed that some sensors were mounted incorrectly in terms of consistent X and Y directions. The incorrect sensors were remounted, tightened down and readings taken again. Results from this initial calibration can be obtained from Appendix 2. To reduce the impact of outlying values, the 95th percentile values of the data was derived and used to calculate the average value of the data. The first three lines of data are shown in Table 4.1.

Table 4.1 First three lines of data from the Camera IMU

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Accel\_X average ms/s2 | Accel\_Y average ms/s2 | Accel\_Z average m/s2 | Gyro\_X average degree/sec | Gyro\_Y average degree/sec | Gyro\_Z average degree/sec |
| -0.19154 | 0.11971 | 10.24718 | 0.027712 | 0.006262 | -0.01092496 |
| -0.1652 | 0.043096 | 10.27831 | 0.031043 | 0.004663 | -0.01652067 |
| -0.17957 | 0.050278 | 10.20888 | 0.033441 | 0.001732 | -0.0085268 |

A Jaycar KJ8916 robotic arm (Jaycar Ltd, 2023) was initially implemented to validate these measurements but this proved to be incapable of the task as the weight limit was 100g and the board, without battery, weighed 486g. On top of this, the KJ8916 robotic arm had significant play and movement was in approximately 1 degree increments on the main stepper motor.  
A set of setsquares (permitting fixed angles of 0,30,45,60 and 90 degrees) was initially used to calibrate the individual IMU sensors to apply appropriate offset and bias values (in the case of the accelerometer sensor).  
The gyroscope calibration is easiest to determine so is dealt with first.  
At rest, all gyroscope readings should be zero so averaging the deviations from this should suffice for a bias offset value in each axis direction. Table 4.2 lists gyroscope offset values for the Camera IMU.

Table 4.2 Initial Camera IMU Gyroscope Offsets

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Axis | Flat, Facing Up | Flat, Facing Down | 90deg, tilting left | 90deg, tilting right | Tilting Up | Tilting Down | Average Offset |
| Gyro\_X | 0.029521 | 0.029307 | 0.028777 | -0.02387 | 0.028599 | -0.02385 | 0.029078 |
| Gyro\_Y | 0.004314 | 0.004279 | 0.018543 | 0.018462 | 0.003848 | 0.018154 | 0.004103 |
| Gyro\_Z | -0.01118137 | -0.0112 | -8.4E-05 | 0.000296 | -0.01144 | 0.000378 | 0.00743 |

Table 4.3 lists the Accelerometer offsets for the Camera IMU. These were obtained from the axes shown in the table and the Matlab polyfit function (Matlab, 2023) was used to determine the first order equation parameters of slope and offset. X values used were -1, 0 and 1.

Table 4.3 Accelerometer Slope and Offset Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Axis | Flat, Facing Down (gravity = -1) | 90deg, tilting left  (gravity = 0) | Flat, Facing Up (gravity=1) | Slope | Offset |
| Accel\_X | 0.02984 | 9.720203 | -0.20886 | 0.0013 | 0.0315 |
| Accel\_Y | -0.10395 | -0.59628 | 0.032943 | -2.338e-4 | 0.0325 |
| Accel\_Z | -9.4977 | 0.368226 | 10.22617 | -1.0244e-4 | 0.0325 |

The offset and scaling values from Tables 4.2 and 4.3 are applied to each measurement to reduce calibration errors.   
Variation in the output matters as it may mask any differences in technique. Note that there is some considerable variation in the output for a completely stationary vehicle, indicating high noise floor of the sensors. The Madgwick, Kalman and Neural Network processes are meant to reduce these errors, providing they are gaussian in nature.  
  
The UR5 robot joint angles themselves are not calibrated and show slight errors when manually compared to a 1.5m long spirit level. It is believed that, in this instance, this is mainly due to the visibly obvious bow in the wooden table used to mount the robot. These were initially compensated for but meant that the robot sensor data showed slight variations away from true positions. Using tool center point (TCP) sensors on the tool section of the arm initially showed true values of the pose compared to the spirit level however so the TCP sensors were used first in initial measurements. When movement occurs, it becamed evident that the TCP sensors calculate positioning based on weight and size of the tool mounted on the arm which introduced some subtle errors. The actual joint angles of the arm components were used instead to obtain angular positions.  
  
When operating the robot it was seen that there is some tolerance and play in moving repeatedly between the same defined angles. In section 5.1 of the user manual, worst case joint position accuracy is given as 1.15 degrees with a detection time of 100ms. The tool center point (TCP) sensors have a worst-case accuracy error of 20mm in positioning and 1.15 degrees in orientation. These are worst-case values, however, and the robot operated well within these limits.

With the robotic arm in the neutral position as shown in Figure 3.1, a sample of static results from the robot arm sensors is shown in Table 4.4.

Table 4.4 Sample of Euler angle results using tool arm sensors while at rest.

|  |  |  |
| --- | --- | --- |
| Euler\_Z | Euler\_Y | Euler\_X |
| |  | | --- | | 0.278238171 | | |  |  | | --- | --- | | 0.52853157 |  | | -90.0214431 |

Table 4.5 Joint angles of Robot at rest

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Angle | Base | Shoulder | Elbow | Wrist 1 (Roll) | Wrist 2 (Pitch) | Wrist 3 (Yaw) |
| Radians | -0.0002992788897913101 | -1.570928399 | -2.3667E-05 | 8.39233E-05 | 1.5705477 | -7.230440248662262e-05 |
| Degrees | -0.017147417 | -90.00756717 | -0.00135602 | 0.004808453 | 89.98575473 | -0.004142737 |

The various orientation determination methods were applied with the following results.

## Filter Performance on a stationary vehicle

Applying the Matlab Madgwick function from Madgwick’s website (Madgwick, 2009) (code dated 28/08/2011) produced the following results on a stationary vehicle, demonstrated in Table 4.7. Settings for the Madgwick filter were left at defaults. These defaults were a sample period of 1/996, and a beta gain value of 0.1. The time taken to process 996 sensor readings from the camera IMU was 0.0427 seconds using the tic/toc method in Matlab. As the code runs faster than the recommended minimum 1/10 second code runtime recommended for the tic/toc approach, the code was looped 10 times and the tic/toc result is an average. The results of the Madgwick filter are shown in Table 4.7.

Table 4.6 Madgwick filter results from a stationary vehicle.

|  |  |  |
| --- | --- | --- |
| Angle X (Roll) | Angle Y (Pitch) | Angle Z (Yaw) |
| -0.2286 | 0.0278 | Varied from -0.0194 to 0.0217 |

A graph of the calculated data is shown in Figure 4.1.

A graph on a white background

Description automatically generated

Figure 4.1 Madgwick filter results on stationary vehicle (Brown=Roll,Yellow=Pitch,Blue=Yaw)

The Yaw value shown in Figure 4.1 has a descending value over time due to the double-integration involved in producing yaw values from gyroscope information and clearly show that calculating yaw without magnetometer data is fraught with errors. The roll and pitch values vary slightly but values are within the noise floor of the sensor data.

The default Matlab imufilter filter was used to represent an error-state Kalman filter with the properties shown below. These properties were configured from experimentation with the datasets to obtain the best results, matched to the robot arm data.  
  
FUSE=imufilter(ReferenceFrame='NED',SampleRate=8.33,AccelerometerNoise=0.16,GyroscopeNoise=0.03,GyroscopeDriftNoise=3.04622e-2,LinearAccelerationNoise=0.0096236,LinearAccelerationDecayFactor=0.1,OrientationFormat='quaternion');

Running the Kalman filter above on the 996 sensor readings took 0.7581 seconds and produced an array of euler angles after the conversion. Figure 4.2 shows the results and indicates that the Kalman filter also can not handle yaw values (blue line) without magnetometer input. As mentioned above, yaw values are not handled well by filters as, without magnetometer data, there is no fixed reference point to provide accurate values. Figure 4.3 shows a magnified view of the area around the x axis showing that the Matlab Kalman filter will produce accurate results to within 0.03208 degrees for roll values and within 0.0257 of a degree for pitch values. The Kalman filter processing time is considerably longer than the Madgwick filter processing time (0.7581 vs 0.0427 seconds).

A graph of a number of people

Description automatically generated with medium confidence

Figure 4.2 Kalman Filter results from a stationary vehicle

A graph showing a number of results

Description automatically generated

Figure 4.3 Magnified view of Kalman filter results from a stationary vehicle.

The robot arm was not sampled for this experiment as expected values are somewhat obvious.  
It was not believed necessary to employ a Neural Network model on a stationary vehicle.

## Filter performance on a rolling vehicle.

The robot arm was rotated according to the appropriate angles in Table 3.1 associated with the roll test and the Robot arm data was synchronised with the IMU data via manual alignment. The single camera IMU data produced 1001 readings in the 120 second period.

The Madgwick filter was run on the camera IMU data with standard settings of a sample period of 1/1001 and a default beta gain value of 0.1 and took 0.0417 seconds to run.  
The results were clearly incorrect as can be shown on Figure 4.4 below. Increasing the beta gain uses the accelerometer values to compensate for the gyroscope values and results from various beta values are shown in Figures 4.5 through to 4.7.

A graph of a graph

Description automatically generated with medium confidence

Figure 4.4 Madgwick results from a stationary vehicle with a default gain of 0.1

A graph of a graph

Description automatically generated

Figure 4.5 Madgwick Filter on a rolling vehicle with a gain of 1

Using a gain of 10, the Madgwick filter took 0.0440 seconds but produced better results as can be seen in Figure 4.6.  
A graph of a graph

Description automatically generated with medium confidence

Figure 4.6 Madgwick filter on a rolling vehicle with a gain of 10

By repeated experimentation, a gain of 15 was the lowest gain required to produce the acceptable results show in in Figure 4.7. This clearly indicates that the gyroscope output is either noisy and/or has significant drift. From looking at the data, and from examining the output of the gyroscope when the vehicle is stationary, it can be seen that the amount of noise is significant. This operation took 0.04997 seconds for the camera IMU. The average elapsed time for all IMUs was 0.046592 with a median of 0.04319 and max and min values of 0.06516 and 0.03855, respectively. The results from Figure 4.7 show a maximum measured value of .9948 radians instead of the expected +/- 1.047 radians and the 0.7854 radian position was calculated as 0.7487 radians. If the 0.0201 radian maximum tool angle error plus the IMU’s own 1.5% inaccuracy values are combined, then this error is within the operating conditions of the sensors.

A graph of a graph

Description automatically generated

Figure 4.7 Madgwick filter on a rolling vehicle with a gain of 15

A Kalman filter was implemented using the standard Matlab imufiter function using the following parameters:  
FUSE=imufilter(ReferenceFrame='NED',SampleRate=8.33,AccelerometerNoise=0.6,GyroscopeNoise=0.3,GyroscopeDriftNoise=3.04622e-2,LinearAccelerationNoise=0.0096236,LinearAccelerationDecayFactor=0.1,OrientationFormat='quaternion');   
  
The filter operation took 0.6516 seconds for the camera IMU and produced the output shown in Figure 4.8. The average elapsed time for all IMUs was 0.3772 with a median of 0.3051 and max and min values of 0.6516 and 0.2905, respectively, taking 8.0958 times longer than the Madgwick filter. Smoothing of this data should be applied to these results but they track the movement of the vehicle adequately (Varying the Kalman filter settings did not show any significant difference in the output results). The average calculated value for the desired 1.0472 angle is 1.0297 which is well within specifications of the sensors and tracks better than the 0.9948 reading of the Madgwick result. It is interesting to see the various phasing differences between the values. This is most likely because of the different timings of the robot arm versus timing of how the different IMU units are polled.   
The multiplexer switching time was measured as taking an average of 515 nanoseconds at the start of the movement set down to 41 nanoseconds after 30 seconds of operation. After 1 minute of operation, the multiplexing switching time was measured at 47 milliseconds. The time taken to read acceleration data is 2202 milliseconds and the time taken to read gyroscope data is 1659 milliseconds. Total time taken to measure each IMU was 2400 milliseconds. Between each measurement the script takes 71 milliseconds (excluding multiplexor switching time) to begin the processing of the next set of measurements. The robot arm is sampled at 6403 readings per second which equates to a time of 0.15625 milliseconds per reading. At this speed, TCP/IP congestion must be considered as a possible influence on the measurements. Reducing the payload of the robot data packet and a separate network for both robot data and IMU data should be considered for evaluation however, in production use, all IMU measurements will reside on the baseboard unit which will eliminate any network congestion issues. A faster computer than the Raspberry Pi should also be considered for production use.

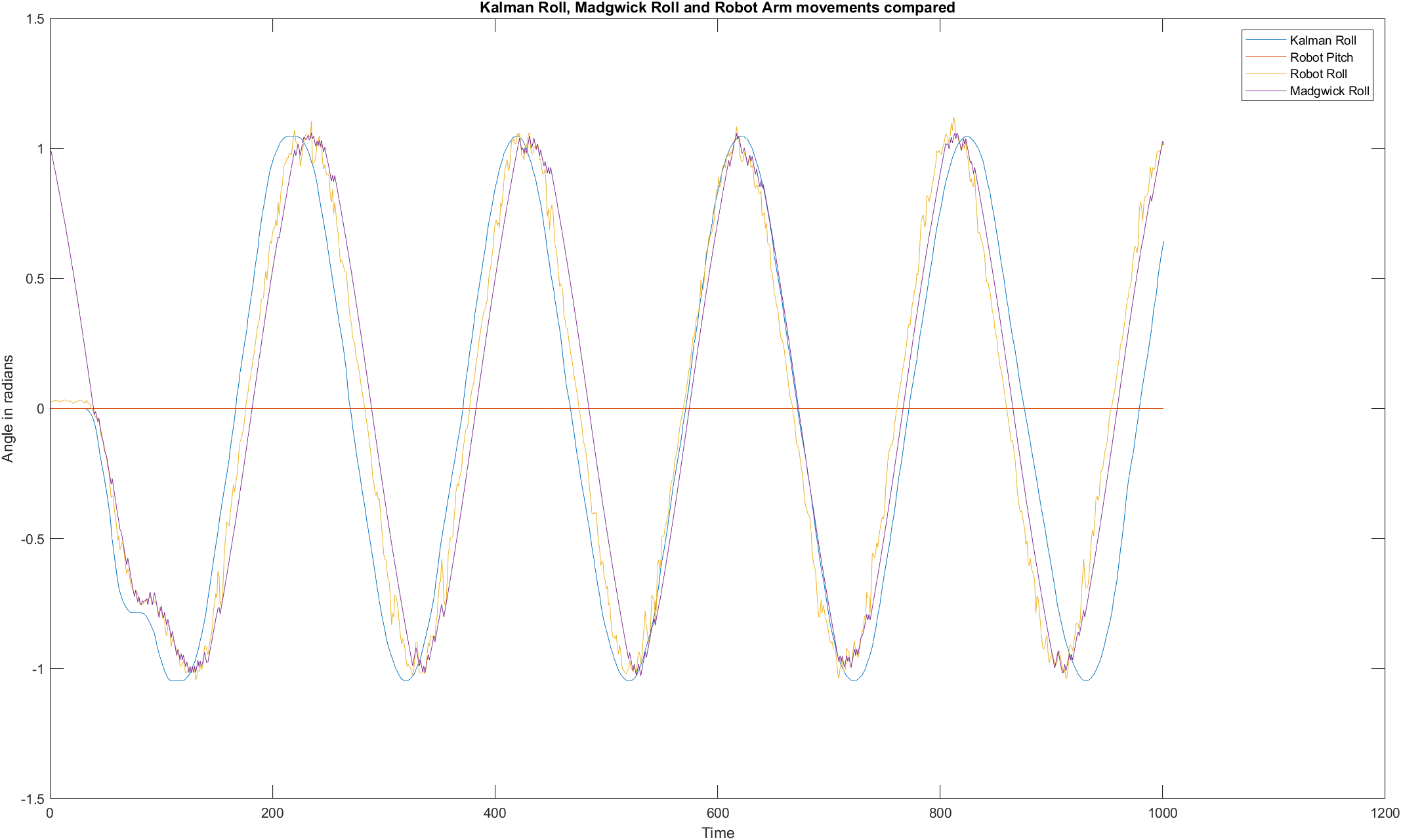


Figure 4.8 Filter and Robot arm data for a rolling vehicle using Camera IMU values.

