APPLICATIONS FOR SENSOR FUSION IN VERTICAL TRANSPORTATION

A Dissertation
Presented to
The Academic Faculty

by

Andrew J Evert

In Partial Fulfillment
of the Requirements for the Degree
Mechanical Engineering in the
Woodruff School of Mechanical Engineering

Georgia Institute of Technology
Universität Stuttgart
August 2018

APPLICATIONS FOR SENSOR FUSION IN VERTICAL TRANSPORTATION

Approved by:

Dr. Christopher Saldana, Advisor
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Thomas Kurfess
School of Mechanical Engineering
Georgia Institute of Technology

Dr. Oliver Sawodny
Institute of System Dynamics
Universität Stuttgart

Thomas Ehrl
Expertise in Vertical Transportation
thyssenkrupp AG

Date Approved: July 23, 2018

ACKNOWLEDGEMENTS

I would like to thank Thomas Felis for starting me on this path leading to a Joint Master's Degree from both Georgia Tech and the University of Stuttgart. I would like to thank my family for understanding my choice to move to Germany to complete my studies. Finally, I would like to thank Dr. Christopher Saldana. With his guidance and willingness to assist me long distance, I have been able to complete such a venture.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ANN	Artificial Neural Network
BA	Business Area
BU	Business Unit
CPS	Cyber-Physical Systems
FST	Field Service Technician
IOT	Internet of Things
IIOT	Industrial Internet of Things
IMU	Inertial Measurement Unit
MRL	Machine-Roomless
SOA	Service Oriented Architecture
tkE	thyssenkrupp Elevator
USD	United States Dollar

SUMMARY

Thyssenkrupp recently unveiled a new product titled MULTI. The marketing video announcing the product starts with the notion that the elevator industry has existed largely unchanged for 160 years. While meant to describe how the horizontal motion and ropeless features of the MULTI are changing the industry, the point that the elevator industry is relatively unchanged since its inception still holds true. This thesis explores the current state of the elevator industry as it relates to modern topics such as Industry 4.0 and Industrial Internet of Things and where there is room for improvement internally at thyssenkrupp Elevator. Maintenance logs are analyzed to determine scenarios where monitoring with sensors could be useful to save time or to improve maintenance quality. Different sensor fusion algorithms are also examined for their relevance to different aspects of elevator systems such as maintenance or development. These topics, while relatively limited in scope, serve as a complement to other previous and ongoing projects all relating to changing elevators from their current, unconnected state, to smarter, cyber physical systems.

CHAPTER 1. INTRODUCTION

1.1 Research Motivation

The elevator industry is growing and evolving rapidly after a century and a half of relatively little change. Building height continues to increase and the elevator industry is pursuing technological advances to support the larger height required to access the entire building. Building design also now utilize more complex shapes as building techniques and materials improve; this can require multiple separate shafts to reach every floor. In short, the number of elevators is increasing as is the travel distance of all the elevators and this will lead to a more challenging and complex maintenance requirement for elevator systems.

Field service is a major activity for elevator companies and maintenance contracts generate substantial revenue. In general, maintenance activities are typically performed by a minimum of two technicians, and this number will increase with the increase in elevator complexity, travel distance, and number of shafts. Due to this increase in complexity, automation of tasks or the addition of tools to decrease lead time is required. Automation in the context of this paper entails the addition of sensors, monitoring systems, and sensing algorithms that can decrease diagnostic or maintenance time requirements and potentially increase diagnostic capabilities and accuracy.

Elevators, in their current state, have limited sensor capabilities. Examples of sensors deployed in elevator systems are in the cab, which has levelling sensors to ensure that the floor of the cab is even with the floor of the building, and in the door, which has a curtain of light sensors to prevent passengers from being impacted by the door. However, these sensors are rudimentary at best, with limited interaction with each other, and provide little

to no useful feedback for the operating company or the technicians. To achieve the benefits associated with ubiquitous sensing, the industry has interest in placing sensors on major aspects of the elevator and multiple sensors per subsystem. These sensors should feed data into a central hub that can combine values from across the systems, and even individual subsystems, to generate a holistic view of the overall health of the elevator.

The prior work has focused on archiving data so to use historical information to better predict failures. In this regard, it is of significant interest to draw conclusions regarding maintenance for elevator health from incomplete data sets that include data from sensors covering every aspect of elevator operation. The present thesis is focused on addressing this gap by leveraging sensor fusion methods and algorithms, as well as retrofit sensor data for elevator systems, so to determine elevator health conditions and operational performance metrics.

1.2 Contributions

This research initially focuses on the technician with the intent of reducing lead time for maintenance. Some tasks require much more time to complete than others and the longer time requirements can stem from a variety of reasons including safety and the complexity of the task. Therefore, conducting a simple measurement with a sensor could be a valuable tool. This paper also reviews how the sensors could be implemented in a way that allows for the use of various sensor fusion methods not only related to maintenance, but also for more theoretical purposes in other elevator monitoring efforts. This project showcases how these sensors and sensor fusion methods set the groundwork for future projects and could be the foundation of more robust and intelligent, predictive or preventative maintenance systems, while also demonstrating how sensor fusion can be

beneficial in other areas. These general benefits stem from work done previously in the area of elevator monitoring. These projects include the MAX data acquisition used for data analytics, Maintenance on Demand which deals with fault detection, and the Elevator Health Check which test vibrations for passenger comfort. The results of thesis will contribute to the future MAX 2.0 project that focuses on using data provided by the elevator for improved data analytics and predictive/preventative maintenance.

1.3 Research Approach and Thesis Structure

Chapter 2 reviews how industries, including the elevator industry, have adopted Industry 4.0. There are brief explanations of Big Data and data analytics, and sensor fusion. The chapter also discusses prior work in these areas. The chapter uses these projects to outline the existing gaps in the elevator industry and where there is potential for improvement.

Chapter 3 discusses the outcome of combining existing callback data from elevator maintenance logs of four different cities in the United States. The list is cross referenced with external inputs to determine possible areas where sensors could be impactful. It also examines the beginnings of a sensor network that monitors multiple aspects of a single subsystem to generate a more holistic view of the subsystem health. The chapter also examines the potential for the proposed systems to also be wireless. It delineates a method of a wireless communication network that will have the required functionality for the technician without compromising data integrity.

Chapter 4 tests possibilities of sensor fusion and enhanced data analytics by implementing these sensors, and additional sensors, with artificial data. These data are theoretical numbers used to explain the benefit of using a combination of the sensor fusion

methods discussed in chapter 2. This chapter also examines how sensor fusion could be applied to the MULTI project.

Chapter 5 summarizes the research and makes recommendations for any future iterations or projects. The work will likely be continued by the company and be a reference for multiple upcoming internal thyssenkrupp projects.

CHAPTER 2. BACKGROUND

2.1 Industry 4.0

Industry 4.0 is described as a wide collection of concepts including smart factories, human centric manufacturing systems, and cyber-physical systems [1]. A smart factory contains automated machines that can self-diagnose and self-regulate. Human-centric manufacturing is the idea that machines evolve based on human needs. Finally, cyber-physical systems (CPS) are systems where the digital and physical systems form a single new entity instead of two separate systems attached to each other. This new industrial revolution takes these concepts and focuses on the monitoring, self-diagnosis, and automation of manufacturing processes through the generation of real time data. An overview of a CPS is shown in Figure 1.

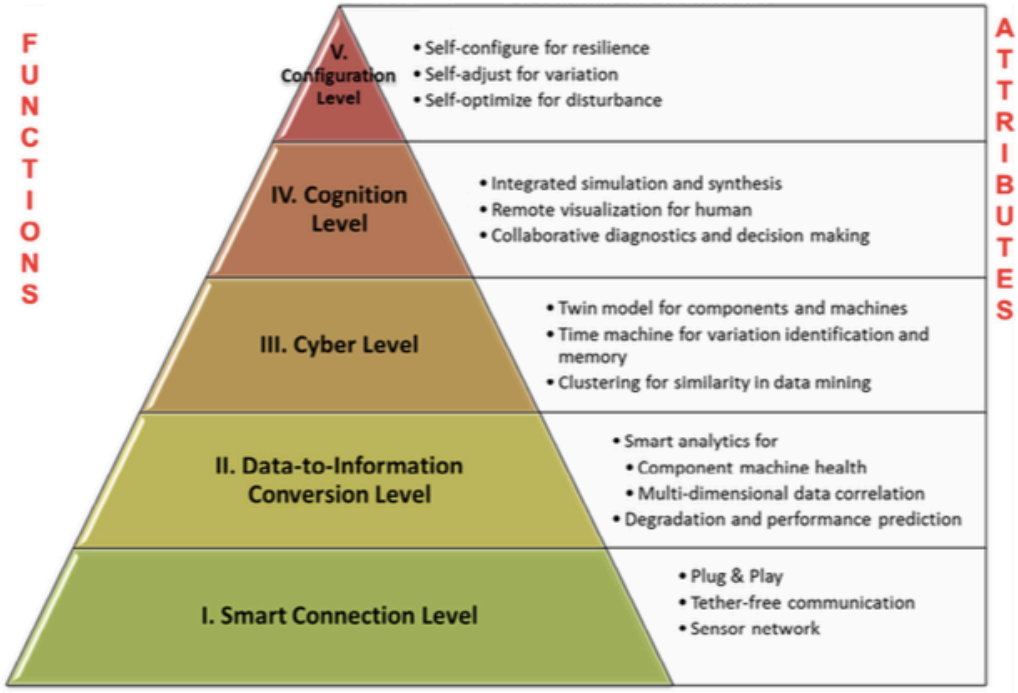


Figure 1: Hierarchy of a CPS [2]

As with any pyramid, the lower levels of Smart Connection, Data-to-Information Conversion, and Cyber levels are important to implement first. Sensors attached to a system forms the basis of a CPS, the Smart Connection, and allow progress in the next block to begin. The third block benefits from multiple machines all being monitored by the same types of sensors and being connected to a central network. This allows the data from all the machines to be directly compared. This comparison of each machine to the rest of the field provides the possibility to discern which machines are experiencing faults. The second to last block deals with advanced analytics including, but not limited to, predictive maintenance. The machine network can use collected data and fault trends to predict when failures will occur. The final block incorporates a feedback loop that gives control over the system to the system itself. The system will make decisions to change its parameters based on when it predicts that a fault will occur. As each level is added, less human interaction is required but still necessary.

Based on the CPS hierarchy, it seems that sensors can have a tremendous impact on a machine. Sensors allow a device to interact with its environment or monitor certain parameters of the device itself [3]. Due to these abilities, sensors have the capability to turn a normal machine or device into a smart or intelligent product. As explained by Schmidt, these smart products can generate not only their name, but also the current and historical status of multiple properties: environmental or internal [4]. Sensors enable a system to perform a wide variety of tasks ranging from simple data collection to self-optimization via advanced data analytics and sensor/data fusion methods.

Combining multiple sensors and multiple machines into one sensor network greatly improves the ability and accuracy of a CPS. A single sensor can generate data about one aspect of a subsystem, but faults could occur in a number of different section meaning there is limited spatial coverage. Multiple sensors on one subsystem allow a network to collect data about every aspect to form a generalized overview allowing for reduced uncertainty. A network of these CPS's allows mass examination of multiple data points reducing imprecision when predicting faults and potentially also reducing any temporal coverage lapses. A CPS network can also introduce new data fields, such as device age and region, that by themselves for one machine would not mean much, but when combined could generate new datasets that were previously unknown to have any impact on a machine.

2.2 Industrial Internet of Things (IIOT)

A CPS is commonly acknowledged as a piece of building block of the Internet of Things (IOT). A widely accepted definition of IOT is "a dynamic global network infrastructure with self- configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'Things' have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network" [5]. Multiple industries are exploring or have implemented IOT including, but not limited to, manufacturing [6], healthcare [7], and transportation [8].

Benefits of IOT for these industries are varied. Manufacturing companies are using IOT for plant monitoring [9], IOT-based cloud manufacturing services [10], and evolving the uses of Mechatronics [11]. The Healthcare industry has developed solutions for human posture analysis [12] and wireless health monitoring of rural patients [13]. Finally, the

transportation industry has produced solutions for flight control [14], smart parking [15], and intelligent traffic information systems [16]. A major activity critical to these industries is maintenance, which is often the initial motivation for a system's venture into "smart" territory. The ability to monitor and report faults can be a precursor or simple starting point for many smart systems including automobiles, factories, and even home appliances. For example, some of the first sensors used widely in automobiles were diagnostic sensors such as the speedometer and fuel gauge. This led to other additions such as tire pressure sensors, air bags, all-wheel drive, brake monitoring, and others. In very recent events, companies are now testing the ability of cars to drive themselves and are now reaching the top of the hierarchy in Figure 1. As such, it could make sense for the elevator industry to begin from the same starting point: IOT for maintenance.

Maintenance can be divided into three main categories: Reactive, Predictive, and Preventative. The bottom of the CPS hierarchy in Figure 1, or the Smart Connection, is the level for Reactive Maintenance. A system is monitored and a notification is triggered when a fault occurs or a value passes a threshold. Levels two and three embody the Predictive Maintenance. Sensors on multiple machines are collecting data that is being used to estimate in advance when a failure could occur. The top two, or fourth and fifth, levels of the hierarchy describe Preventative Maintenance. The system is given control over itself so that it can change parameters and attempt to prevent, or at least delay, a failure from occurring. A different overview of these relations is shown in Figure 2 and explained in Table 1. The table provides a different way of dividing the architecture for IOT but ultimately achieves the same outcome as the delineation in [2]. The paper by Xu [5] also depicts the different layers in Figure 3.

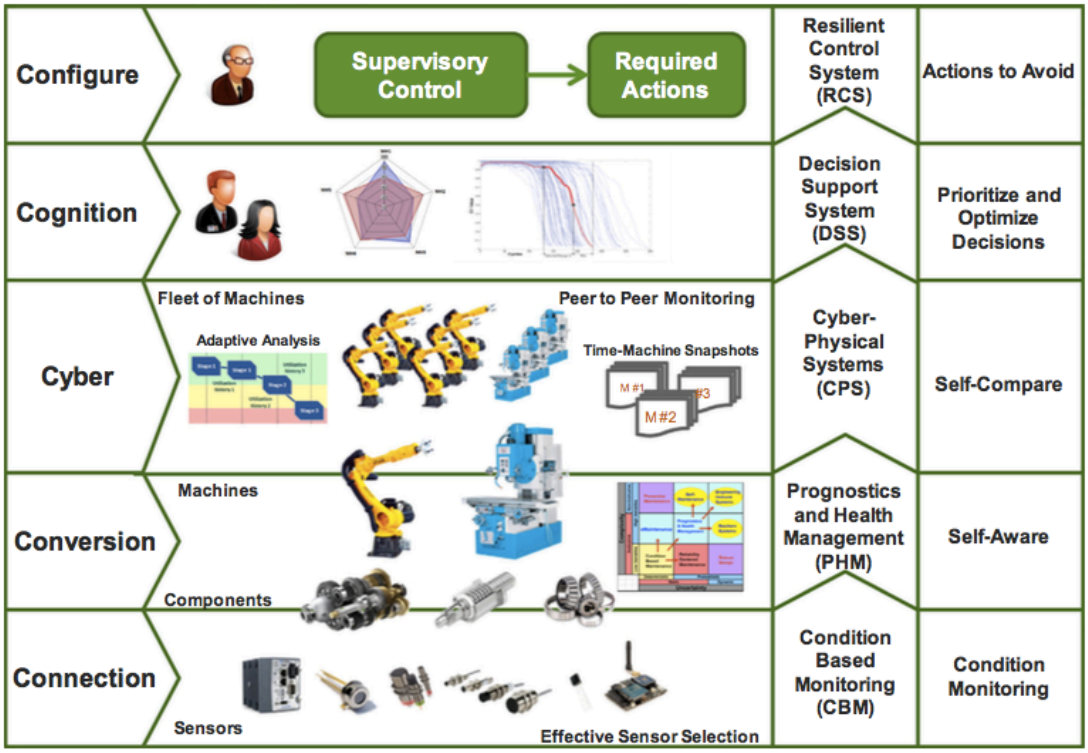


Figure 2: Applications for each level of CPS hierarchy [2]