A graph of a graph

Description automatically generated

Figure 4.9 Averages of Filter Roll values against robot arm movements

Plotting the mean of the Madgwick IMU values and the mean of the Kalman IMU values against the robot arm movements shows that both phasing and amplitude change as seen in Figure 4.9. This indicates that, as somewhat expected, using different IMUs for alignment against the robot arm will change the phasing discrepancies and using an average of the data significantly decreases peak amplitude in both positive and negative peak values, indicating a wide spread of values obtained from the various IMUs. In terms of tracking the robot arm movements, both the Kalman and the Madgwick filters make good attempts and there is little between the filter results to merit the use of the more computationally expensive Kalman filter. <TBC>Explain phasing differences.

It was not deemed necessary to configure a Neural Network for a simple rolling movement.

## Filter performance on a pitching vehicle

The robot arm was rotated according to the appropriate angles in Table 3.1 associated with the pitch test. The Robot arm data was synchronised with the IMU data via manual alignment. The single camera IMU data produced 995 readings in the 120 second period.

Applying the Madgwick filter with a beta of 15 on the gathered camera IMU data took an average of 0.04758 seconds. The average elapsed time for all IMUs was 0.0432 with a median of 0.0431 and max and min values of 0.04758 and 0.0401, respectively.  
The Kalman filter took 0. 4268 seconds on the same data. The average elapsed time for all IMUs was 0.3271 with a median of 0.2991 and max and min values of 0.4268 and 0.2897, respectively.  
Using the average values, the Kalman filter took 7.568 times longer than the Madgwick filter to process the pitch values.   
Figure 4.10 shows the results of the filters compared to the robot arm data.  
A graph showing a graph

Description automatically generated with medium confidence

Figure 4.10 Filter and robot arm data for a pitching vehicle using camera IMU values

Figure 4.11 shows the mean of Madgwick and Kalman results from all IMUs against the robot arm data.

A graph of a graph

Description automatically generated

Figure 4.11 Averages of Filter Roll values against pitching robot arm movements

Averaging the values over all IMU pitch results produced errors in amplitude and phase, concentrating at zero crossing points, as seen in Figure 4.11. The Madgwick filter in particular shows some considerable oscillation around the zero crossing points when averaged that did not appear when using the single camera IMU. The Kalman filter faltered at an average of -0.655 radians while the Madgwick filter oscillated between this value and between +0.589 and +0.606.  
These results indicate there are some significant data discrepancies between the IMU units and/or variations in the input data between the IMUs.

It was not deemed necessary to configure a Neural Network for a simple pitching movement.

## Filter Performance on a varied roll and pitch vehicle

The robot arm was rotated according to the appropriate angles relating to the varied movement test outlined in Table 3.1. The Robot arm data was synchronised with the IMU data via manual alignment. The single camera IMU data produced 4998 readings in the 599.5 second period.

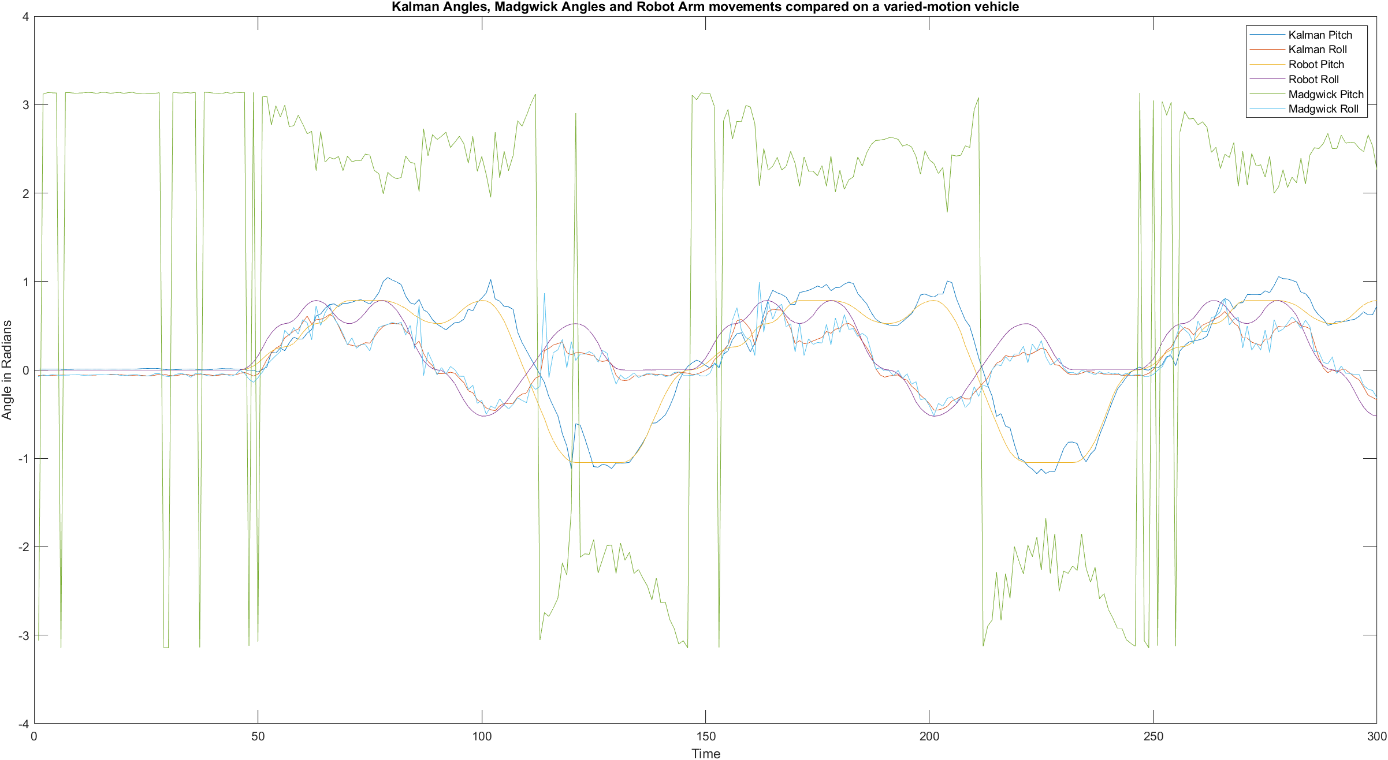
The Madgwick filter was applied to this data with standard settings of a Sample Period of 1/8.3333 and a default beta gain value of 15 and took an average of 0.2626 seconds to run. The average elapsed time for all IMUs was 0.2610 with a median of 0.2626 and max and min values of 0.2675 and 0.2532, respectively. Figure 4.12 shows these results, along with the Kalman results discussed below.  
  
The Kalman filter previously used was applied to the data and the output is also presented in Figure 4.12.  


Figure 4.12 Filter and robot arm varied-motion data using a Madgwick sample rate of 8.33

The results shown in Figure 4.12 indicate that the Kalman filter adequately tracks the robot arm movement, as does the Madgwick roll component, but using the actual sample rate as the sampling rate in the Madgwick filter is clearly producing nonsensical results. In Figure 4.13, a sampling rate of the size of the sampled data (4998 samples in this case) is selected, which is also clearly not sensible.

A graph of a graph

Description automatically generated with medium confidence

Figure 4.13 Filter and robot arm varied-motion data using a Madgwick sample rate of 4998.

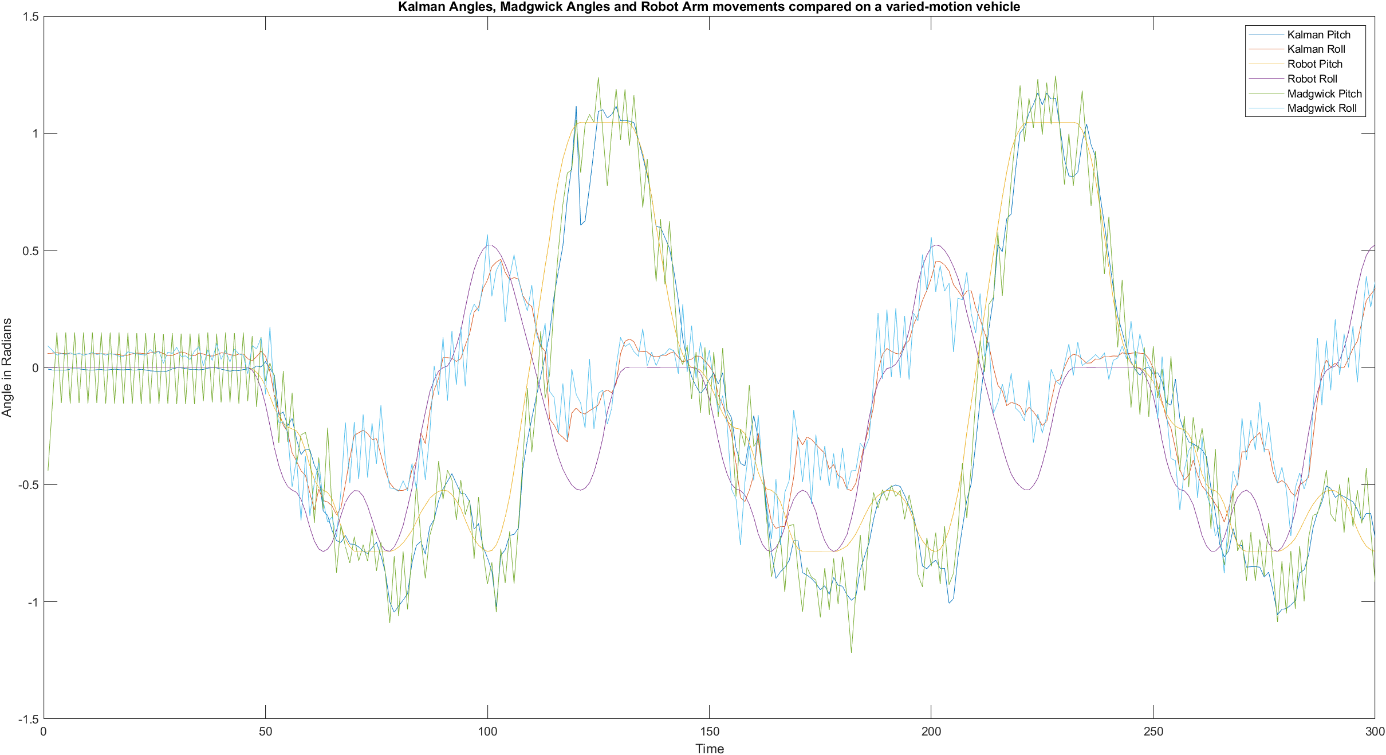
Since the Madgwick filter operates by integrating past data, it was surmised that setting the sample rate to the number of time periods used to describe a complete movement set would be ideal. For this set of movement data, the time period is approximately 100. Utilising this value in the Madgwick filter parameters produced a better result as can be seen in Figure 4.14 but does indicate the importance of setting the correct sampling rate. Too short and the descent algorithm overfits and too long and the descent algorithm underfits.   


Figure 4.14 Madgwick and Kalman values with a sample rate of 100

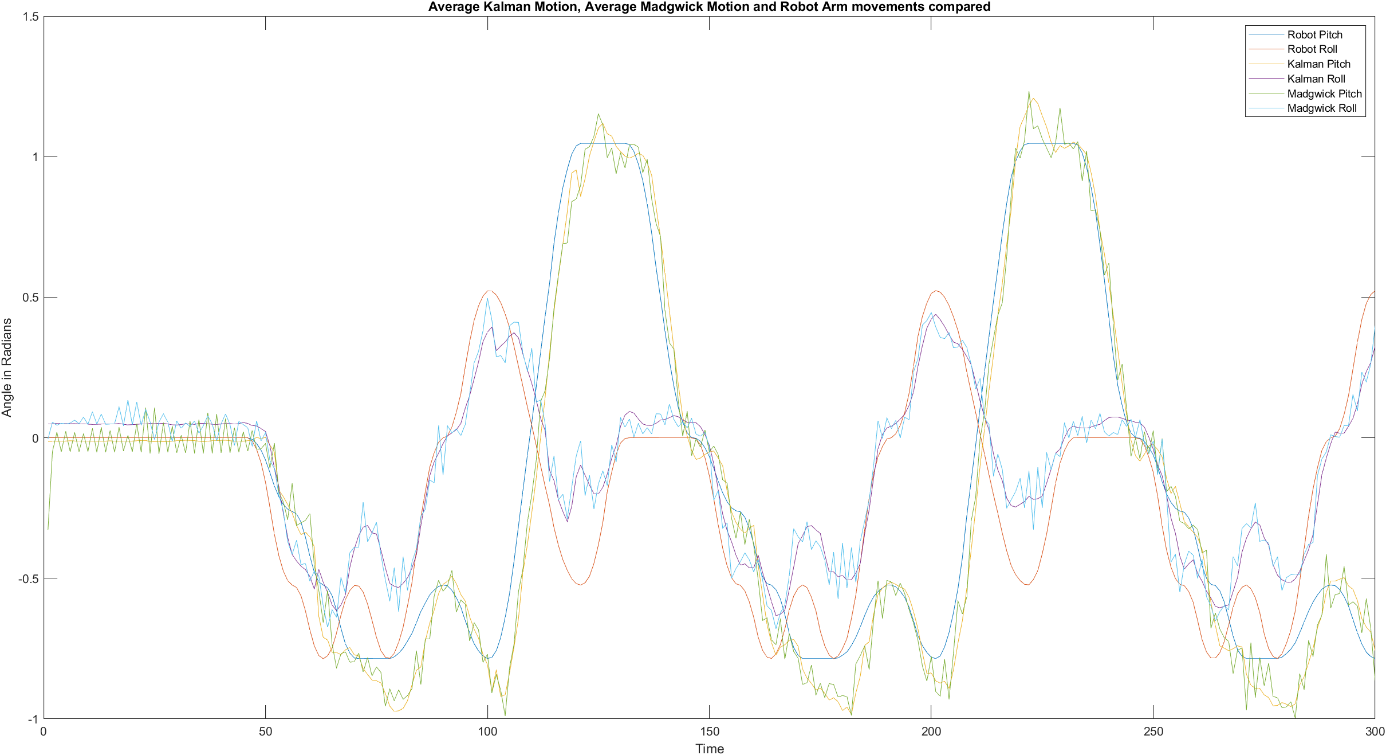
Averaging the IMU results reduces the noise somewhat as shown in Figure 4.15.  