Table 1 – A Four-Layered Architecture for IOT [5]

Layer	Count of Fault
Sensing Layer	This layer is integrated with existing hardware (RFID, sensors, actuators, etc.) to sense/control the physical world and acquire data
Networking Layer	This layer provides basic networking support and data transfer over wireless or wired network
Service Layer	This layer creates and manages services. It provides services to satisfy user needs
Interface Layer	This layer provides interaction methods to users and other applications

The service oriented architecture (SOA) for IOT is ideal for Maintenance as it is a service aspect of most industries including elevators. With a sensing layer as complicated as that shown in Figure 3, an issue arises with compounding and conflicting data sources. The

architecture is clear, but the analysis method is less certain. Therefore, there is a need to take the data and turn it into something useful. This occurs in the service layer of the SOA in Figure 3 and is commonly referred to as Big Data.

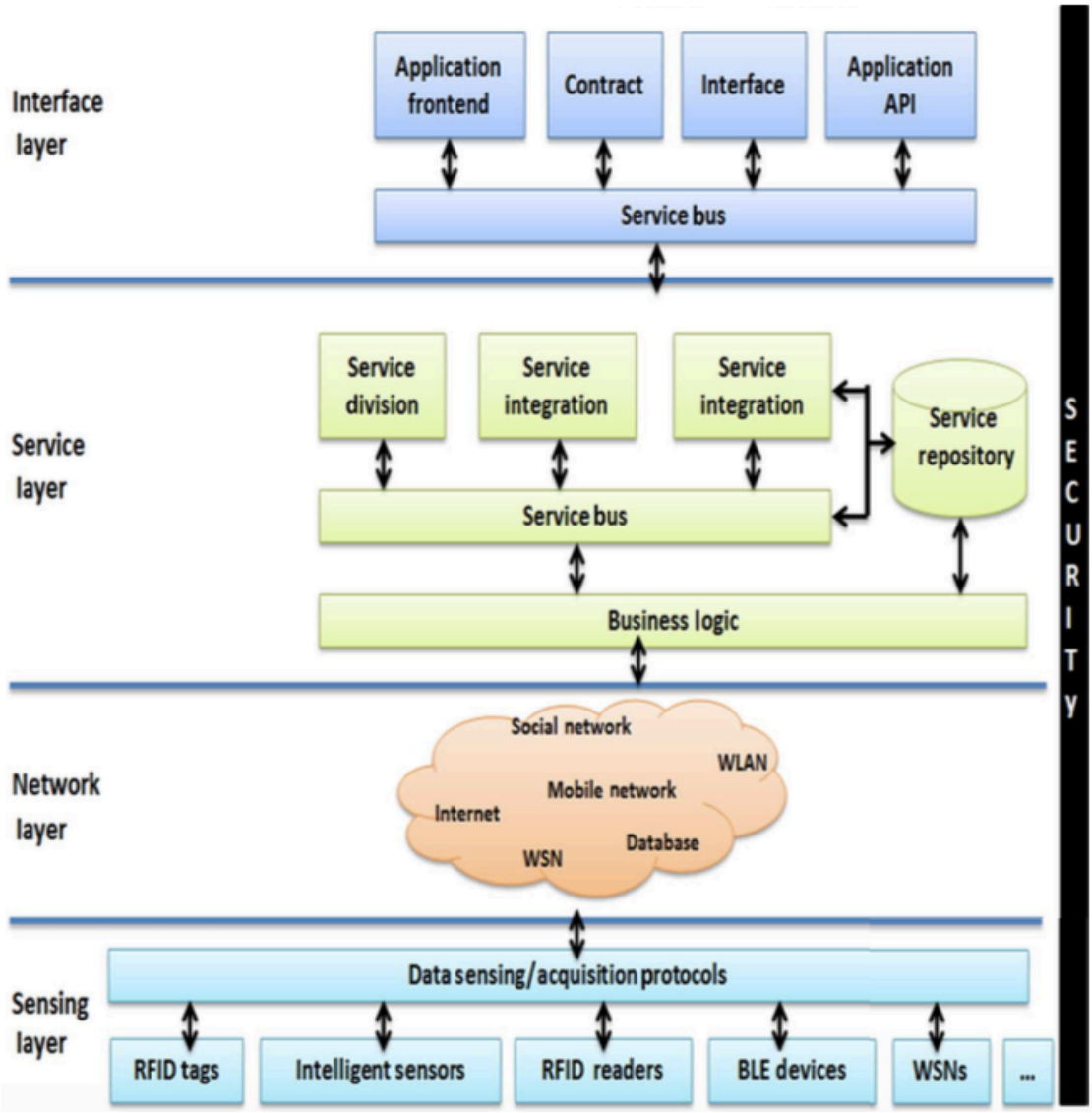


Figure 3: Service Oriented Architecture (SOA) for IOT [5]

2.3 Big Data and Data Analytics

The term "Big Data" has been used across multiple industries. Data is critical for supporting decision making and often ownership of data is a contentious issue. These issues

are due to the potentially immense benefit that big data offers. As mentioned in Ref. [17], "You can't manage what you don't measure", implying there is paradigm shift with big data because parties can now measure aspects or parameters that were previously either unmeasurable or difficult to measure. One example in McAfee's article is Amazon. Amazon was originally an online book retailer. However, one advantage they had over traditional book stores, besides the convenience of being online, was their recommendation system. Traditional book stores might make recommendations based on static reference lists such as the New York Times best seller list, but these suggestions might not appeal to every consumer. In comparison, Amazon's recommendations are predictive and data-driven. In this regard, Amazon can compare data on millions of consumers, thus providing a better estimate of what book a consumer will want to read even before they decide. Before big data and data analytics, bookstores only knew what books consumers would read after they read them and even then, they likely would not record that information or at least not on large scales. Amazon found a way to measure book sales and, if it can be measured, then it can be managed.

A similar concept applies to a vast number of industries. Estimated arrival times of packages, airplanes, or commute to work can all be enhanced by big data. Manufacturing times could be improved by reducing equipment failures. Healthcare could also benefit from big data with potential for increased accuracy of diagnoses. However, once a party has an idea of what they want to improve, the question is how they can turn a large collection of data into a meaningful result and provide a noticeable adjustment. Part of this hinges on the quality of data recorded. As stated in Ref. [18], big data relies on "Volume, Variety, and Velocity". There must be large quantities of data of varying forms and metrics

and they must be collected rapidly. If the data collected fits these three metrics, then a company can begin extracting value.

Some ways of extracting information from the data are also described in Ref. [18] as text, audio, visual, social media, and predictive analytics. The methods used largely depend on both the data collected and the desired outcome, but generally at least one of these five methods is used. Text analytics are common due to the shear amount of written work already available, but the other methods are becoming popular. Social media analytics are becoming more accessible due to the emergence of Facebook and other sites. Audio and visual analytics are more widespread because cameras and microphones have become inexpensive. Due to the decreasing costs of cameras, microphones, and other sensors, it makes sense to use as many as will fit on a device. However, more is not always better, and there are some issues that can arise with an abundance of sensors. Their readings must be combined in such a way that the data produced is valid, and this fusion of sensor data using different combination techniques has its own area of research.

2.4 Sensor Fusion

Sensor fusion is the way or ways that a sensor network combines data from the sensing layer of Figure 3. Certain sensor types have large data bandwidth requirements and often it can be difficult to transmit these large data streams. Other scenarios require multiple sensors measuring the same parameter for redundancy. Some subsystems can also have multiple sensors measuring different parameters [19]. In each of these cases, it could make sense to implement sensor fusion. For the benefit of data processing algorithms, it is possible, and in some cases necessary, to simplify or condense the data on a local level before transmitting to the rest of the sensor network or CPS. For example, if multiple

sensors are recording the same parameter, this might involve combining these recordings into one value before transmitting. For single subsystems, it could make sense to combine all the data streams into one output before transmitting a single value representing the entire subsystem.

Given that there is a need for local sensor/data fusion, there are multiple sensor fusion methods available. Some of these include the Kalman-Filter [20], Dempster-Shafer evidence theory [21], the Bayesian method [22], Neural Networks [23] and others, as well as combinations of these methods. Each method might perform better depending on the scenario and the requirements of the CPS. For example, if multiple sensors of the same type are used, it could be more beneficial to use a neural network or statistical average. If varied types of sensors are being used, it could make sense to use Bayes, Dempster-Shafer, Kalman Filter, or even Fuzzy logic. Selection of approach depends on user preference, or a side by side comparison of available methods to determine which has the highest accuracy. To explain the methods in more detail, an example scenario will be covered later in chapter 4 using the sensors determined in chapter 3.

2.5 Previous Sensor Related Projects

Thyssenkrupp has worked on a myriad of projects relating to sensors in a variety of the fields discussed in this chapter. The MAX 1.0 project is the gateway to Big Data and data analytics as a connection between the elevator controller and cloud storage. Eventually more sensors will be fed through the MAX system and in the MAX 2.0 project. The project will decide which sensors to use and develop algorithms that make decisions based on the data.

The Maintenance on Demand was a project in the European business area (BA) that looked at determining a number of door faults simply with vibration and noise data. The project is a perfect candidate for sensor fusion methods and could be picked up again in the MAX 2.0 project using any similar finding from the ongoing project with the University of Northampton. The findings of the Maintenance on Demand project were not used in this paper other than when the sensing scenario is chosen in chapter 3, but they could be useful in other projects.

Elevator Health Check is an ongoing project in the Americas BA that attempts to determine elevator health from vibration data. It its current state, a tool is used by the technician to measure the vibrations and it is not an onboard, permanent data collection system. The data is fed into and analyzed by a smart phone app that all the technicians were issued. The project idea heavily influenced the initial focus of this paper in terms of its design and final purpose. There could be other products, such as the MULTI, that fall into one or more of the Industry 4.0 topics covered in this chapter, but they are either in other BAs or they have not yet been made public. Regardless, even though sensors are widespread in thyssenkrupp, there are still gaps and room for improvement.

CHAPTER 3. SYSTEM DEFINITION

3.1 Existing Gaps in Elevator Industry

3.1.1 Callback Data

To have a representative sample of multiple regions and environment types, data was examined from four different cities from around the United States. These four cities were the only ones with pre-existing maintenance logs, but happened to ensure that the study captured issues that were not region specific (e.g. humidity related breakdowns) or dependent on the average unit age (e.g., Denver has more new installations than Syracuse). Not that these issues are unimportant, but they can be specific to a region and the idea was to find faults that are widespread. Each data set contains information from the major city branch field offices over a three-year period. In total, there are over three million callbacks in this period. Unfortunately, most of the data entries were either not entered in a way that facilitated mass examination, or the fault description was missing. After filtering out insufficient entries, over 100,000 total callbacks remained.

3.1.1.1 Denver Callbacks

The most frequent callbacks from the Denver area are shown in the following table. This list only shows the faults that generated at least 200 callbacks over the three-year period. The generic service request comes up the most as it is an easy input for the technician. Outside of this field, events related to the door operator came up the most. These callbacks generally deal with dirty sills, an impact, or another fault causing the door

operator to get stuck or to lock itself. The oil in hydraulic elevators is also mentioned frequently with varying faults, as are the brakes listed toward the bottom.

Table 2 – Denver Callbacks: Branch 106400

Subsystem	Fault Count
Service Request	5733
Controller Component	3315
Hoistway Door Gibbs & Interlock	2319
Door Operator	1165
Hall Push Button	671
Car Door, Gibbs & Interlock	622
Selector & Attachments	414
Low Oil Level	390
Car Push Button	382
Main Power Supply	358
Microprocessor	354
Communication Device	319
Jack Seal / Packing	317
Photo Cell / Electronic Door Edge	314
Door Linkage / Relating Cables	296
S. S. Circuit Board (s)	286
Drive	277
Clutch Assemly / Retiring Cam	253
Hydraulic Pump / Motor	247
Power Unit Value	243
Brake	231

3.1.1.2 Houston Callbacks

The most frequent callbacks from three Houston branches are shown in the following table. The following lists show the faults that generated at least 200 callbacks over the three-year period. Like the Denver branch, the door tends to come up frequently with similar if not the exact same faults. However, in these Houston branches, there were more issues with hydraulic elevators. The oil level faults and even Jack Seal/Packing faults came up more frequently than Denver. This could mean that either Houston had more

hydraulic elevators than Denver or that the hydraulics in Houston were older. Either way, Houston had a larger need for sensors monitoring aspects of hydraulic elevators.

Table 3 – Houston Callbacks: Branch 106850

Subsystem	Fault Count
Service Request	3416
Controller Component	2847
Hoistway Door Gibbs & Interlock	1894
Door Operator	1037
Microprocessor	733
Car Door, Gibbs & Interlock	611
Hall Push Button	501
Selector & Attachments	371
Car Push Button	358
Clutch Assemly / Retiring Cam	345
Brake	272
Photo Cell / Electronic Door Edge	263
Machine / Hoist Motor	262
Leveling	254
Low Oil Level	250
Drive	250
Power Unit Value	247
S. S. Circuit Board (s)	229
Door Linkage / Relating Cables	206
Communication Device	204

Table 4 – Houston Callbacks: Branch 106800

Subsystem	Fault Count
Hoistway Door Gibbs & Interlock	2089
Controller Component	2024
Service Request	1914
Door Operator	840
Car Door, Gibbs & Interlock	674
Hall Push Button	499
Drive	427
Microprocessor	374
Selector & Attachments	321
Clutch Assemly / Retiring Cam	317
Car Push Button	288
Brake	286
Low Oil Level	284
Communication Device	248
Fuse	203

Table 5 – Houston Callbacks: Branch 106860

Subsystem	Fault Count
Controller Component	2899
Hoistway Door Gibbs & Interlock	1885
Service Request	1735
Door Operator	709
Car Door, Gibbs & Interlock	525
Microprocessor	504
Hall Push Button	333
Drive	320
Low Oil Level	268
Selector & Attachments	265
Car Push Button	264
Clutch Assemly / Retiring Cam	255
Jack Seal / Packing	254
Brake	219
Door Linkage / Relating Cables	212
Car Cladding / Accessories	207

3.1.1.3 Phoenix Callbacks

The most frequent callbacks from the Phoenix area are shown in the following table. The following lists show the faults that generated at least 200 callbacks over the three-year period. The Phoenix branch yielded many more fields with the number of call backs over 200. Yet again, the most frequent faults deal with the doors and door operator. The Jack Seal/Packing vastly increased over the previous four branches and the hydraulic subsystem had more than 1000 total faults as well. Therefore, hydraulic sensors will also be beneficial for this branch.