Figure 4.15 Averages of Madgwick and Kalman values with a sample rate of 100

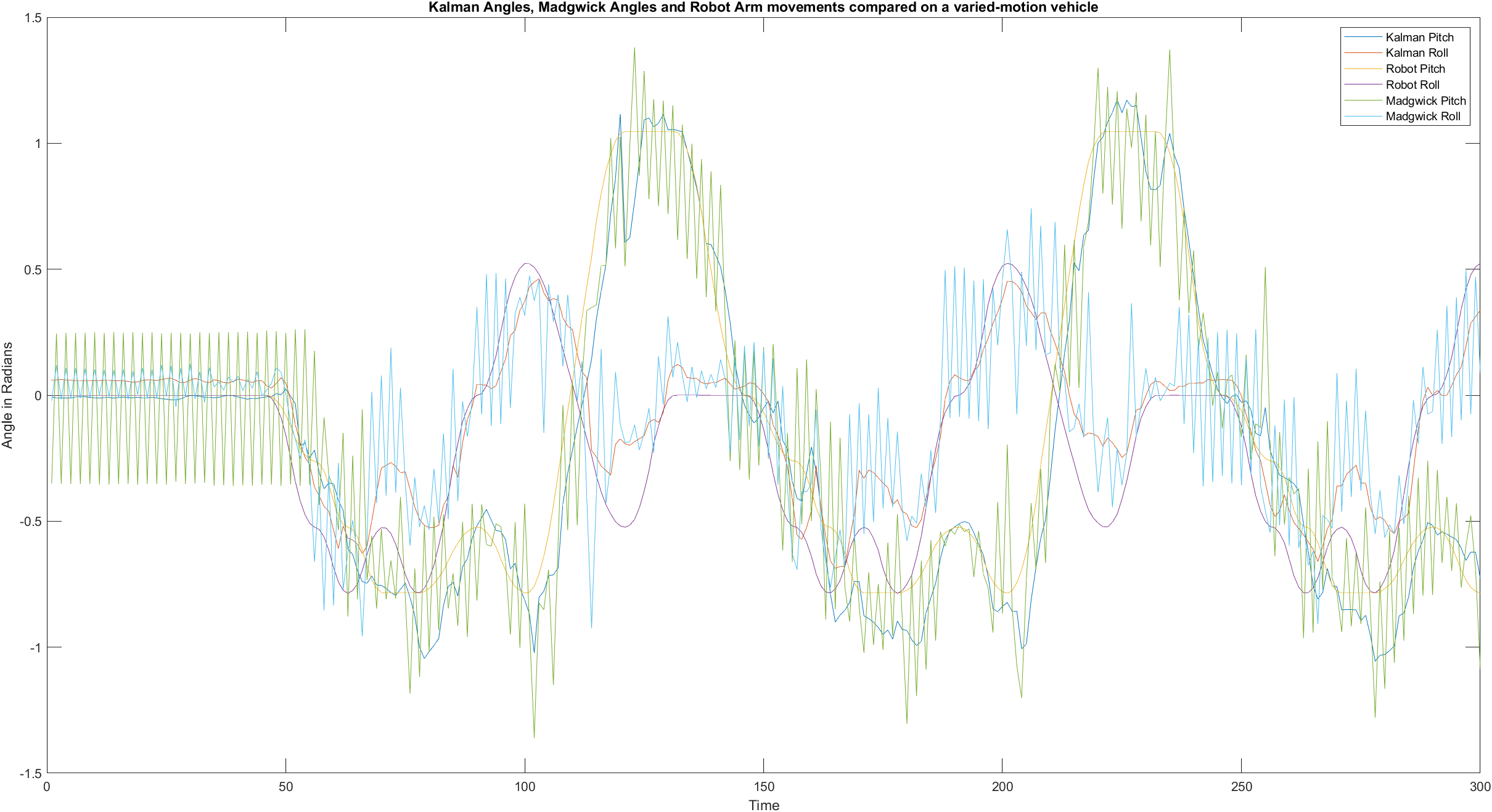
Experimenting with different sample rate values (Figure 4.16 and 4.17 show two examples of this experimentation) indicated that the best fit was the surmised opinion that the sample rate should contain a set of samples that describe a group of movements. Over a long period of time the Madgwick method will become increasingly inaccurate without a form of reset. In production, this will prove difficult to implement reliably and will likely need to be adjustable based on implementation. Averaging the results across IMUs reduces noise which partially mitigates the issue but the problem still remains when sampling over a long period.

Figure 4.16 Madgwick and Kalman values with a sample rate of 50

A graph of a graph

Description automatically generated with medium confidence

Figure 4.17 Madgwick and Kalman values with a sample rate of 500

The interplay between sample rate and beta value was explored in Figures 4.18 and 4.19. The sample rate was left at 100 and the beta value was altered down from 15 to the configured default of 0.5 (Figure 4.18), and to 1/3 of the previously-determined value of 15, which is 5.0 (Figure 4.19).

A graph of a graph

Description automatically generated with medium confidence

Figure 4.18 Madgwick filter beta value set to 0.5 with a sample rate of 100

A graph of a graph

Description automatically generated with medium confidence

Figure 4.19 Madgwick filter beta value set to 5.0 with a sample rate of 100

## Neural network using a single IMU.

Feeding the measured gyroscope and accelerometer data from the camera IMU into the initial design of the neural network, and taking all network defaults, the network was trained and the results obtained are shown in Figures 4.21 to 4.24 and Table 4.7.

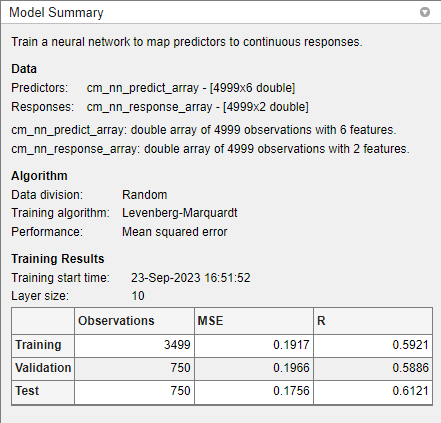


Figure 4.20 Neural network Model training summary

Table 4.7 Initial Neural Network training results with default layer size of 10

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 17 | 1000 |
| Elapsed Time | - | 00:00:01 | - |
| Performance | 1.31 | 0.191 | 0 |
| Gradient | 2.62 | 0.00533 | 1e-07 |
| Mu | 0.001 | 1e-05 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |

A graph of a graph

Description automatically generated with medium confidence

Figure 4.21 Initial Neural Network Performance plot with default layer size of 10

A graph of a graph

Description automatically generated

Figure 4.22 Initial Neural Network Error Histogram plot with default layer size of 10

A screenshot of a graph

Description automatically generated

Figure 4.23 Initial Neural Network Regression plots with default layer size of 10

The initial neural network selected by nnstart has 10 hidden sigmoid layers. Doubling the number of layers from 10 to 20 produced the results shown in Tables 4.8 and 4.9 and Figures 4.25 to 4.28.

Table 4.8 Training results of initial neural network with a layer size of 20.

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 12 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.46 | 0.191 | 0 |
| Gradient | 2.65 | 0.0556 | 1e-07 |
| Mu | 0.001 | 1e-05 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |
|  |  |  |  |

Table 4.9 Training Results

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.19566 | 0.5804 |
| Validation | 750 | 0.1785 | 0.6187 |
| Test | 750 | 0.1927 | 0.5886 |

A graph of a graph

Description automatically generated with medium confidence

Figure 4.24 Initial Neural Network Performance plot with 20 layers

A graph of a bar graph

Description automatically generated

Figure 4.25 Initial Neural Network Error Histogram plot with 20 layers

A screenshot of a graph

Description automatically generated

Figure 4.26 Initial Neural Network Regression plot with 20 layers

A graph of a graph

Description automatically generated with medium confidence

Figure 4.27 Initial Neural Network Training State plot with 20 layers

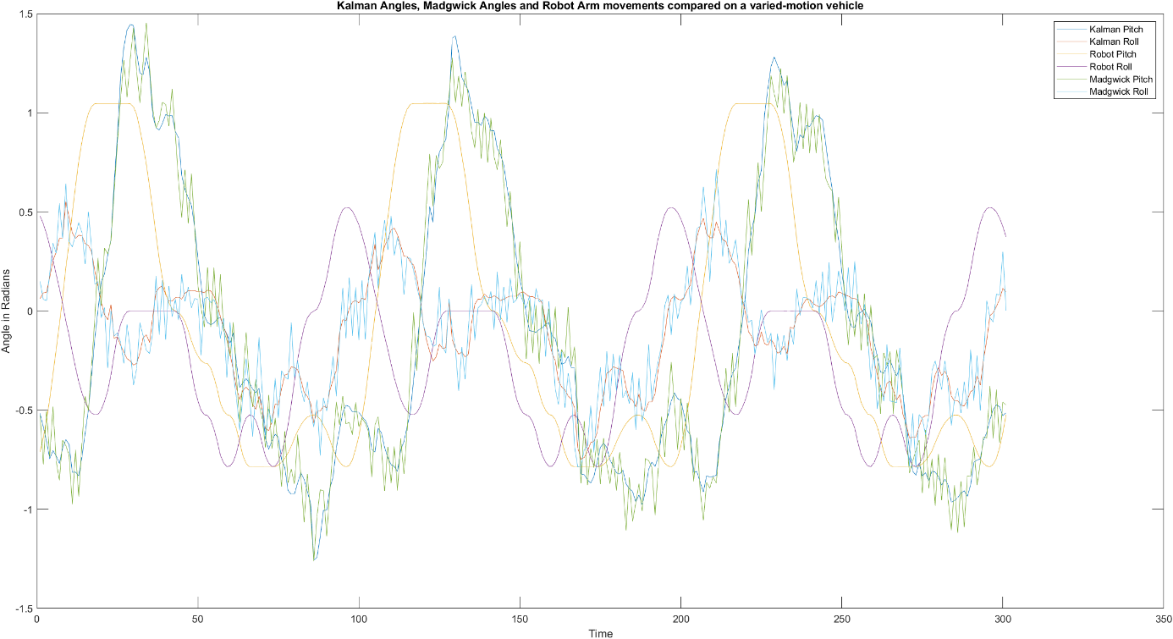
The MSE values of around 0.19 are fair results but the R values of around 0.6 are not desirable.  
Doubling the hidden layers did not appreciably improve the values indicating that the model itself and/or the data is likely to be the constraint. This was verified by increasing the number of layers to 100 – the error and R values did not appreciably move.  
The MSE value gives the average of the square value of "error” values where errors are the difference between the model’s predicted outputs and the actual outputs. MSE is a measure of overall accuracy of the model predictions. The Pearson coefficient, R, is a measure of variance between the expected results and the input data within a given model. Ideally, MSE should be 0 and R should be 1. With relatively low MSE values of around 0.19 and low R values (0.6), the model is providing good predictive accuracy on individual data points but is not understanding the relationships well.  
This could result from underfitting the data or the model might be missing other important data. The camera IMU dataset has only 5000 values, of which a few are zero values so underfitting is a distinct possibility. It is also likely that the approximations used to align the IMU data to the robot data creates issues when the datasets are longer, as is the case with this experiment.  
To test long-term alignment, a graph was produced of all filter output and related robot arm data.

Figure 4.28 Plot IMU and robot values from t=1000 to t=1300 from a dataset of 4999 values

From Figure 4.29 it is clear that the approximations used to align the IMU and robot datasets are not sufficiently accurate enough for IMU and robot data to remain in synchronisation across a large sample set.  
Looking at the datasets for both the imu and the robot arm, it was clear that the number of samples to record one second’s worth of data varied (sometimes dramatically) throughout the experiment on both systems.  
While it might have been possible to reduce some of these discrepancies by aligning the laptop and Raspberry Pi better, and reducing the number of running processes on both machines, the different processing power of these two machines would almost certain cause some differences between them. It was decided that leaving these discrepancies in the dataset would reflect more of a real world environment as the Pi is likely to be operating multiple processes at once to control the rover.

The manual synchronisation method was modified by testing proposed values across the entire sample set rather than just the beginning sequences and the experiment ran again with the new values of offsets. Outputs are shown below in Figures 4.30 to 4.34. These graphs show that a simple automated alignment approach to accurate offset values is not possible as there is some significant drift in sampling times between the Raspberry Pi and the robot arm as outlined above.

A graph showing a sound wave

Description automatically generated with medium confidence

Figure 4.29 Filter and Robot angles after new offset values –OK at this level

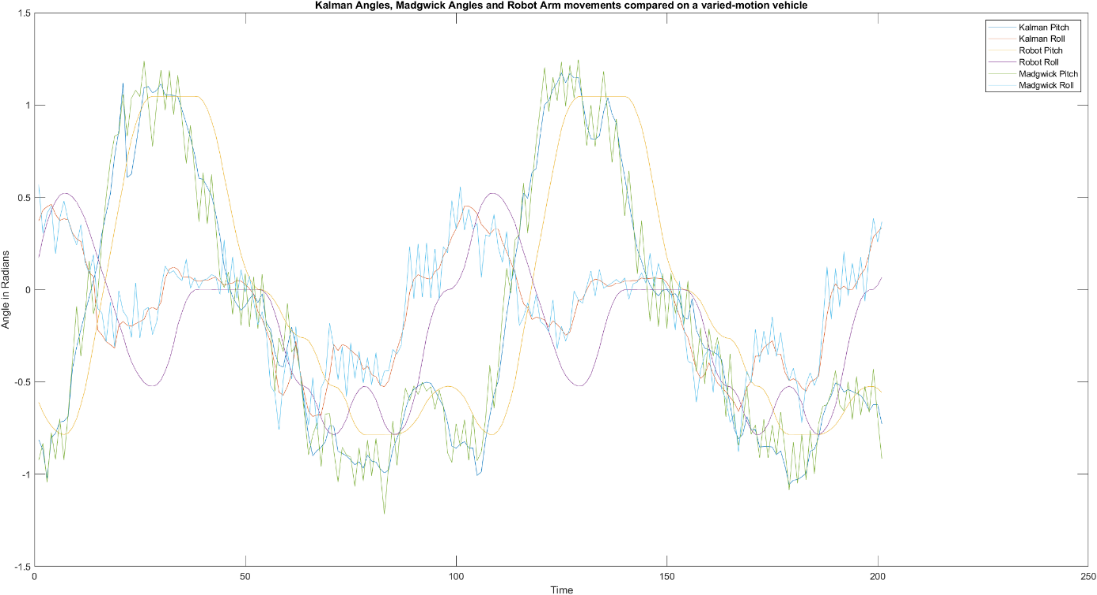


Figure 4.30 Filter and Arm results after new offset values - samples from 100-300.

A graph showing a variety of colored lines

Description automatically generated with medium confidence

Figure 4.31 Filter and Arm results after new offset values - samples from 1000-1300.

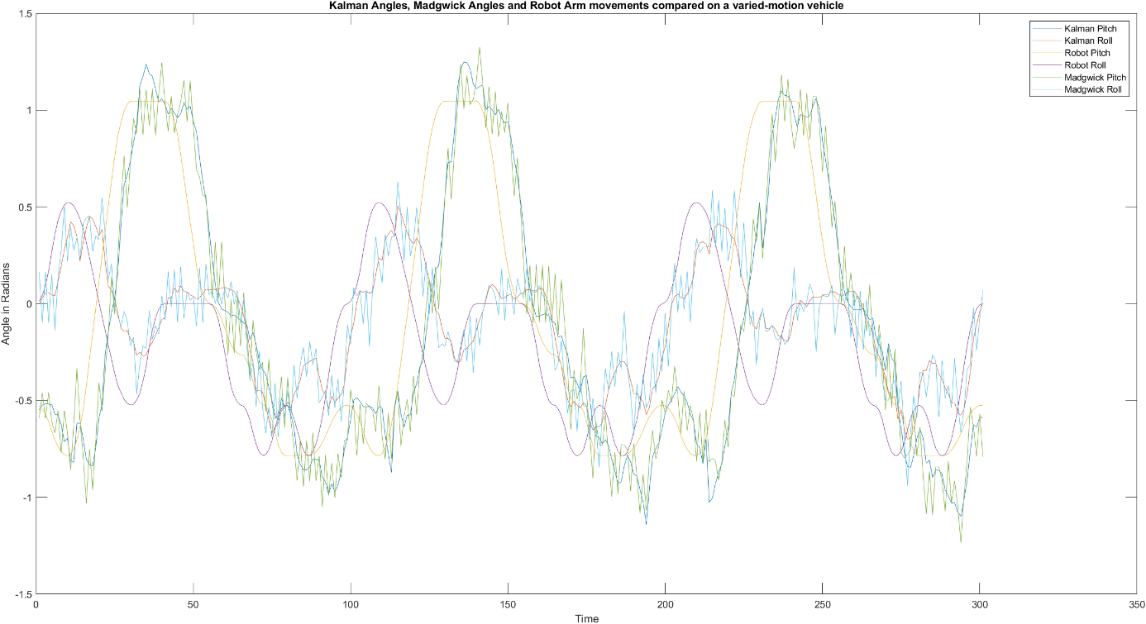


Figure 4.32 Filter and Arm results after new offset values - samples from 3000-3300.

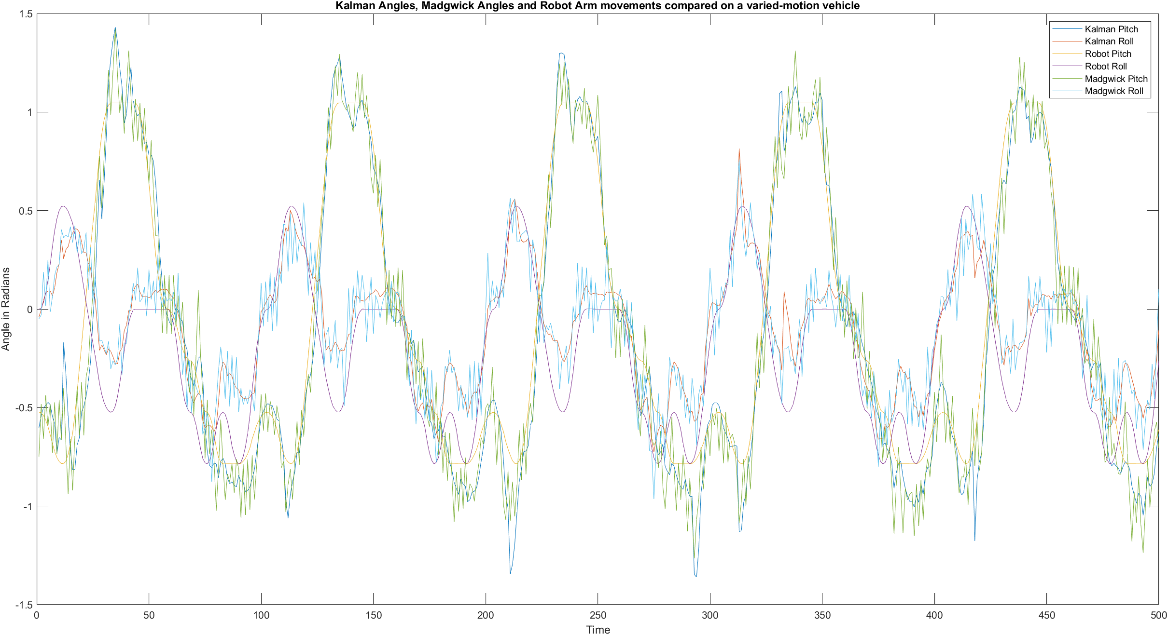


Figure 4.33 Filter and Arm results after new offset values - samples from 4500-4999.

As can be seen in Figures 4.30 to 4.34 the timing differences between the IMU data and the robot data vary considerably and not in a directly linear manner, however, over the entire dataset they appear to even out.  
  
Applying the same Neural network model as before on the “optimised” aligned data produced the results outlined in Tables 4.10 and 4.11 and Figures 4.35 to 4.38. This is a considerable improvement over the previous non-optimised results, and shows an acceptable performance level in both MSE and R values.

Table 4.10 Neural Network training progress after optimised alignment

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 19 | 1000 |
| Elapsed Time | - | 00:00:01 | - |
| Performance | 1.28 | 0.0438 | 0 |
| Gradient | 2.61 | 0.00683 | 1e-07 |
| Mu | 0.001 | 1e-05 | 1e+10 |
| Validation checks | 0 | 6 | 6 |

Table 4.11 Training results of Neural Network after optimised alignment.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0441 | 0.9219 |
| Validation | 750 | 0.0421 | 0.9250 |
| Test | 750 | 0.0423 | 0.9241 |

A graph on a computer screen

Description automatically generated

Figure 4.34 Training state plot after alignment optimisation

A graph with a line

Description automatically generated

Figure 4.35 Performance plot after alignment optimisation

A graph of a bar graph

Description automatically generated

Figure 4.36 Error Histogram plot after alignment optimisation

A group of graphs with dots

Description automatically generated with medium confidence

Figure 4.37 Regression plot after alignment optimisation  
  
The regression plots of Figure 4.37 show the fitting line in the middle of a pleasing random balance of data points on either side, indicating that the fit line is a good representation of the dataset. This good fit result occurs despite the presence of some outlying data values.

Dropping the number of layers down to 5 gave the training results shown in Tables 4.12 and 4.13. These are good results and only slightly less performant than the 10-layer model, showing that this problem is a relatively simple problem for a neural network to solve.

Table 4.12 Training progress on a 5 layer Neural network

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 20 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 0.608 | 0.0453 | 0 |
| Gradient | 1.24 | 0.00024 | 1e-07 |
| Mu | 0.001 | 1e-06 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |

Table 4.13 Training results on a 5-layer Neural Network

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0454 | 0.9188 |
| Validation | 750 | 0.0460 | 0.9183 |
| Test | 750 | 0.0414 | 0.9282 |

The R values did not significantly drop (from 0.91 at 2 layers to 0.80 at 1 layer) until the number of layers was lowered to 1 although the testing phase showed an increase in MSE errors (0.528 with 3 layers from 0.414 at 10 layers) when the number of layers was reduced to 3. This indicates that the data matching element of the problem is relatively simple for the model but that understanding the data relationships requires at least 3 layers. This finding matches the theorem of universal approximators proposed by Hornik (Hornik et al., 1989) that any multi-layer perceptron network of a depth of at least 1 for simple problems and a depth of 3 for more difficult problems can perform as a universal approximator.

### Filter and Neural network performance with three IMUs

Using the inputs of all three front IMUs, the 10-layer neural network above trained with the Levenberg-Marquardt method performed as outlined in Tables 4.14 and 4.15.  
  
Table 4.14 LM Training progress on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 14 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.2 | 0.0407 | 0 |
| Gradient | 2.25 | 0.0101 | 1e -07 |
| Mu | 0.001 | 1e-05 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |

Table 4.15 LM Training results on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0417 | 0.9259 |
| Validation | 750 | 0.0435 | 0.9227 |
| Test | 750 | 0.0447 | 0.9220 |

Reducing the layers to 5 produced the tabulated outcomes shown in Figures 4.16 and 4.17.

Table 4.16 LM Training progress on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 17 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 0.76 | 0.042 | 0 |
| Gradient | 1.22 | 0.00111 | 1e -07 |
| Mu | 0.001 | 1e-06 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |

Table 4.17 LM Training results on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0427 | 0.9239 |
| Validation | 750 | 0.0407 | 0.9277 |
| Test | 750 | 0.0442 | 0.9235 |

Training with the Bayesian regularisation training method on a 10-layer NN with 3 IMU inputs produced the results shown in Tables 4.18 and 4.19.

Table 4.18 Bayesian Training progress on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 243 | 1000 |
| Elapsed Time | - | 00:00:06 | - |
| Performance | 0.611 | 0.0387 | 0 |
| Gradient | 1.75 | 0.000168 | 1e -07 |
| Mu | 0.005 | 5e+10 | 1e+10 |
| Effective # Param | 212 | 185 | 0 |
| Sum Squared Param | 42.7 | 105 | 0 |

Table 4.19 Bayesian Training results on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 4249 | 0.0387 | 0.9311 |
| Validation | 0 | - | - |
| Test | 750 | 0.0472 | 0.9191 |

The Bayesian training method used on a 5-layer network with 3 IMU inputs produced the results demonstrated in Tables 4.20 and 4.21.

Table 4.20 Bayesian Training progress on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 131 | 1000 |
| Elapsed Time | - | 00:00:01 | - |
| Performance | 0.613 | 0.0423 | 0 |
| Gradient | 1.02 | 0.000163 | 1e -07 |
| Mu | 0.005 | 5e+10 | 1e+10 |
| Effective # Param | 107 | 90.8 | 0 |
| Sum Squared Param | 19.6 | 31.2 | 0 |

Table 4.21 Bayesian Training results on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 4249 | 0.0423 | 0.9251 |
| Validation | 0 | - | - |
| Test | 750 | 0.0422 | 0.9248 |

Results from training with the scaled conjugate training method on a 10-layer NN with 3 IMU inputs are shown in Tables 4.23 and 4.24.

Table 4.22 Scaled Conjugate Training progress on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 135 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.1 | 0.0419 | 0 |
| Gradient | 2.7 | 0.0105 | 1e-06 |
| Validation Checks | 0 | 6 | 6 |

Table 4.23 Scaled Conjugate Training results on 3 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0420 | 0.9239 |
| Validation | 750 | 0.0403 | 0.9332 |
| Test | 750 | 0.0505 | 0.9126 |

Results from training with the scaled conjugate training method on a 5-layer NN with 3 IMU inputs are shown in Tables 4.24 and 4.25.

Table 4.24 Scaled Conjugate Training progress on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 58 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.43 | 0.0446 | 0 |
| Gradient | 2.72 | 0.00934 | 1e-06 |
| Validation Checks | 0 | 6 | 6 |

Table 4.25 Scaled Conjugate Training results on 3 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0449 | 0.9217 |
| Validation | 750 | 0.0466 | 0.9138 |
| Test | 750 | 0.0437 | 0.9193 |

### Neural network of a varied-motion vehicle: All IMUs

Results from training with the Levenberg-Marquardt training method on a 10-layer NN with 5 IMU inputs are shown in Tables 4.26 and 4.27. Matlab used 13.6% of CPU time and 242Mb of RAM during the process.

Table 4.26 LM Training progress on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 11 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 2.4 | 0.0386 | 0 |
| Gradient | 4.75 | 0.00179 | 1e-07 |
| Mu | 0.001 | 0.0001 | 1e+10 |
| Validation Checks | 0 | 6 | 6 |

Table 4.27 LM Training results on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0413 | 0.9265 |
| Validation | 750 | 0.0453 | 0.9193 |
| Test | 750 | 0.0432 | 0.9246 |

Results from training with the Bayesian regularisation training method on a 10-layer NN with 5 IMU inputs are shown in Tables 4.28 and 4.29. Matlab used 34.3% of CPU time and 237Mb of RAM during the process.

Table 4.28 Bayesian Training progress on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 958 | 1000 |
| Elapsed Time | - | 00:00:42 | - |
| Performance | 1.55 | 0.0373 | 0 |
| Gradient | 3.23 | 0.000228 | 1e-07 |
| Mu | 0.005 | 5e+10 | 1e+10 |
| Effective # Param | 332 | 283 | 0 |
| Sum Squared Param | 42.2 | 127 | 0 |

Table 4.29 Bayesian Training results on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 4249 | 0.0372 | 0.9344 |
| Validation | 0 | - | - |
| Test | 750 | 0.0448 | 0.9204 |

Results from training with the Scaled Conjugate training method on a 10-layer NN with 5 IMU inputs are shown in Tables 4.30 and 4.31. Matlab used 6.3% of CPU time and 243Mb of RAM during the process.

Table 4.30 Scaled Conjugate Training progress on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 55 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.24 | 0.0436 | 0 |
| Gradient | 3.53 | 0.00603 | 1e-06 |
| Validation Checks | 0 | 6 | 6 |

Table 4.31 Scaled Conjugate Training results on 5 IMUs with 10-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0440 | 0.9216 |
| Validation | 750 | 0.0470 | 0.9197 |
| Test | 750 | 0.0405 | 0.9258 |

Results from training with the Scaled Conjugate training method on a 10-layer NN with 5 IMU inputs are shown in Tables 4.32 and 4.33 and Figures Y. Matlab used 5.0% of CPU time and 244Mb of RAM during the process.

Table 4.32 Scaled Conjugate Training progress on 5 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Initial Value | Stopped Value | Target Value |
| Epoch | 0 | 55 | 1000 |
| Elapsed Time | - | 00:00:00 | - |
| Performance | 1.24 | 0.0436 | 0 |
| Gradient | 3.53 | 0.00603 | 1e-06 |
| Validation Checks | 0 | 6 | 6 |

Table 4.33 Scaled Conjugate Training results on 5 IMUs with 5-layer NN

|  |  |  |  |
| --- | --- | --- | --- |
|  | Observations | MSE | R |
| Training | 3499 | 0.0440 | 0.9216 |
| Validation | 750 | 0.0470 | 0.9197 |
| Test | 750 | 0.0405 | 0.9258 |

### Evaluating differing orientations of IMUs

In Chapter 5, analysis of the results will be undertaken with conclusions reached in Chapter 6.

# Discussion and Analysis

## Experimental Design Analysis

First, the experimental design and implementation needs to be discussed.  
  
The overall design concept is sound but implementation can be significantly improved by removing the requirement for an I2C multiplexor, and ideally, utilising individual SPI connections to increase the data collection rate, instead of the original I2C implementation**.** (The selected IMUs do not support a Universal Asynchronous Receiver-Transmitter (UART) interface).  
  
Ideally, the Raspberry Pi should be replaced with a system capable of utilising TensorFlow models so that the developed model in Matlab can be directly implemented on the unit and/or to increase the data collection rate.  
  
The potential for TCP/IP collisions when collecting data (a minor factor in data alignment but easily mitigated) can be addressed by direct connections from individual network cards, or better, two time-synchronised computers directly connected to the vehicle-processor and the robot arm. Using reliable and consistent time-stamped data would decrease the time necessary to manually align readings and it is possible that this can be automated. Reducing the number of processes running on the Raspberry Pi may help in training but will not aid accuracy under operation.  
  
The aluminium and plastic baseboard design was proposed for rigidity to ensure the IMU sensors all operated on the same plane to make comparisons simpler but it is likely that IMU sensors will be directly mounted to the panels on a vehicle and so it is possible that magnetometer data can be used to aid in orientation determination but at possible extra computation costs of dealing with sensors not on the same plane as the camera-based IMU. If a single baseboard is to be used to eliminate multiple plane concerns, then this should be changed to a non-magnetic rigid material such as abs plastic.  
  
The robot arm provides predictable angles once calibrated however the robot joint construction does not easily permit roll and pitch modifications while travelling in forward except for small distances (limited by the reach of the arm and its joints). As such, this experiment does not consider roll and pitch movements while under any form of lateral motion. A track should be designed with accurate angles so that the effects of lateral motion while the vehicle is undergoing pitch and roll movements can be analysed. The time constraints of this experiment did not permit the construction of such a track that is long enough to give a good set of data useful for training purposes. The track will need to contain a sled that the baseboard can travel on because the vehicle contains oil-filled suspension and suspension springs so determining the actual rotation of the vehicle will be challenging.

## Filter results analysis.

The results section shows that using a Madgwick complimentary filter with gradient-descent optimisation to fuse gyroscope and accelerometer data to predict roll, pitch and yaw values is somewhat accurate when the timing of the datasets are predictable but, over time, this approach would need to be “reset” every so often to prevent the accumulation of errors, especially when the vehicle will change its speed when responding to local conditions. While this issue can be partially explained by the inconsistencies of the alignment of the imu and robot arm data, in operation, these timing inconsistencies are likely to be present, due to processor scheduling. This aspect of the Madgwick filter will need more research if this filter is implemented.  
   
The Kalman filter approximates the robot arm control movement more accurately than the Madgwick and does not appear reactive to the length of the data samples fed into it but has a tendency to overshoot on changes of angle and takes at least 7 times the processing power (this will vary depending on the processor implemented). Reducing the overshooting behaviour will almost certainly alter responsiveness to change (this is a classic filter dynamic) and ideally would be tuned to the vehicle’s speed to optimise the value. This should be achievable in a dynamic fashion.

Of the two filters, the Kalman is preferable for a long-range rover to avoid the sampling size conditions of the Madgwick filter, despite the additional processing overhead.

## Neural Network results analysis.

A default MLP 10-layer sigmoid neural network appears easily capable of determining euler angles from gyroscope and accelerometer data from a single imu without any optimisation.  
  
In Chapter 6, conclusions and future recommendations will be discussed.

# . Conclusion and Future Works

The research objectives have been answered through this experiment.  
  
Research Objective One was to determine whether multiple inertial measurement units in conjunction with a neural network can improve image stabilisation of a camera on an RC vehicle, compared with a single inertial measurement unit.  
  
Research Objective Two was to determine the least number of inertial measurement units required to provide a significant measurable improvement.