Table 6 – Phoenix Callbacks: Branch 107200

Subsystem	Fault Count
Service Request	6025
Controller Component	4447
Hoistway Door Gibbs & Interlock	3153
Door Operator	1831
Microprocessor	960
Hall Push Button	841
Car Door, Gibbs & Interlock	761
Jack Seal / Packing	639
Selector & Attachments	615
Car Push Button	545
Power Unit Value	517
Photo Cell / Electronic Door Edge	497
Upper Guideshoes (Roller & Slides)	414
Automatic Doors	365
Clutch Assemly / Retiring Cam	345
Door Linkage / Relating Cables	339
Drive	322
Brake	318
Hydraulic Pump / Motor	303
Communication Device	298
S. S. Circuit Board (s)	290
Main Power Supply	266
Low Oil Level	258
Leveling	244
Key Switch	231
Upper Limit	209
Overloads	205

3.1.1.4 Syracuse Callbacks

The most frequent callbacks from the Syracuse area are shown in the following table. The following lists show the faults that generated at least 100 callbacks over the three-year period. This table has a more inclusive minimum for number of call backs due to the low number of total call backs for this branch (e.g. about half the number of blocks as branch 106400). From this table, the same data trends appear. The door and door

operator remain the main reason for fault with the controller taking the top spot. Hydraulic elevator issues also remain among the top faults.

Table 7 – Syracuse Callbacks: Branch 101400

Subsystem	Fault Count
Controller Component	2253
Hoistway Door Gibbs & Interlock	1609
Service Request	953
Door Operator	852
Car Door, Gibbs & Interlock	445
Microprocessor	437
Hall Push Button	359
Jack Seal / Packing	320
Selector & Attachments	280
Car Push Button	246
Photo Cell / Electronic Door Edge	219
Power Unit Value	214
Clutch Assemly / Retiring Cam	201
Machine / Hoist Motor	185
Leveling	172
Communication Device	157
Door Linkage / Relating Cables	156
Upper Limit	149
Low Oil Level	149
Hydraulic Pump / Motor	131
Brake	121
S. S. Circuit Board (s)	113
Upper Guideshoes (Roller & Slides)	102

3.1.2 Summary of Callback Data

After the data was taken from each region, it was combined into a larger dataset intended to be representative of the entire BU Americas. From the top overall faults, a few were chosen based on some external factors and the number of total faults. The total list is shown in Table 8. The potential of the ideas marked in the right most column is based on needs of the branch, recommendations from the tkE Senior Field Engineer, areas already covered under previous projects, and input from conversations with a few field service

technicians. From these inputs, oil temperature, brake wear/stroke, and the level of oil in the pit bucket were chosen for implementation. The brake wear/stroke sensors cover not only brake callbacks, but also assist the technician in routine brake wear measurements. The oil temperature and pit bucket level sensors address problems with the hydraulic subsystem of hydraulic elevators and have the potential to be part of a combined sensor network that measures the health of the entire hydraulic subsystem. The network could monitor faults from multiple aspects of the subsystem including the amount of oil leaking from jack seal/packing and overheating in the oil reservoir.

Table 8 – Total Callbacks with Potential

Event	Count	Potential
Service Request	19776	
Controller Component	17785	
Hoistway Door Gibbs & Interlock	12949	
Door Operator	6434	
Car Door, Gibbs & Interlock	3638	
Microprocessor	3362	
Hall Push Button	3204	
Selector & Attachments	2266	
Car Push Button	2083	
Jack Seal / Packing	1798	x
Clutch Assembly / Retiring Cam	1716	
Drive	1688	
Photo Cell / Electronic Door Edge	1632	
Low Oil Level	1599	x
Power Unit Value	1506	
Brake	1447	x
Door Linkage / Relating Cables	1380	
Communication Device	1379	
S. S. Circuit Board (s)	1232	
Machine / Hoist Motor	1174	
Upper Guideshoes (Roller & Slides)	1163	
Leveling	1152	x
Main Power Supply	1149	
Hydraulic Pump / Motor	1128	x
Automatic Doors	1013	

The door operator, while easily the most frequent cause of callbacks, is already covered under the Maintenance on Demand project and the ongoing project with the University of Northampton. Other callbacks such as the Hall Push Button were not chosen because a broken button is mainly cosmetic and easy to fix.

3.2 System Requirements

For each of the scenarios determined in the previous section, a sensor needed to be selected. By definition of this study, the sensors needed to be low cost while still performing the required functions. The system initially also needed a data acquisition method to collect, store, and process data from the sensors. The requirements for these topics were straight forward but still required some analysis to choose appropriate solutions.

3.2.1 Sensors

3.2.1.1 Brake Gap

An increasing number of buildings are being installed with machine-roomless (MRL) elevators which feature the machine suspended above the top floor in the elevator shaft itself. Any maintenance on the machine has to be done from the top of the elevator cabin and can require safety harnesses and in some cases even a safety platform that must be built or installed every time maintenance is performed. The brake subsystem is a part of this machine and requires regular maintenance visits, some as often as every 3 months. Due to the frequency of the visits and the egregious amount of time required to perform maintenance on the machine of an MRL elevator, a remote solution would be extremely beneficial in both cost and time savings.

The brake sensor needed to be the most accurate of any of the chosen scenarios. This measurement scenario applies to two different machines that are both common in thyssenkrupp Elevators. The problem here was the measurement of how much a brake has worn over time and whether or not it needed to be adjusted. There were two measurement types for each machine. The first measurement was of the brake stroke or how far the brake needs to move to engage. Taken from an internal thyssenkrupp maintenance manual, the GTW3M machine needed to be measured at the points shown in Figure 4 with the following instruction: "Check the gap "A" (brake stroke) at point 1 and 2, which are marked on the brake as well, with feeler gauge as the following picture (Figure 4) shows. Max gap "A" after wear should be <0.4mm for PZD140MA1, <0.45mm for PZD140MB1, otherwise the gap must be readjusted."

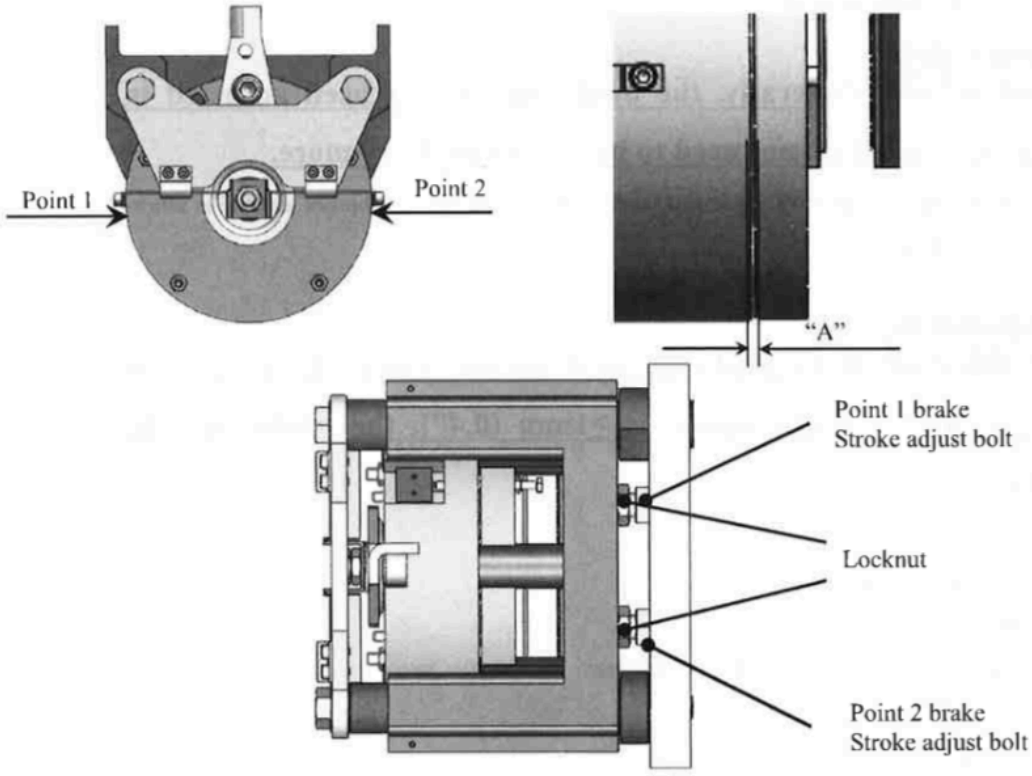


Figure 4: Brake Stroke Measurement Points – GTW3M

The second measurement was the overall brake wear. Also for the GTW3M machine, and also taken from the thyssenkrupp internal maintenance manual, the measurement points are shown in Figure 5 and have the following instruction: “The brake lining wear must be <1mm. Use feeler gauges to check gap “a1”, “a2”, and “b” between brake disc and brake lining frame. The feeler must be able to fit in all 3 gaps.”