# 

# Appendices

## Appendix 1. Code

All code can be obtained from https://github.com/BratNZ/Thesis

## Appendix 2. Calibration results.

Table 7.1 IMU Error Coefficients produced from initial calibration (rounded to 8 decimal places)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| IMU | GyroX | GyroY | GyroZ | AccelX | AccelY | AccelZ |
| RR | -0.02998955 | 0.01596670 | 0.00715132 | 0.00078995 | -0.01058559 | -0.02925765 |
| FR | -0.03219639 | 0.00790274 | -7.2477796985e-05 | -0.00159294 | 0.01550039 | -0.01817877 |
| CM | 0.023254980 | 0.00284742 | -0.00660733 | 0.01240663 | 0.0155684 | -0.03045218 |
| FL | 0.008131502 | -0.00708547 | 0.00312161 | -0.00665329 | -0.01244544 | -0.01997862 |
| RL | -0.00307724 | 0.00518083 | -0.00114486 | 0.0027521 | 0.0128008 | 0.00237755 |

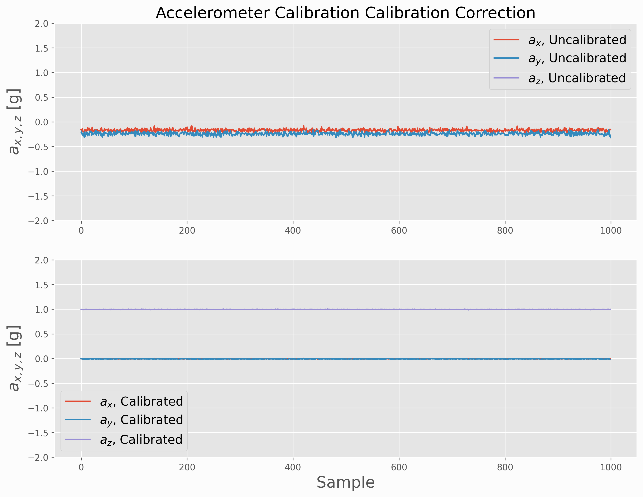
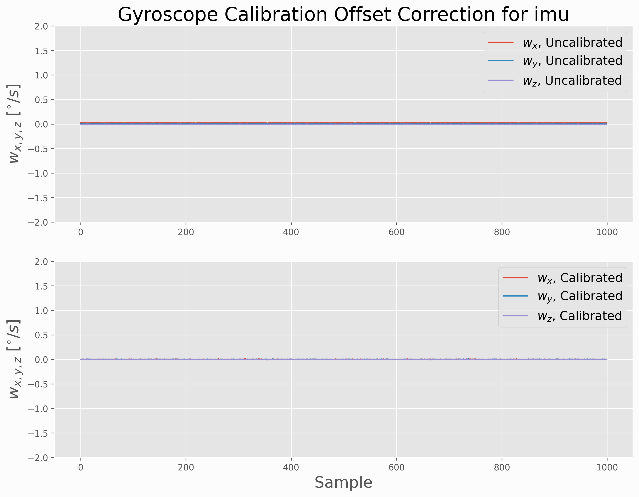
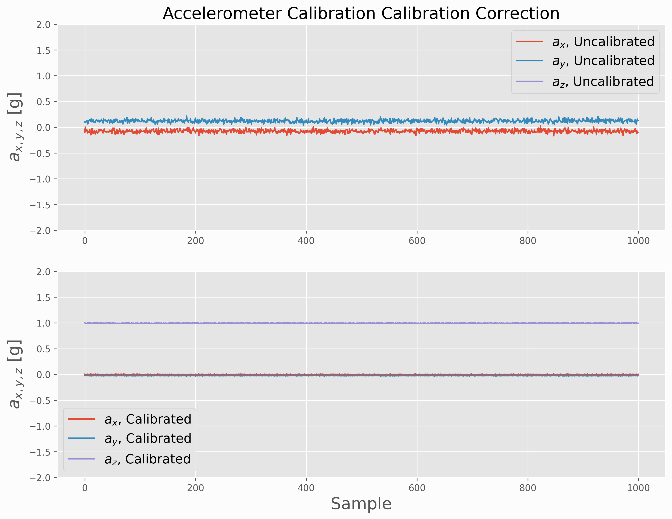
  


Figure 7.1 IMU Calibration graphs for Central Camera IMU

Figure 7.2 IMU Calibration graphs for Front Left IMU

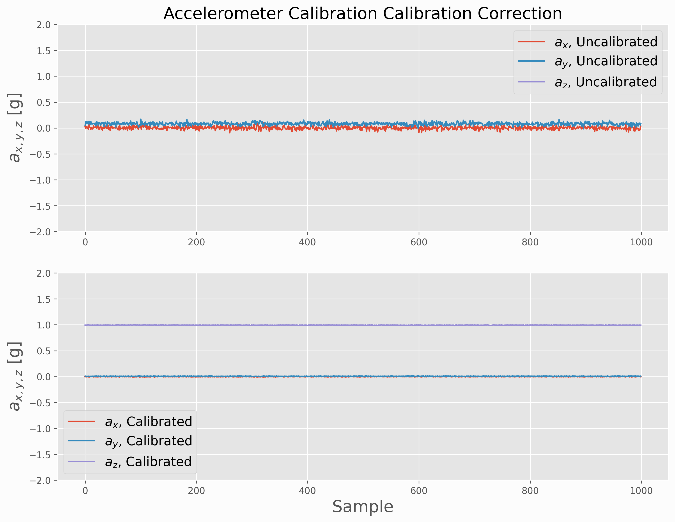
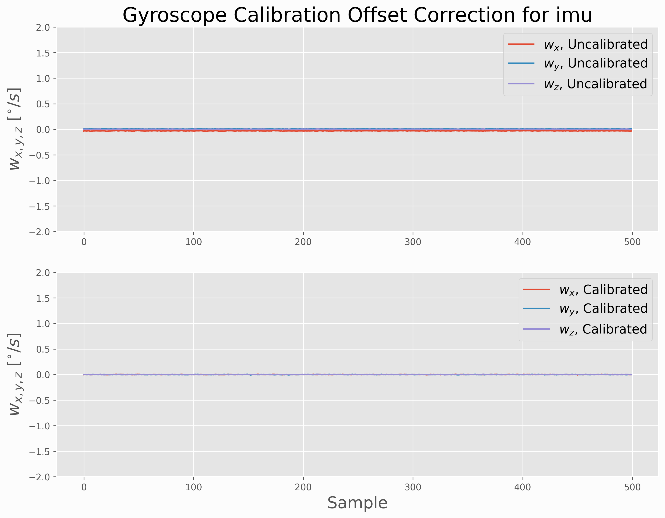


Figure 7.3 IMU Calibration graphs for Front Right IMU

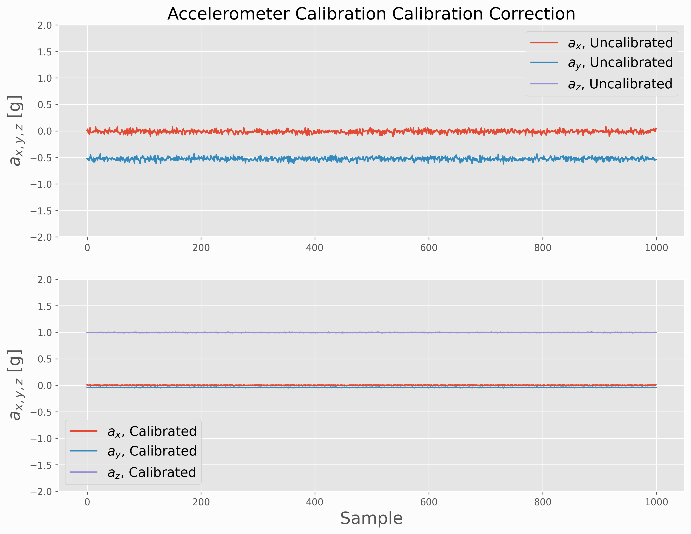
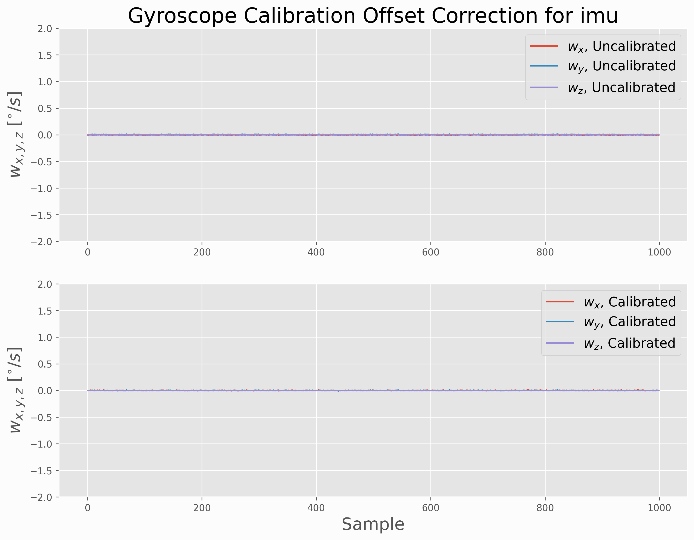


Figure 7.4 IMU Calibration graphs for Rear Left IMU

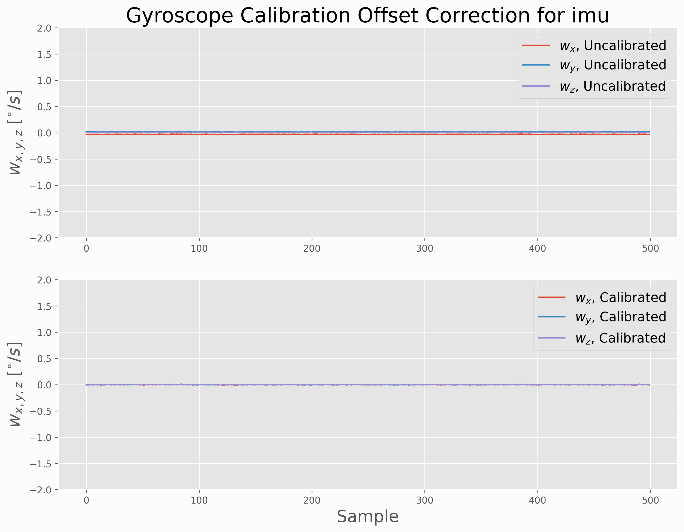
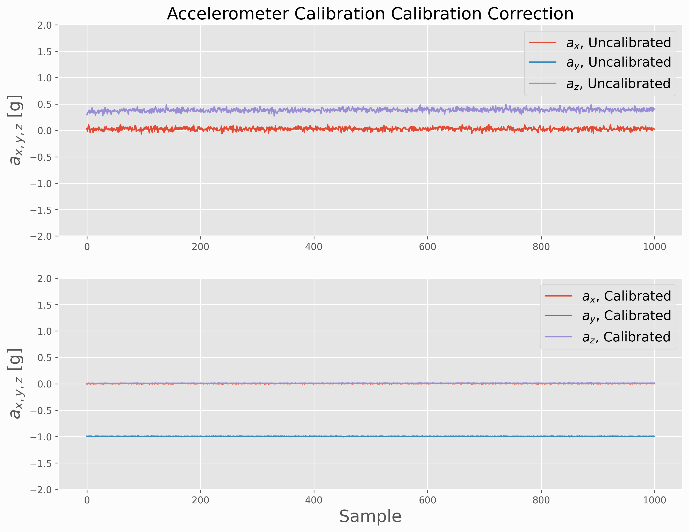


Figure 7.5 Figure IMU Calibration graphs for Right Rear IMU

## Appendix 3. Electrical Characteristics of IMU-29048 IMU

|  |  |
| --- | --- |
| Digital Output | I2C  SPI |
| VDD Power Supply / V | 1.71 to 3.6 |
| VDDIO I/O Power Supply / V | 1.71 to 3.6 |
| Gyro FSR | ±250/500/1000/2000 |
| Gyro Sensitivity Error | ±1.5% |
| Gyro Rate Noise | 0.015dps/√Hz |
| Accel FSR | ±2/4/8/16 |
| Accel Sensitivity Error | ±0.5% |
| Accel Noise | 230μg/√Hz |
| Compass FSR | ±4900μT |
| Pressure Sensor Relative Accuracy |  |
| Pressure Sensor Noise |  |

# Chapter 8. References

Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. (2015). A comprehensive review of swarm optimization algorithms. *PLoS ONE*, *10*(5). https://doi.org/10.1371/JOURNAL.PONE.0122827

Adafruit. (2022). *Slamtec RPLIDAR A1 - 360 Laser Range Scanner : ID 4010 : $99.95 : Adafruit Industries, Unique & fun DIY electronics and kits*. https://www.adafruit.com/product/4010

Adelantado, F., Vilajosana, X., Tuset-Peiro, P., Martinez, B., Melia-Segui, J., & Watteyne, T. (2017). Understanding the Limits of LoRaWAN. *IEEE Communications Magazine*, *55*(9), 34–40. https://doi.org/10.1109/MCOM.2017.1600613

Ahmed, S., Qiu, B., Ahmad, F., Kong, C. W., & Xin, H. (2021). A State-of-the-Art Analysis of Obstacle Avoidance Methods from the Perspective of an Agricultural Sprayer UAV’s Operation Scenario. *Agronomy 2021, Vol. 11, Page 1069*, *11*(6), 1069. https://doi.org/10.3390/AGRONOMY11061069

Akagawa, K. (1996). *US5994699A - Thermal camera for infrared imaging - Google Patents*. https://patents.google.com/patent/US5994699A/en

Al-Sarawi, S., Anbar, M., Alieyan, K., & Alzubaidi, M. (2017). Internet of Things (IoT) communication protocols: Review. *ICIT 2017 - 8th International Conference on Information Technology, Proceedings*, 685–690. https://doi.org/10.1109/ICITECH.2017.8079928

Andreev, S., Galinina, O., Pyattaev, A., Gerasimenko, M., Tirronen, T., Torsner, J., Sachs, J., Dohler, M., & Koucheryavy, Y. (2015). Understanding the IoT connectivity landscape: A contemporary M2M radio technology roadmap. *IEEE Communications Magazine*, *53*(9), 32–40. https://doi.org/10.1109/MCOM.2015.7263370

Android Authority. (n.d.-a). *Image-Stablization-GIF.gif (404×296)*. Retrieved August 17, 2023, from https://www.androidauthority.com/wp-content/uploads/2020/02/Image-Stablization-GIF.gif

Android Authority. (n.d.-b). *Image-Stablization-Perspective.gif (350×261)*. Retrieved August 17, 2023, from https://www.androidauthority.com/wp-content/uploads/2020/02/Image-Stablization-Perspective.gif

Arsalan, M., Umair, A., & Kumar Verma, V. (2018). Dash7: Performance. In *IOSR Journal of Electronics and Communication Engineering (IOSRJECE)* (Vol. 2, Issue 5). www.iosrjournals.orgwww.iosrjournals.org

Augustin, A., Yi, J., Clausen, T., & Townsley, W. M. (2016). A study of Lora: Long range & low power networks for the internet of things. *Sensors (Switzerland)*, *16*(9). https://doi.org/10.3390/s16091466

Auysakul, J., Xu, H., & Pooneeth, V. (2018). A Hybrid Motion Estimation for Video Stabilization Based on an IMU Sensor. *Sensors 2018, Vol. 18, Page 2708*, *18*(8), 2708. https://doi.org/10.3390/S18082708

Auysakul, J., Xu, H., & Pooneeth, V. (2019). Video stabilization with a dual system based on an IMU sensor for the mobile robot. *Advances in Intelligent Systems and Computing*, *856*, 618–626. https://doi.org/10.1007/978-3-030-00214-5\_78/COVER

Aviation Authority of New Zealand, C. (2015). *Published by the Civil Aviation Authority of New Zealand Unmanned Aircraft Operator Certification*. www.caa.govt.nz

Baronti, P Pillai, P Chook, V.W.C Chessa, S Gotta, A Hu, F. (2007). Wireless sensor networks: a survey on the state of the art and the 802.15.4 and ZigBee standards. *Computer Communications*, *30*(7), 1655–1695.

Battiato, S., Gallo, G., Puglisi, G., & Scellato, S. (2007). SIFT features tracking for video stabilization. *Proceedings - 14th International Conference on Image Analysis and Processing, ICIAP 2007*, 825–830. https://doi.org/10.1109/ICIAP.2007.4362878

Beef & Lamb NZ. (2021). *45Th Edition Farm Facts*. www.beeflambnz.com

Beef and Lamb NZ. (2017). *Guide to New Zealand Cattle Farming*.