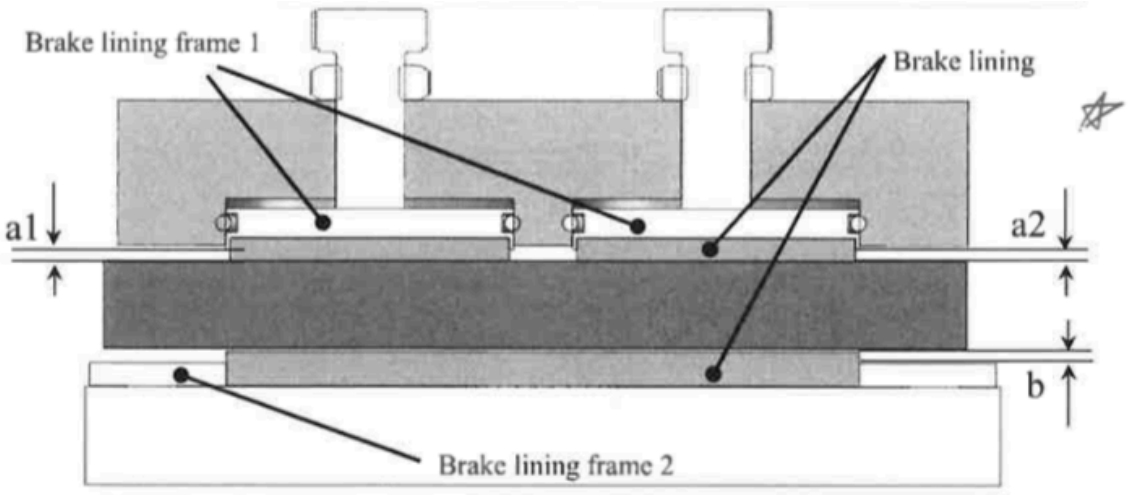


Figure 5: Brake Wear Measurement Points – GTW3M

The measurements for the GTW8M machine were similar as shown in Figure 6 and Figure 7. This machine also had instructions for the required amount of gap for each measurement type. For the brake stroke: “DO NOT adjust gap “A” (brake stroke) at point 1 and 2, point 3 and 4 unless gap “A” is 0.42mm or bigger. Use feeler gauge 0.42mm as test gauge that can NOT be inserted into gap “A” at point 1-4. If gap “A” is smaller than 0.42mm, there is no need to adjust the gap (brake stroke).”

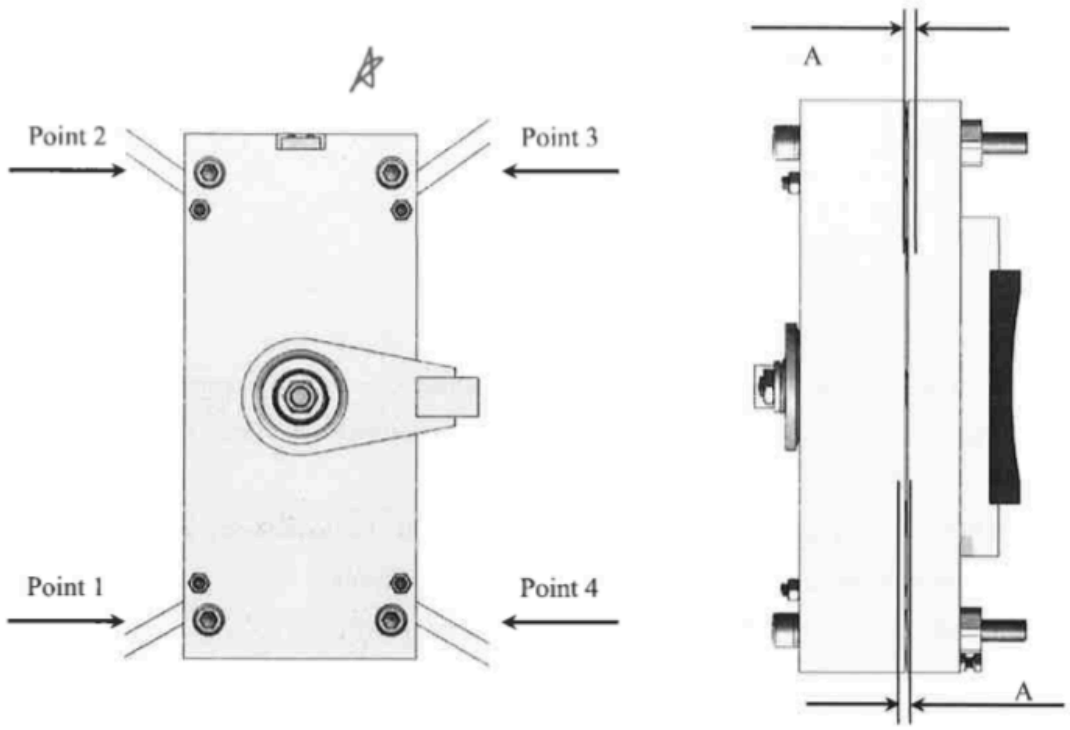


Figure 6: Brake Stroke Measurement Points – GTW8M

The GTW8M manual also has a similar criteria for the brake wear as the GTW3M: "The brake lining wear out must be <2mm. Use beam calipers (sic) to check the distance "B" (eye able guide bushing) between the anchor disk and the machine frame at the side as shown (in Figure 7), it must be >8mm. Generally the distance "B" is about 10mm and it has been adjusted in the factory."