Beliveau, A., Spencer, G. T., Thomas, K. A., & Roberson, S. L. (1999). Evaluation of MEMS capacitive accelerometers. *IEEE Design & Test of Computers*, *16*(4), 48–56. https://doi.org/10.1109/54.808209

Bernini, N., Bertozzi, M., Castangia, L., Patander, M., & Sabbatelli, M. (2014). Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey. *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, 873–878. https://doi.org/10.1109/ITSC.2014.6957799

BHPhotoVideo.com. (n.d.). *13-fisheye\_dsc2260.jpg (960×640)*. Retrieved August 17, 2023, from https://static.bhphotovideo.com/explora/sites/default/files/styles/960/public/13-fisheye\_dsc2260.jpg?itok=JkCNVbW6

Blanche, J., Chirayil Nandakumar, S., Mitchell, D., Harper, S., Groves, K., West, A., Lennox, B., Watson, S., Flynn, D., & Yamamoto, I. (n.d.). *Millimeter-Wave Sensing for Avoidance of High-Risk Ground Conditions for Mobile Robots*.

Borges De Sousa, J., & Andrade Gonçalves, • G. (2010). *Unmanned vehicles for environmental data collection*. https://doi.org/10.1007/s10098-010-0313-5

Byeon, M., & Yoon, S. W. (2020). Analysis of Automotive Lidar Sensor Model Considering Scattering Effects in Regional Rain Environments. *IEEE Access*, *8*, 102669–102679. https://doi.org/10.1109/ACCESS.2020.2996366

Castano, R., Manduchi, R., & Fox, J. (2014). *Classification experiments on real-world texture*. Jpl.Nasa.Gov. https://trs.jpl.nasa.gov/handle/2014/41475

Chang, S., Zhong, Y., Quan, Z., Hong, Y., Zeng, J., & Du, D. (2016). A real-time object tracking and image stabilization system for photographing in vibration environment using OpenTLD algorithm. *Proceedings of IEEE Workshop on Advanced Robotics and Its Social Impacts, ARSO*, *2016*-*Novem*, 141–145. https://doi.org/10.1109/ARSO.2016.7736271

Chao, H., Gu, Y., Gross, J., Guo, G., Fravolini, M. L., & Napolitano, M. R. (2013). A comparative study of optical flow and traditional sensors in UAV navigation. *Proceedings of the American Control Conference*, 3858–3863. https://doi.org/10.1109/ACC.2013.6580428

Chaudhary, S., Zakhami, S., & Roy, S. D. (2019). Visual Feedback based Trajectory Planning to Pick an Object and Manipulation using Deep learning. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3352593.3352616

Chemhengcharoen, P., Nilsumrit, P., Pongpetrarat, P., & Phanomchoeng, G. (2019). *Development of a Prototype of Autonomous Vehicle for Agriculture Applications*. https://doi.org/10.1145/3330180.3330191

Cheong, M. K., Bahiki, M. R., & Azrad, S. (2016). Development of collision avoidance system for useful UAV applications using image sensors with laser transmitter. *IOP Conference Series: Materials Science and Engineering*, *152*(1), 012026. https://doi.org/10.1088/1757-899X/152/1/012026

Cheong, P. S., Bergs, J., Hawinkel, C., & Famaey, J. (2017). Comparison of LoRaWAN classes and their power consumption. *2017 IEEE Symposium on Communications and Vehicular Technology, SCVT 2017*, *2017*-*December*, 1–6. https://doi.org/10.1109/SCVT.2017.8240313

Cho, Y., Han, J., Kim, J., Lee, P., & Park, S. B. (2019). Experimental validation of a velocity obstacle based collision avoidance algorithm for unmanned surface vehicles. *IFAC-PapersOnLine*, *52*(21), 329–334. https://doi.org/10.1016/j.ifacol.2019.12.328

Choit, J., Ahnt, S., & Chungt, W. K. (2005). Robust sonar feature detection for the slam of mobile robot. *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 2083–2088. https://doi.org/10.1109/IROS.2005.1545284

Crane, C. D., Armstrong, D. G., Touchton, R., Galluzzo, T., Solanki, S., Lee, J., Kent, D., Ahmed, M., Montane, R., Ridgeway, S., Velat, S., Garcia, G., Griffis, M., Gray, S., Washburn, J., & Routson, G. (2006). Field report: Team CIMAR’s NaviGATOR: An unmanned ground vehicle for the 2005 DARPA Grand Challenge. *Journal of Field Robotics*, *23*(8), 599–623. https://doi.org/10.1002/ROB.20127

Dai, R., Stein, R. B., Andrews, B. J., James, K. B., & Wieler, M. (1996). Application of tilt sensors in functional electrical stimulation. *IEEE Transactions on Rehabilitation Engineering*, *4*(2), 63–72. https://doi.org/10.1109/86.506403

Damlamian, A., & Jaffard, S. (2019). Wavelet methods in mathematical analysis and engineering. *Wavelet Methods in Mathematical Analysis and Engineering*, 1–178. https://doi.org/10.1142/7899

*DASH7 Alliance – An open specification*. (n.d.). Retrieved January 14, 2021, from https://dash7-alliance.org/

De Sanctis, M., Cianca, E., Araniti, G., Bisio, I., & Prasad, R. (2016). Satellite communications supporting internet of remote things. *IEEE Internet of Things Journal*, *3*(1), 113–123. https://doi.org/10.1109/JIOT.2015.2487046

De Simone, M. C., Rivera, Z. B., & Guida, D. (2018a). Obstacle Avoidance System for Unmanned Ground Vehicles by Using Ultrasonic Sensors. *Machines 2018, Vol. 6, Page 18*, *6*(2), 18. https://doi.org/10.3390/MACHINES6020018

De Simone, M. C., Rivera, Z. B., & Guida, D. (2018b). Obstacle Avoidance System for Unmanned Ground Vehicles by Using Ultrasonic Sensors. *Machines 2018, Vol. 6, Page 18*, *6*(2), 18. https://doi.org/10.3390/MACHINES6020018

Derek W. Bailey, Mark G. Trotter, Colt W. Knight, M. G. T. (2018). Use of GPS tracking collars and accelerometers for rangeland livestock production research. *American Society of Animal Science*, 81–88. https://doi.org/10.1093/tas/txx006

Deruyck, M., Vereecken, W., Tanghe, E., Joseph, W., Pickavet, M., Martens, L., & Demeester, P. (2010). Comparison of power consumption of mobile WiMAX, HSPA and LTE access networks. *2010 9th Conference of Telecommunication, Media and Internet, CTTE 2010*. https://doi.org/10.1109/CTTE.2010.5557715

Druzhkov, P. N., & Kustikova, V. D. (2016). A survey of deep learning methods and software tools for image classification and object detection. *Pattern Recognition and Image Analysis 2016 26:1*, *26*(1), 9–15. https://doi.org/10.1134/S1054661816010065

Duroc, Y. (2022). From Identification to Sensing: RFID Is One of the Key Technologies in the IoT Field. *Sensors 2022, Vol. 22, Page 7523*, *22*(19), 7523. https://doi.org/10.3390/S22197523

Dwork, C., McSherry, F., Nissim, K., & Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *3876 LNCS*, 265–284. https://doi.org/10.1007/11681878\_14

Euston, M., Coote, P., Mahony, R., Kim, J., & Hamel, T. (2008). A complementary filter for attitude estimation of a fixed-wing UAV. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS*, 340–345. https://doi.org/10.1109/IROS.2008.4650766

Fan, B., Li, Q., & Liu, T. (2017). How Magnetic Disturbance Influences the Attitude and Heading in Magnetic and Inertial Sensor-Based Orientation Estimation. *Sensors 2018, Vol. 18, Page 76*, *18*(1), 76. https://doi.org/10.3390/S18010076

Fang, H., Gutowski, M., Li, N., & Disogra, M. (2013). *Performance Evaluation of NCDOT W-beam Guardrails under MASH TL-2 Conditions*. 5.

*Fixed link licence | Radio Spectrum Management New Zealand*. (n.d.). Retrieved December 7, 2020, from https://www.rsm.govt.nz/licensing/licences-you-must-pay-for/fixed-location-licences/fixed-link-licence/

Foix, S., Alenyà, G., & Torras, C. (2011). Lock-in time-of-flight (ToF) cameras: A survey. *IEEE Sensors Journal*, *11*(9), 1917–1926. https://doi.org/10.1109/JSEN.2010.2101060

Formsma, O., Dijkshoorn, N., Van Noort, S., & Visser, A. (2010). *Realistic Simulation of Laser Range Finder Behavior in a Smoky Environment*. http://www.science.uva.nl/research/isla

Fossum, E. R., & Hondongwa, D. B. (2014). A review of the pinned photodiode for CCD and CMOS image sensors. *IEEE Journal of the Electron Devices Society*, *2*(3), 33–43. https://doi.org/10.1109/JEDS.2014.2306412

Gaddam, S. C., & Rai, M. K. (2018). A comparative study on various LPWAN and cellular communication technologies for IoT based smart applications. *2018 International Conference on Emerging Trends and Innovations In Engineering And Technological Research, ICETIETR 2018*. https://doi.org/10.1109/ICETIETR.2018.8529060

Ganesan, P., & Sajiv, G. (2018). A comprehensive study of edge detection for image processing applications. *Proceedings of 2017 International Conference on Innovations in Information, Embedded and Communication Systems, ICIIECS 2017*, *2018*-*Janua*, 53–54. https://doi.org/10.1109/ICIIECS.2017.8275968

Gao, H., Liu, J., Wei, Q., & Yu, Y. (2016). Mobile Robot Obstacle Avoidance Algorithms Based on Information Fusion of Vision and Sonar Space robot View project Reconfigurable Modular Robot View project Mobile Robot Obstacle Avoidance Algorithms Based on Information Fusion of Vision and Sonar. *Article in International Journal of Future Generation Communication and Networking*, *9*(8), 111–120. https://doi.org/10.14257/ijfgcn.2016.9.8.11

Ghaffari, M., Alhaj Ali, S. M., Murthy, V., Liao, X., Gaylor, J., & Hall, E. L. (2004). Design of an unmanned ground vehicle, bearcat III, theory and practice. *Journal of Robotic Systems*, *21*(9), 471–480. https://doi.org/10.1002/ROB.20027

Gheorghiu, R., & Iordache, V. (2018). Use of Energy Efficient Sensor Networks to Enhance Dynamic Data Gathering Systems: A Comparative Study between Bluetooth and ZigBee. *Sensors*, *18*(6), 1801. https://doi.org/10.3390/s18061801

Gilmore, M. S., Castaño, R., Mann, T., Anderson, R. C., Mjolsness, E. D., Manduchi, R., & Saunders, R. S. (n.d.). *Strategies for autonomous rovers at Mars*.

Gomez, C., Oller, J., & Paradells, J. (2012). Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. *Sensors (Switzerland)*, *12*(9), 11734–11753. https://doi.org/10.3390/s120911734

Goodin, C., Carruth, D., Doude, M., & Hudson, C. (2019). Predicting the influence of rain on LIDAR in ADAS. *Electronics (Switzerland)*, *8*(1). https://doi.org/10.3390/ELECTRONICS8010089

Hall, C. (2021). *Future batteries, coming soon: Charge in seconds, last months a*. https://www.pocket-lint.com/gadgets/news/130380-future-batteries-coming-soon-charge-in-seconds-last-months-and-power-over-the-air

Haq, I., Anwar, S., Shah, K., Khan, M. T., & Shah, S. A. (2015). Fuzzy Logic Based Edge Detection in Smooth and Noisy Clinical Images. *PLOS ONE*, *10*(9), e0138712. https://doi.org/10.1371/JOURNAL.PONE.0138712

Hautì, N., Tarel, J.-P., Lavenant, J., Aubert, D., Hautì Ere (b, N., Tarel, J.-P., Lavenant, · J, & Aubert, · D. (2006). Automatic fog detection and estimation of visibility distance through use of an onboard camera. *Machine Vision and Applications 2005 17:1*, *17*(1), 8–20. https://doi.org/10.1007/S00138-005-0011-1

Haxhibeqiri, J., De Poorter, E., Moerman, I., & Hoebeke, J. (2018a). A survey of LoRaWAN for IoT: From technology to application. In *Sensors (Switzerland)* (Vol. 18, Issue 11). MDPI AG. https://doi.org/10.3390/s18113995

Haxhibeqiri, J., De Poorter, E., Moerman, I., & Hoebeke, J. (2018b). A Survey of LoRaWAN for IoT: From Technology to Application. *Sensors 2018, Vol. 18, Page 3995*, *18*(11), 3995. https://doi.org/10.3390/S18113995

Higgins, W. T. (1975). A Comparison of Complementary and Kalman Filtering. *IEEE Transactions on Aerospace and Electronic Systems*, *AES*-*11*(3), 321–325. https://doi.org/10.1109/TAES.1975.308081

Hoffman, B. D., Baumgartner, E. T., Huntsberger, T. L., & Schenker, P. S. (1999). Improved Rover State Estimation in Challenging Terrain. *Autonomous Robots 1999 6:2*, *6*(2), 113–130. https://doi.org/10.1023/A:1008879310128

Hoffmann, J., Jüngel, M., & Lötzsch, M. (2004). *LNAI 3276 - A Vision Based System for Goal-Directed Obstacle Avoidance*. http://www.aiboteamhumboldt.com

Hornik, K., Stinchcombe, M., & White, H. (1989). Sci-Hub | Multilayer feedforward networks are universal approximators. Neural Networks, 2(5), 359–366 | 10.1016/0893-6080(89)90020-8. *Nerual Networks*, *2*, 359–366. https://sci-hub.se/10.1016/0893-6080(89)90020-8

Hsu, S. C., Liang, S. F., & Lin, C. T. (2005). A robust digital image stabilization technique based on inverse triangle method and background detection. *IEEE Transactions on Consumer Electronics*, *51*(2), 335–345. https://doi.org/10.1109/TCE.2005.1467968

Huang, J., Jiang, S., & Lu, X. (2001). Rain backscattering properties and effects on the radar performance at MM wave band. *International Journal of Infrared and Millimeter Waves*, *22*(6), 917–922. https://doi.org/10.1023/A:1014922632540

Huang, S., Qin, H., Zhou -, Z., Liu, Z., Liu, X., Duan, G., -, A., Li, X., Liu, W., Pan, Y., Wang, Q., Meng, Z., & Liu, H. (2020). Review on Application of Binocular Vision Technology in Field Obstacle Detection You may also like Vision measurement for flat parts based on local line-angle contour segmentation Precise pose and radius estimation of circular target based on binocular vi. *IOP Conference Series: Materials Science and Engineering*. https://doi.org/10.1088/1757-899X/806/1/012025

Irsyad Bin Sahalan, M. (2018). *DEVELOPMENT OF OBSTACLE AVOIDANCE TECHNIQUE FOR MULTI AGENT AUTONOMOUS SURFACE VESSEL USING OPTIMAL RECIPROCAL COLLISION AVOIDANCE*.

Jawad, H. M., Nordin, R., Gharghan, S. K., Jawad, A. M., & Ismail, M. (2017). Energy-efficient wireless sensor networks for precision agriculture: A review. In *Sensors (Switzerland)* (Vol. 17, Issue 8). MDPI AG. https://doi.org/10.3390/s17081781

Jaycar Ltd. (2023). *Jaycar KJ8916 Robotic Arm*. https://www.jaycar.com.au/robot-arm-kit-with-controller/p/KJ8916

Jia, X., Feng, Q., Fan, T., & Lei, Q. (2012). RFID technology and its applications in Internet of Things (IoT). *2012 2nd International Conference on Consumer Electronics, Communications and Networks, CECNet 2012 - Proceedings*, 1282–1285. https://doi.org/10.1109/CECNET.2012.6201508

Jiang, X., Zong, Y., Liu, Z., Wang, X., & Chen, Z. (2010). Research on combinational image stabilization technology based on MEMS IMU. *Optoelectronic Imaging and Multimedia Technology*, *7850*, 785002. https://doi.org/10.1117/12.868889

Jiménez, F., Naranjo, J. E., Gómez, O., & Anaya, J. J. (2014). Vehicle tracking for an evasive manoeuvres assistant using low-cost ultrasonic sensors. *Sensors (Switzerland)*, *14*(12), 22689–22705. https://doi.org/10.3390/S141222689

Johnstone-Wallace, D. B., & Kennedy, K. (1944). Grazing management practices and their relationship to the behaviour and grazing habits of cattle. *The Journal of Agricultural Science*, *34*(4), 190–197. https://doi.org/10.1017/S0021859600023649

Kapoor, R., Ramasamy, S., Gardi, A., Van Schyndel, R., & Sabatini, R. (2018). Acoustic sensors for air and surface navigation applications. *Sensors (Switzerland)*, *18*(2). https://doi.org/10.3390/S18020499

Karpenko, A., Jacobs, D., Baek, J., & Levoy, M. (2011). *Digital Video Stabilization and Rolling Shutter Correction using Gyroscopes*.

Karuppuswamy, J. (2000). *Detection and Avoidance of Simulated Potholes in Autonomous Vehicles in an Unstructured Environment*.

Katiyar, S., & Dutta, A. (n.d.). PSO Based Path Planning and Dynamic Obstacle Avoidance in CG Space of a 10 DOF Rover; PSO Based Path Planning and Dynamic Obstacle Avoidance in CG Space of a 10 DOF Rover. *Advances in Robotics - 5th International Conference of The Robotics Society*. https://doi.org/10.1145/3478586

Katiyar, S., & Dutta, A. (2019). Path Planning and Obstacle Avoidance in CG Space of a 10 DOF Rover using RRT. *ACM International Conference Proceeding Series*. https://doi.org/10.1145/3352593.3352615

Kawai, S., Takeuchi, K., Shibata, K., & Horita, Y. (2014). A Smart Method to Distinguish Road Surface Conditions at Night-time using a Car-Mounted Camera. *IEEJ Transactions on Electronics, Information and Systems*, *134*(6), 878–884. https://doi.org/10.1541/IEEJEISS.134.878

Khan, M. N., & Ahmed, M. M. (2021). Weather and surface condition detection based on road-side webcams: Application of pre-trained Convolutional Neural Network. *International Journal of Transportation Science and Technology*. https://doi.org/10.1016/J.IJTST.2021.06.003

KISHORE, P. S. V., P. SURESH, K., SMRUTI, D., & RAMESH, K. (2018). SOLAR POWERED OBSTACLE AVOIDING LAWN MOWER. *I-Manager’s Journal on Electrical Engineering*, *12*(2), 1. https://doi.org/10.26634/JEE.12.2.14169

Kleeman, L., & Kuc, R. (2008). Sonar Sensing. *Springer Handbook of Robotics*, *January 2008*. https://doi.org/10.1007/978-3-540-30301-5

Koenig, S., Likhachev, M., & Furcy, D. (2004). Lifelong Planning A∗. *Artificial Intelligence*, *155*(1–2), 93–146. https://doi.org/10.1016/J.ARTINT.2003.12.001

Krotkov, E., Simmons, R., Cozman, F., & Koenig, S. (n.d.). *Safeguarded Teleoperation for Lunar Rovers: From Human Factors to Field Trials*.

Kusuma Arbawa, Y., Utaminingrum, F., Setiawan, E., sitasi, C., Arbawa, Y. K., Utaminingrum, F., & Setiawan, E. (2021). Three combination value of extraction features on GLCM for detecting pothole and asphalt road. *Jurnal Teknologi Dan Sistem Komputer*, *9*(1), 64–69. https://doi.org/10.14710/JTSISKOM.2020.13828

Kwaśniewski, K. K., & Gosiewski, Z. (2018). Genetic algorithm for mobile robot route planning with obstacle avoidance. *Acta Mechanica et Automatica*, *12*(2), 151–159. https://doi.org/10.2478/AMA-2018-0024

La Rosa, F., Celvisia Virzì, M., Bonaccorso, F., & Branciforte STMicroelectronics, M. (n.d.). *Optical Image Stabilization (OIS)*.

Landaluce, H., Arjona, L., Perallos, A., Falcone, F., Angulo, I., & Muralter, F. (2020). A Review of IoT Sensing Applications and Challenges Using RFID and Wireless Sensor Networks. *Sensors 2020, Vol. 20, Page 2495*, *20*(9), 2495. https://doi.org/10.3390/S20092495

Lavric, A., & Popa, V. (2018). Performance Evaluation of LoRaWAN Communication Scalability in Large-Scale Wireless Sensor Networks. *Wireless Communications and Mobile Computing*, *2018*. https://doi.org/10.1155/2018/6730719

Lebakula, V., Tang, B., Goodin, C., & Bethel, C. L. (2021). Shape Estimation of Negative Obstacles for Autonomous Navigation. *IEEE International Conference on Intelligent Robots and Systems*, 4525–4531. https://doi.org/10.1109/IROS51168.2021.9636250

Lecun, Y., Muller, U., Ben, J., Cosatto, E., & Flepp, B. (n.d.). *Off-Road Obstacle Avoidance through End-to-End Learning*. Retrieved June 6, 2022, from http://yann.lecun.com

Li, C., Peng, Z., Huang, T. Y., Fan, T., Wang, F. K., Horng, T. S., Munoz-Ferreras, J. M., Gomez-Garcia, R., Ran, L., & Lin, J. (2017). A Review on Recent Progress of Portable Short-Range Noncontact Microwave Radar Systems. *IEEE Transactions on Microwave Theory and Techniques*, *65*(5), 1692–1706. https://doi.org/10.1109/TMTT.2017.2650911

Li, W., Tan, M., Wang, L., & Wang, Q. (2019). A cubic spline method combing improved particle swarm optimization for robot path planning in dynamic uncertain environment. *International Journal of Advanced Robotic Systems*, *16*(6). https://doi.org/10.1177/1729881419891661

Ljubičić, R., Strelnikova, D., Perks, M. T., Eltner, A., Peña-Haro, S., Pizarro, A., Fortunato, S., Sasso, D., Scherling, U., Vuono, P., & Manfreda, S. (2021). A comparison of tools and techniques for stabilising UAS imagery for surface flow observations. *Hydrology and Earth System Sciences*, *Discussion*. https://doi.org/10.5194/hess-2021-112

Lu, L., Yunda, A., Carrio, A., & Campoy, P. (2020). Robust autonomous flight in cluttered environment using a depth sensor. *International Journal of Micro Air Vehicles*, *12*. https://doi.org/10.1177/1756829320924528

Lwowski, J., Sun, L., Mexquitic-Saavedra, R., Sharma, R., & Pack, D. (2014). A Reactive bearing angle only obstacle avoidance technique for unmanned ground vehicles. *International Conference of Control, Dynamic Systems, and Robotics*. https://doi.org/10.11159/JACR.2014.004

Madgwick, S. O. H. (2009). *Madgwick Algorithm*. https://x-io.co.uk/downloads/madgwick\_algorithm\_matlab.zip

Madgwick, S. O. H. (2010). *An efficient orientation filter for inertial and inertial/magnetic sensor arrays | Resourcium*. https://resourcium.org/resource/efficient-orientation-filter-inertial-and-inertialmagnetic-sensor-arrays

Madgwick, S. O. H., Harrison, A. J. L., Sharkey, P. M., Vaidyanathan, R., & Harwin, W. S. (2013). Measuring motion with kinematically redundant accelerometer arrays: Theory, simulation and implementation. *Mechatronics*, *23*(5), 518–529. https://doi.org/10.1016/J.MECHATRONICS.2013.04.003

Madgwick, S. O. H., Harrison, A. J. L., & Vaidyanathan, R. (2011). Estimation of IMU and MARG orientation using a gradient descent algorithm. *IEEE International Conference on Rehabilitation Robotics*, 1–7. https://doi.org/10.1109/ICORR.2011.5975346

Madgwick, S. O. H., Wilson, S., Turk, R., Burridge, J., Kapatos, C., & Vaidyanathan, R. (2020). An Extended Complementary Filter for Full-Body MARG Orientation Estimation. *IEEE/ASME TRANSACTIONS ON MECHATRONICS*, *25*(4), 2054–2064. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9103115

Mahony, R., Hamel, T., & Pflimlin, J. M. (2005). Complementary filter design on the special orthogonal group SO(3). *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference, CDC-ECC ’05*, *2005*, 1477–1484. https://doi.org/10.1109/CDC.2005.1582367

Mahony, R., Hamel, T., & Pflimlin, J. M. (2008). Nonlinear complementary filters on the special orthogonal group. *IEEE Transactions on Automatic Control*, *53*(5), 1203–1218. https://doi.org/10.1109/TAC.2008.923738

Mai, Y., Zhao, H., & Guo, S. (2012). The analysis of image stabilization technology based on small-UAV airborne video. *Proceedings - 2012 International Conference on Computer Science and Electronics Engineering, ICCSEE 2012*, *3*, 586–589. https://doi.org/10.1109/ICCSEE.2012.77

Manderson, A., & Hunt, C. (2013). INTRODUCING THE AGRI-ROVER: AN AUTONOMOUS ON-THE-GO SENSING ROVER FOR SCIENCE AND FARMING. *Accurate and Efficient Use of Nutrients on Farms. Occasional Report No. 26*.

Manduchi, R., Castano, A., Talukder, A., & Matthies, L. (2005). Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation. *Autonomous Robots 2005 18:1*, *18*(1), 81–102. https://doi.org/10.1023/B:AURO.0000047286.62481.1D

Mannar, S., Thummalapeta, M., Saksena, S. K., & Omkar, S. N. (2018). *Vision-based Control for Aerial Obstacle Avoidance in Forest Environments*. *51*(1), 480–485. https://doi.org/10.1016/J.IFACOL.2018.05.081

Marins, J. L., Yun, X., Bachmann, E. R., McGhee, R. B., & Zyda, M. J. (2001). An extended Kalman filter for quaternion-based orientation estimation using MARG sensors. *IEEE International Conference on Intelligent Robots and Systems*, *4*, 2003–2011. https://doi.org/10.1109/IROS.2001.976367

Martina, N., Hicksona, R., Draganovab, I., Horneb, D., Kenyona, P., & Morris, S. (2015). *Walking distance and energy expenditure of beef cows grazing on hill country in winter*. Proceedings of the New Zealand Society of Animal Production. http://www.sciquest.org.nz/elibrary/download/142531/BRIEF+COMMUNICATION%3A+Walking+distance+and+energy+expenditure+of+beef+cows+grazing+on+hill+country+in+winter?

Masehian, E., & Katebi, Y. (2014). Sensor-based motion planning of wheeled mobile robots in unknown dynamic environments. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *74*(3–4), 893–914. https://doi.org/10.1007/S10846-013-9837-3

Matlab. (2023). *Matlab PolyFit function*. https://au.mathworks.com/help/matlab/ref/polyfit.html

Matthies, L., & Rankin, A. (2003). Negative Obstacle Detection by Thermal Signature. *IEEE International Conference on Intelligent Robots and Systems*, *1*, 906–913. https://doi.org/10.1109/IROS.2003.1250744

Matthies, Larry, Kelly, A., Litwin, T., & Tharp, G. (1995). Obstacle detection for unmanned ground vehicles: A progress report. *Intelligent Vehicles Symposium, Proceedings*, 66–71. https://doi.org/10.1007/978-1-4471-1021-7\_52

Michels, J. (2005). High Speed Obstacle Avoidance using Monocolar Vision and Reinforcement Learning. *Machine Learning*, 1–8.

Mingkhwan, E., & Khawsuk, W. (2017). Digital image stabilization technique for fixed camera on small size drone. *Proceedings - ACDT 2017: 3rd Asian Conference on Defence Technology: Advance Research Collaboration on Defence Technology*, 12–19. https://doi.org/10.1109/ACDT.2017.7886149

Moeller, R. (2020). *GPS-Guided Autonomous Robot with Obstacle Avoidance and Path Optimization Agricultural Robot-Potato Virus Y View project GPS-Guided Autonomous Robot with Obstacle Avoidance and Path Optimization*. https://www.researchgate.net/publication/342916625

Mohammed, A. S., Amamou, A., Ayevide, F. K., Kelouwani, S., Agbossou, K., & Zioui, N. (2020). The Perception System of Intelligent Ground Vehicles in All Weather Conditions: A Systematic Literature Review. *Sensors 2020, Vol. 20, Page 6532*, *20*(22), 6532. https://doi.org/10.3390/S20226532

Mohanty, P. K., & Dewang, H. S. (2021). A smart path planner for wheeled mobile robots using adaptive particle swarm optimization. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, *43*(2). https://doi.org/10.1007/S40430-021-02827-7

Moravec, H. P. (n.d.). *Rover Visual Obstacle Avoidance*.

Morimoto, C., & Chellappa, R. (1998). Evaluation of image stabilization algorithms. *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, *5*, 2789–2792. https://doi.org/10.1109/ICASSP.1998.678102

Mubarak, M. N. (2013). *Degree Programme in Machine Automation Mubarak, Muhammad Nauman: Outdoor Obstacle Detection using Ultrasonic sensors for an autonomous vehicle ensuring safe operations*.

New Zealand IoT Alliance. (2019). *IoT Spectrum in New Zealand Spectrum available in New Zealand for IoT Systems*. http://www.iot.org.au/wp/wp-content/uploads/2016/12/IoTSpectrumFactSheet.pdf

Nguyen-Huu, P.-N., & Titus, J. (2009). *Reliability and Failure in Unmanned Ground Vehicle (UGV)*. http://grrc.engin.umich.edu/

Nguyen Viet, C., & Marshall, I. (n.d.). *VISION-BASED OBSTACLE AVOIDANCE FOR A SMALL, LOW-COST ROBOT*.

Noori, K. K., Kumar Mishra, M., & Abdullah, Z. (n.d.). *VISION-BASED OBSTACLE AVOIDANCE ON QUADCOPTER*. Retrieved June 4, 2022, from www.tjprc.org

Norouzian, F., Marchetti, E., Hoare, E., Gashinova, M., Constantinou, C., Gardner, P., & Cherniakov, M. (2019). Experimental study on low-THz automotive radar signal attenuation during snowfall. *IET Radar, Sonar and Navigation*, *13*(9), 1421–1427. https://doi.org/10.1049/IET-RSN.2018.5644

NZ Government. (2018). *National Animal Identification and Tracing (NAIT) programme | NZ Government*. https://www.mpi.govt.nz/animals/national-animal-identification-tracing-nait-programme/

Odeh, S., & Faqeh, R. (2009). *Vision-Based Obstacle Avoidance of Mobile Robot UsingQuantized Spatial Model Formal Analysis of Attestation in Trusted Execution Environments View project*. https://doi.org/10.3844/ajeassp.2009.611.619

Odelga, M., Kochanek, N., & Bulthoff, H. H. (2017). Efficient real-time video stabilization for UAVs using only IMU data. *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems, RED-UAS 2017*, 210–215. https://doi.org/10.1109/RED-UAS.2017.8101668

Ofir, N., Galun, M., Nadler, B., & Basri, R. (n.d.). *Fast Detection of Curved Edges at Low SNR*.

OneNZ. (2023a). *OneNZ and Starlink*. https://one.nz/why-choose-us/spacex/

OneNZ. (2023b). *OneNZ Network coverage*. https://one.nz/network/coverage/

Parhi, D. R., & Mohanty, P. K. (2016). IWO-based adaptive neuro-fuzzy controller for mobile robot navigation in cluttered environments. *International Journal of Advanced Manufacturing Technology*, *83*(9–12), 1607–1625. https://doi.org/10.1007/S00170-015-7512-5

Petterson, T. C. (2020). *Development of an electric vehicle for autonomous use on a New Zealand dairy farm*.