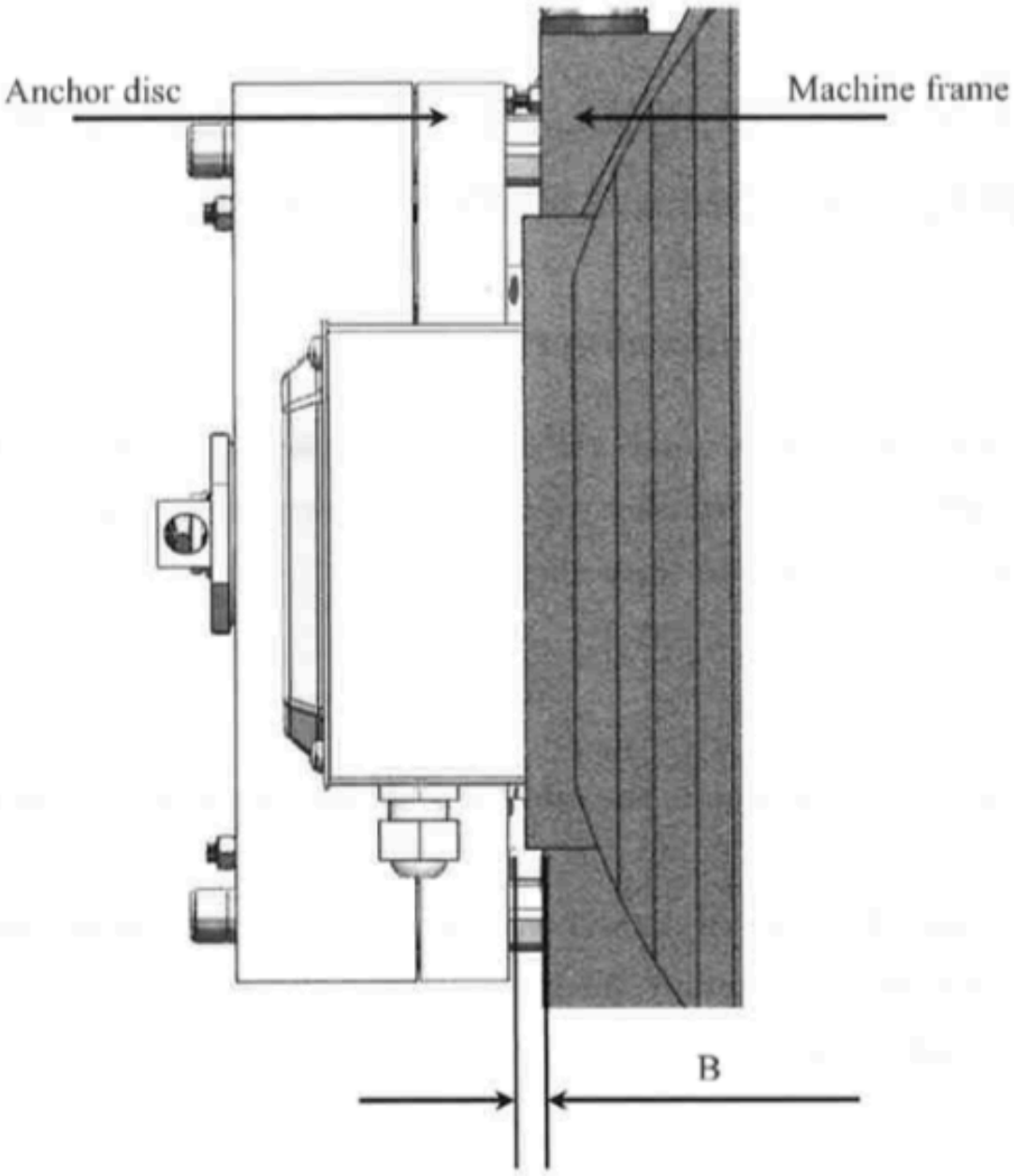


Figure 7: Brake Wear Measurement Points – GTW8M

While similar, the different testing scenarios were actually quite varied. In order to use one sensor, that sensor needed to measure 0-10mm at a minimum and have a resolution high enough to discern a minimum difference of 0.05mm. Using this information, a number of distance sensors were found that matched the criteria. The sensor that best fit the scenario

was the TT Electronics' Model 404 Series position sensor seen in Figure 8 and in appendix A.1 TT Electronics Linear Position Sensor. It had an electrical travel range of 12.7mm and a resolution that is theoretically infinite and would vary depending on the data acquisition method. It also featured a spring loaded shaft meaning that a return mechanism was not required. The interface to data acquisition equipment was simple as well, allowing it to fit with many possibilities. It has a single purchase price of 21.83 USD and as it fit under all the sensing requirements, it can be purchased in bulk further reducing the cost. The one downside of this sensor is that while it can be retrofitted and would not need a redesign of the existing machine, it would require an adaptation to effectively measure the brake parameters.

Figure 8: TT Electronics Linear Position Sensor

3.2.1.2 Oil Level: Pit Bucket

For the oil pit bucket scenario, a highly accurate sensor was not necessary. Measuring the oil level in this specific case was limited to the amount that leaks from an elevator and collects in a bucket in the pit underneath the cabin, but could also be applied to the oil level in the hydraulic tank itself. This aptly named “pit bucket” is a rudimentary fix to a larger problem in the Jack seals and packings, but has associated problems that could be addressed now with monitoring while a more comprehensive solution is developed.

Figure 9: Oil collection bucket in the pit

The bucket collects oil leaked out of the seals and packings of a hydraulic elevator. When excessive oil leaks out, the elevator experiences cavitation, or excess vibrations, which can be unpleasant to passengers. The bucket is also prone to overflows which can be dangerous to technicians and potentially harmful to the environment in flood prone regions. It is therefore vital to monitor the amount of fluid in the bucket.

The area and bucket mentioned are shown in Figure 9. As oil leaks from the hydraulic jack, it gets collected in the white bucket. The bucket is normally a used 5 gallon bucket that the hydraulic fluid is shipped in, but there are slight variations. Size can vary from site to site and they do not always have a lid. Therefore, a slightly robust sensing method was required. The height of a 5 gallon bucket can vary. Due to this fact, a uniform sensor would perform better in any bucket if it measures from the top of the bucket. In some buckets, this leaves a gap between the bottom of the sensor and the bottom of the bucket, but the goal is to prevent overflows and the bottom portion of the bucket is not as important.

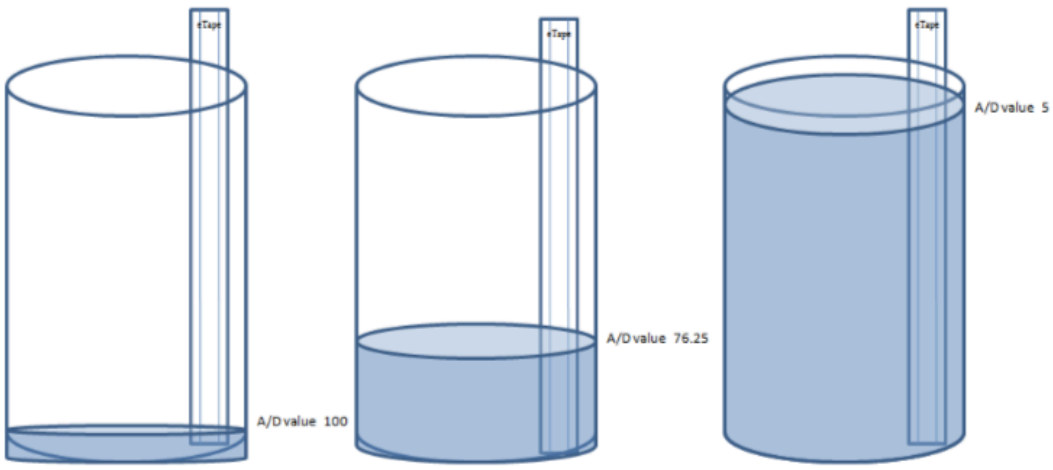


Figure 10: eTape Continuous Fluid Level Sensor

The eTape fluid level sensor from Milone Technologies works in a few different setups. The sensor shown in Figure 10, and also seen in the spec sheet in appendix A.2 Milone Technologies eTape Fluid Sensor, is set up as a voltage divider. The sensor comes with an internal reference resistance that is connected to 5v from the controller. This value is read by the controller and compared to the variable resistive value of the sensor itself as it changes with the fluid level. The controller then calculates the fluid level from the voltage difference. The sensor has a high resolution of 1/32 inch and, with a sensor length of 14.3 inches, is long enough to measure the most common 5 gallon buckets which can be 15-16 inches tall. A single sensor costs 40.00 USD and the cost can be further reduced through a bulk purchase.

3.2.1.3 Oil Temperature: Pump Reservoir

The Pump Reservoir for hydraulic elevators is typically located in a pump house or room next to the elevator shaft as seen in Figure 11. As the system begins to wear and the fluid leaks or breaks down, the reservoir can overheat to levels that prevent a technician from accessing the room. However, the system might only overheat during peak travel hours (e.g. for an office building 8am, 12pm, or 5pm) and appear otherwise healthy when a technician is present. Both of these reasons contribute to a need to have a sensor constantly or regularly monitoring the temperature that can also be accessed remotely in the case that the technician cannot enter the room.