Rajavarshini, R., Shruthi, S., Mahanth, P., Kumar, B. C., & Suyampulingam, A. (2021). Comparative analysis of image processing techniques for obstacle avoidance and path deduction. *Journal of Physics, Conference Series*, *2070*(1), 1–10. https://doi.org/10.1088/1742-6596/2070/1/012121

Ramseyer, A., Boissy, A., Thierry, B., & Dumont, B. (2009). Individual and social determinants of spontaneous group movements in cattle and sheep. *Animal*, *3*(9), 1319–1326. https://doi.org/10.1017/S1751731109004790

Raspberry. (n.d.). *Buy a Raspberry Pi 4 Model B – Raspberry Pi*. Retrieved August 12, 2023, from https://www.raspberrypi.com/products/raspberry-pi-4-model-b/?variant=raspberry-pi-4-model-b-4gb

Ren, K., Karlsson, J., Liuska, M., Hartikainen, M., Hansen, I., & Jørgensen, G. H. M. (2020). A sensor-fusion-system for tracking sheep location and behaviour. *International Journal of Distributed Sensor Networks*, *16*(5). https://doi.org/10.1177/1550147720921776

Robots, U. (n.d.). *Universal Robots UR5 robotic arm*. https://www.universal-robots.com/download/manuals-cb-series/user/ur5/315/user-manual-ur5-cb-series-sw315-english-international-en/

Rodriguez F., S. A., Frémont, V., & Bonnifait, P. (2008). Extrinsic calibration between a multi-layer lidar and a camera. *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 214–219. https://doi.org/10.1109/MFI.2008.4648067

Rosique, F., Navarro, P. J., Fernández, C., & Padilla, A. (2019). A systematic review of perception system and simulators for autonomous vehicles research. *Sensors (Switzerland)*, *19*(3). https://doi.org/10.3390/S19030648

Ryu, Y. G., Roh, H. C., & Chung, M. J. (2010). Video stabilization for robot eye using IMU-aided feature tracker. *ICCAS 2010 - International Conference on Control, Automation and Systems*, 1875–1878. https://doi.org/10.1109/ICCAS.2010.5670177

Sabatini, A. M. (2006). Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing. *IEEE Transactions on Biomedical Engineering*, *53*(7), 1346–1356. https://doi.org/10.1109/TBME.2006.875664

Samarasinghe, S. (2006). *Neural Networks for Applied Sciences and Engineering: From Fundamentals to Complex Pattern Recognition*. Auerbach Publications. https://www.amazon.com/Neural-Networks-Applied-Sciences-Engineering-ebook/dp/B008GXJR9S

Schäfer, B. H., Proetzsch, M., & Berns, K. (2005). *Stereo-Vision-Based Obstacle Avoidance in Rough Outdoor Terrain*. www.deere.com

Schneider, R., & Wenger, J. (2003). High resolution radar for automobile applications. *Advances in Radio Science*, *1*, 105–111. https://doi.org/10.5194/ARS-1-105-2003

Schwertfeger, S., Birk, A., & Bülow, H. (2011). Using iFMI spectral registration for video stabilization and motion detection by an Unmanned Aerial Vehicle (UAV). *9th IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2011*, 61–67. https://doi.org/10.1109/SSRR.2011.6106770

Sethi, P., & Sarangi, S. R. (2017). Internet of Things: Architectures, Protocols, and Applications. *Journal of Electrical and Computer Engineering*, *2017*, 1–25. https://doi.org/10.1155/2017/9324035

Shanmuga Sundaram, B. (2016). A quantitative analysis of 802.11ah wireless standard. *International Journal of Latest Research in Engineering and Technology*, *2*.

Sharma, P. S., & Chitaliya, N. G. (2007). Obstacle Avoidance Using Stereo Vision: A Survey. *International Journal of Innovative Research in Computer and Communication Engineering (An ISO*, *3297*(1). https://doi.org/10.15680/ijircce.2015.0301005

Sharma, S. U., & Shah, D. J. (2017). A Practical Animal Detection and Collision Avoidance System Using Computer Vision Technique. *IEEE Access*, *5*, 347–358. https://doi.org/10.1109/ACCESS.2016.2642981

Shen, H., Pan, Q., Cheng, Y., & Yu, Y. (2009). Fast video stabilization algorithm for UAV. *Proceedings - 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS 2009*, *4*, 542–546. https://doi.org/10.1109/ICICISYS.2009.5357609

Shene, T. N., Sridharan, K., & Sudha, N. (2016). Real-Time SURF-Based Video Stabilization System for an FPGA-Driven Mobile Robot. *IEEE Transactions on Industrial Electronics*, *63*(8), 5012–5021. https://doi.org/10.1109/TIE.2016.2551684

Shing, L. K., Lwin, K. Z., Sein, K. K., Chan, Y., & Thar, W. W. (2008). *Real-Time Obstacle Avoidance and Fuzzy Dynamic Steering Control for an Unmanned Ground Vehicle*. 978–983.

*Short Range Devices GURL | Radio Spectrum Management New Zealand*. (n.d.). Retrieved December 10, 2020, from https://www.rsm.govt.nz/licensing/frequencies-for-anyone/short-range-devices-gurl

Siegwart, R., Nourbakhsh, I. R., & Scaramuzza, D. (2011). *Introduction to Autonomous Mobile Robots, Second Edition | The MIT Press*. https://mitpress.mit.edu/books/introduction-autonomous-mobile-robots-second-edition

SigFox. (2023a). *Coverage - Sigfox 0G Technology*. https://www.sigfox.com/coverage/

SigFox. (2023b). *What is Sigfox? - Sigfox 0G Technology*. https://www.sigfox.com/what-is-sigfox/

Simmons, R., Henriksen, L., Chrisman, L., & Whelan, G. (n.d.). *Obstacle Avoidance and Safeguarding for a Lunar Rover*.

Šljivo, A., Kerkhove, D., Tian, L., Famaey, J., Munteanu, A., Moerman, I., Hoebeke, J., & De Poorter, E. (2018). Performance evaluation of IEEE 802.11ah networks with high-throughput bidirectional traffic. *Sensors (Switzerland)*, *18*(2), 1–28. https://doi.org/10.3390/s18020325

Spark. (2021). *Spark Network Coverage*.

Spark NZ. (2023). *Spark NZ network coverage*. https://www.spark.co.nz/shop/mobile/network.html

Sparkfun. (n.d.). *SparkFun 9DoF IMU Breakout - ICM-20948 (Qwiic) - SEN-15335 - SparkFun Electronics*. Retrieved August 12, 2023, from https://www.sparkfun.com/products/15335

Spenko, M., Kuroda, Y., Dubowsky, S., & Iagnemma, K. (2006). Hazard avoidance for high-speed mobile robots in rough terrain. *Journal of Field Robotics*, *23*(5), 311–331. https://doi.org/10.1002/ROB.20118

*Standards for the IoT*. (2016). https://www.3gpp.org/news-events/1805-iot\_r14

Starlink. (2022). *Starlink satellite coverage*. https://satellitemap.space

Starlink. (2023). *Everything you need to know about Starlink in New Zealand*. https://starlink.co.nz/starlink-new-zealand/

Sugihara, R., & Gupta, R. K. (2011). Path Planning of Data Mules in Sensor Networks. *ACM Transactions on Sensor Networks (TOSN)*, *8*(1), 27. https://doi.org/10.1145/1993042.1993043

Suh, Y. S. (2019). Laser Sensors for Displacement, Distance and Position. *Sensors (Basel, Switzerland)*, *19*(8). https://doi.org/10.3390/S19081924

Sulaimon Alli, K., Oluwafemi Onibonoje, M., Oluwole, A. S., Adegoke Ogunlade, M., Mmonyi, A. C., Ayamolowo, O., & Olushola Dada, S. (2018). *DEVELOPMENT OF AN ARDUINO-BASED OBSTACLE AVOIDANCE ROBOTIC SYSTEM FOR AN UNMANNED VEHICLE*. *13*(3). www.arpnjournals.com

Sun, J., Rehg, J. M., & Bobick, A. (n.d.). *Learning for Ground Robot Navigation with Autonomous Data Collection*.

Swain, D., Gregg, D., Bishop-Hurley, G., Trotter, M., & Petty, S. (2013). On property benefits of Precision Livestock Management. In *Meat & Livestock Australia* (Vol. 364, Issue July).

Tamiya. (2012). *Tamiya Bruiser RC Model*. https://tamiyabase.com/tamiya-models/58519

Tang, P., Zhang, R., Liu, D., Huang, L., Liu, G., & Deng, T. (2015). Local reactive obstacle avoidance approach for high-speed unmanned surface vehicle. *Ocean Engineering*, *106*, 128–140. https://doi.org/10.1016/J.OCEANENG.2015.06.055

Team, E. (2019). *Bluetooth 5 range put to the test*. https://www.eedesignit.com/bluetooth-5-range-put-to-the-test/

Thillainayagi, R., & Senthil Kumar, K. (2017). Video stabilization technique for thermal infrared Aerial surveillance. *Proceedings of 2016 Online International Conference on Green Engineering and Technologies, IC-GET 2016*. https://doi.org/10.1109/GET.2016.7916630

Tian, L., Santi, S., Seferagić, A., Lan, J., & Famaey, J. (2021). Wi-Fi HaLow for the Internet of Things: An up-to-date survey on IEEE 802.11ah research. *Journal of Network and Computer Applications*, *182*, 103036. https://doi.org/10.1016/J.JNCA.2021.103036

Tran, T.-Y., Houre, E., & Clarke, N. (2015). *Wading depth estimation for a vehicle*. United States Patent. https://www.freepatentsonline.com/9026310.pdf

van Hecke, K., de Croon, G., van der Maaten, L., Hennes, D., & Izzo, D. (2018). Persistent self-supervised learning: From stereo to monocular vision for obstacle avoidance: *Https://Doi.Org/10.1177/1756829318756355*, *10*(2), 186–206. https://doi.org/10.1177/1756829318756355

Waitzman, D. (1990). *RFC 1149*. https://datatracker.ietf.org/doc/html/rfc1149

Wallace, A. M., Halimi, A., & Buller, G. S. (2020). Full Waveform LiDAR for Adverse Weather Conditions. *IEEE Transactions on Vehicular Technology*, *69*(7), 7064–7077. https://doi.org/10.1109/TVT.2020.2989148

Wang, J., Song, Q., Jiang, Z., & Zhou, Z. (2016). A novel InSAR based off-road positive and negative obstacle detection technique for unmanned ground vehicle. *International Geoscience and Remote Sensing Symposium (IGARSS)*, *2016*-*November*, 1174–1177. https://doi.org/10.1109/IGARSS.2016.7729297

Wang, L., Zhao, H., Guo, S., Mai, Y., & Liu, S. (2012). The adaptive compensation algorithm for small UAV image stabilization. *2012 IEEE International Geoscience and Remote Sensing Symposium*, 4391–4394. https://doi.org/10.1109/IGARSS.2012.6350400

Wang, S., Wang, L., He, X., & Cao, Y. (2021). A Monocular Vision Obstacle Avoidance Method Applied to Indoor Tracking Robot. *Drones 2021, Vol. 5, Page 105*, *5*(4), 105. https://doi.org/10.3390/DRONES5040105

Wangchuk, K., Wangdi, J., & Mindu, M. (2017). Comparison and reliability of techniques to estimate live cattle body weight. *Http://Www.Tandfonline.Com/Action/JournalInformation?Show=aimsScope&journalCode=taar20#.VsXoziCLRhE*, *46*(1), 349–352. https://doi.org/10.1080/09712119.2017.1302876

*What is LoRaWAN® | LoRa Alliance®*. (n.d.). Retrieved November 18, 2020, from https://lora-alliance.org/resource-hub/what-lorawanr

Williams, L. R., Fox, D. R., Bishop-Hurley, G. J., & Swain, D. L. (2019). Use of radio frequency identification (RFID) technology to record grazing beef cattle water point use. *Computers and Electronics in Agriculture*, *156*, 193–202. https://doi.org/10.1016/j.compag.2018.11.025

Wiriyaprasat, K., & Ruchanurucks, M. (2015). Realtime VDO stabilizer for small UAVs using a modified homography method. *Proceedings 2015 International Conference on Science and Technology, TICST 2015*, 40–43. https://doi.org/10.1109/TICST.2015.7369337

Wong, D. C., Bui, K., Nguyen, L. H., Smith, G., & Ton, T. T. (2003). Integration of differential global positioning system with ultrawideband synthetic aperture radar for forward imaging. *Https://Doi.Org/10.1117/12.488581*, *5083*, 529–538. https://doi.org/10.1117/12.488581

X-IO. (2008). *GitHub - xioTechnologies/Fusion*. https://github.com/xioTechnologies/Fusion

Y Ropert-Coudert, R. W. (2005). Trends and perspectives in animal-attached remote sensing. *Front Ecol Environ*, *3*, 437–444. https://doi.org/10.1890/1540-9295(2005)003[0437:tapiar]2.0.co

Yamauchi, B. (2006). Autonomous urban reconnaissance using man-portable UGVs. *Https://Doi.Org/10.1117/12.660435*, *6230*, 261–271. https://doi.org/10.1117/12.660435

Yamauchi, B. (2008). All-weather perception for small autonomous UGVs. *Https://Doi.Org/10.1117/12.776792*, *6962*, 28–37. https://doi.org/10.1117/12.776792

Yan, W. Y., Shaker, A., & El-Ashmawy, N. (2015). Urban land cover classification using airborne LiDAR data: A review. *Remote Sensing of Environment*, *158*, 295–310. https://doi.org/10.1016/J.RSE.2014.11.001

Yean, S., Lee, B. S., Yeo, C. K., Vun, C. H., & Oh, H. L. (2018). Smartphone Orientation Estimation Algorithm Combining Kalman Filter with Gradient Descent. *IEEE Journal of Biomedical and Health Informatics*, *22*(5), 1421–1433. https://doi.org/10.1109/JBHI.2017.2780879

Yibin, L., Caihong, L., & Zijian, Z. (2006). Q-learning based method of adaptive path planning for mobile robot. *Proceedings of IEEE ICIA 2006 - 2006 IEEE International Conference on Information Acquisition*, 983–987. https://doi.org/10.1109/ICIA.2006.305871

Yoo, T. S., Hong, S. K., Yoon, H. M., & Park, S. (2011). Gain-Scheduled Complementary Filter Design for a MEMS Based Attitude and Heading Reference System. *Sensors 2011, Vol. 11, Pages 3816-3830*, *11*(4), 3816–3830. https://doi.org/10.3390/S110403816

Yuan, Q., Asadi, E., Lu, Q., Yang, G., & Chen, I. M. (2019). Uncertainty-Based IMU Orientation Tracking Algorithm for Dynamic Motions. *IEEE/ASME Transactions on Mechatronics*, *24*(2), 872–882. https://doi.org/10.1109/TMECH.2019.2892069

Zhang, J., Huang, M., Jin, X., & Li, X. (2017). A real-time Chinese traffic sign detection algorithm based on modified YOLOv2. *Algorithms*, *10*(4). https://doi.org/10.3390/A10040127

Zhang, Z. (2012). Microsoft kinect sensor and its effect. *IEEE Multimedia*, *19*(2), 4–10. https://doi.org/10.1109/MMUL.2012.24

Zhong, Z., Wang, Z., Lin, L., Liang, H., & Xu, F. (2020). Robust Negative Obstacle Detection in Off-Road Environments Using Multiple LiDARs. *2020 6th International Conference on Control, Automation and Robotics, ICCAR 2020*, 700–705. https://doi.org/10.1109/ICCAR49639.2020.9108058

Zhou, L., Li, Z., & Kaess, M. (2018). Automatic Extrinsic Calibration of a Camera and a 3D LiDAR Using Line and Plane Correspondences. *IEEE International Conference on Intelligent Robots and Systems*, 5562–5569. https://doi.org/10.1109/IROS.2018.8593660

Zhu, J., Li, C., & Xu, J. (2016). Digital Image Stabilization for Cameras on Moving Platform. *Proceedings - 2015 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, IIH-MSP 2015*, 255–258. https://doi.org/10.1109/IIH-MSP.2015.23

Zhu, W. H., & Lamarche, T. (2007). Velocity estimation by using position and acceleration sensors. *IEEE Transactions on Industrial Electronics*, *54*(5), 2706–2715. https://doi.org/10.1109/TIE.2007.899936

Zogopoulos, E. (2021). *Battery Technology: A New Era Emerging | Energy Industry Review*. Energy Industry Review. https://energyindustryreview.com/analysis/battery-technology-a-new-era-emerging/