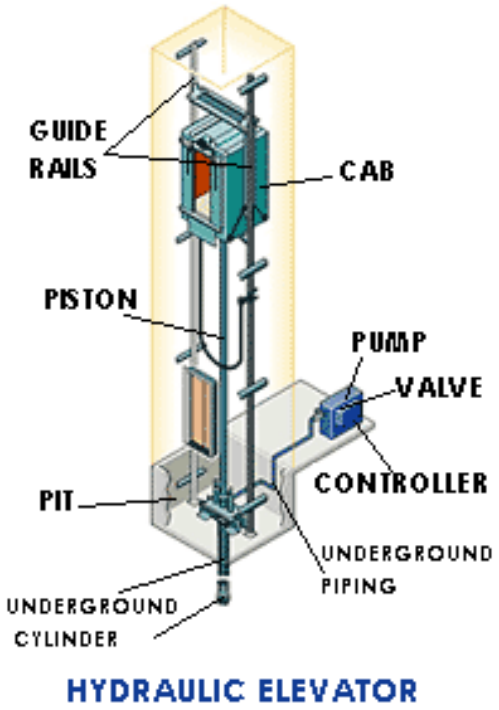


Figure 11: General Hydraulic Elevator System [24]

From the technician call-back logs discussed in section 3.1.1, the temperature of the pump is typically below 110 degrees Fahrenheit, and can overheat to temperatures above 140. With a range of temperatures this large (~80-150 F), the sensor did not need to have a large resolution or sample rate. The main requirement was that it be submersible in hydraulic fluid at these temperatures. From those requirements, the DS18B20 was chosen.

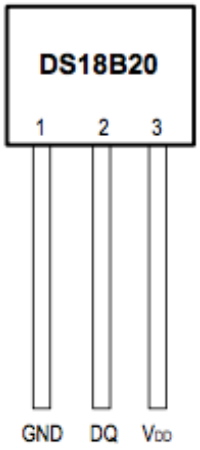


Figure 12: DS18B20 Temperature Sensor

The DS18B20 can be purchased as the stand-alone as in Figure 12 or as a submersible version. The sensor functions in a temperature range of -67 to 257F with a precision of half a degree. It can operate in a 1-wire mode or with all 3 wires depending on the needs of the system. With minimal sensors connected to one DQ line, similar to an I2C protocol, it is possible to operate in 1-wire mode. As more sensors are added, more power will be needed and it would be beneficial if not required to connect the GND and Voo pins to an external power source. The instructions on both methods are shown in the spec sheet attached in appendix A.3 DS18B20 Temperature Sensor Also, one of the submersible sensors can be purchased at 9.95 USD and cheaper if purchased in bulk. It would be relatively inexpensive to include multiple of these on one system if necessary.

3.2.1.4 MULTI – Accelerometers and Gyroscopes

The new elevator currently in development at thyssenkrupp is called the MULTI due to the ability for the cabin to move horizontally as well as vertically. As the elevator is still in development, much work still needs to be done to get it to market. One of the projects done on the elevator addresses vibrations that affect passenger comfort. Vibration values need to be below certain thresholds, varying depending on vibration direction, according to VDI-2057 and EN ISO 8041:2005. To monitor these values, four inertial measurement units (IMUs) were attached to four points of the current MULTI prototype as seen in Figure 13. The sensors S_u and C_u are both IMUs from iMAR shown in Figure 14 and the sensors $B_{u,i}$ are both of the type shown in Figure 15. These sensors are separate from the communication network and other sensors described in this section, but could be added to the Dempster-Shafer algorithm described in sections 4.1.2 and 4.2.1 to record the data for vibration and frequency. While these sensors record gyroscopic and acceleration

data, only the acceleration data was used in the study. There is potential use for the gyroscopic data in future projects.

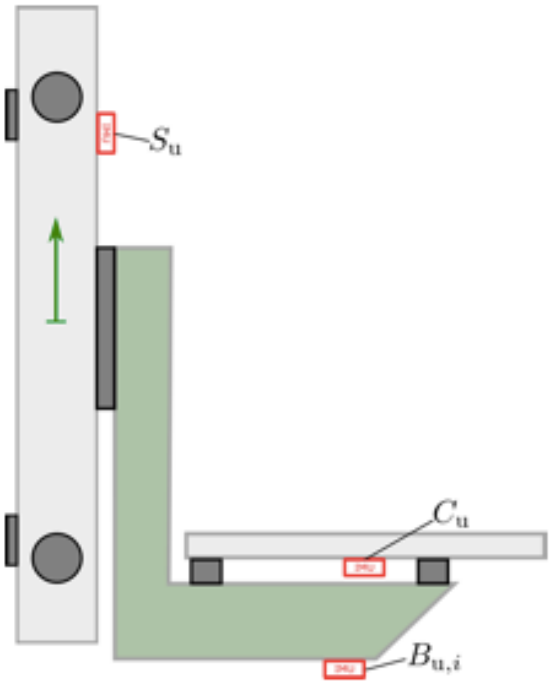


Figure 13: IMU placement on MULTI prototype

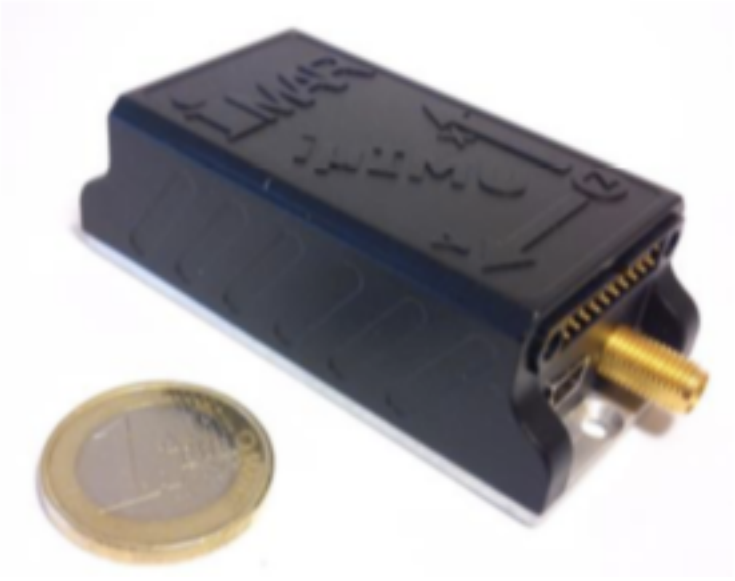


Figure 14: iMAR IMU (200 Hz)

Figure 15: IMUs NGS2 from Sensor-Technik Wiedemann GmbH

3.2.2 Communication Network

The basis for a remote communication network stemmed from the need of a field technician for a tool to measure aspects of the elevator. The primary motivation was to develop an expanded network based on the Elevator Health Check application which features acceleration sensors transmitting data to an iPhone app. Due to the distributed nature of the application scenarios described in the previous sections, there was also a need for a wireless network between the sensors before the data is sent to an app. If the results are implemented in the future, it will likely be integrated into the MAX 2.0 project now that the MAX team has started allowing full access to the associated data recording and transmitting hardware.

The three scenarios determined from the call-back logs in section 3.1.1 are low bandwidth and therefore do not need to be transmitted via a wired connection or Wi-Fi. Bluetooth has a high enough bandwidth to transfer recordings that are made at a maximum of once every minute. Because these sensor scenarios are distributed and additional scenarios would be further distributed, a central collection point was used that would be able to access all of the sensors, while still allowing access to technicians. Because the elevator cab is already moving through the shaft, and the technician has to access it during maintenance, the elevator cab was the logical collection point. As the elevator moves through the shaft and comes into range of a sensor, it collects any recorded data and stores it for future transmission to the technician. The data would be transmitted via an HC-05 Bluetooth chip and each sensor would have its own Arduino Nano micro-controller for short term storage and for sending the data. However, this setup can always be amended if high-bandwidth sensors are required or a better collection point is viable. An overview of this setup is shown in Figure 16.

In Figure 16, sensors 1, 2, and 3 are currently connected to the Raspberry Pi on top of the elevator cabin and the Pi is also connected to the technicians iPhone. Sensors 4 and 5 are out of range of the Pi and are thus shown with a broken or dotted connection. Sensor 4 is expanded to show the sensor sending data to the Nano where it waits to be transmitted via the HC-05 Bluetooth module. It should be noted that while these 5 sensors each have their own Nano and HC-05, it is possible and likely necessary to connect multiple sensors to one Nano and HC-05. However, the Nano and HC-05 together only cost 11.99 USD, and cheaper in bulk, so it would not be too expensive for each Nano to connect to only a single sensor.

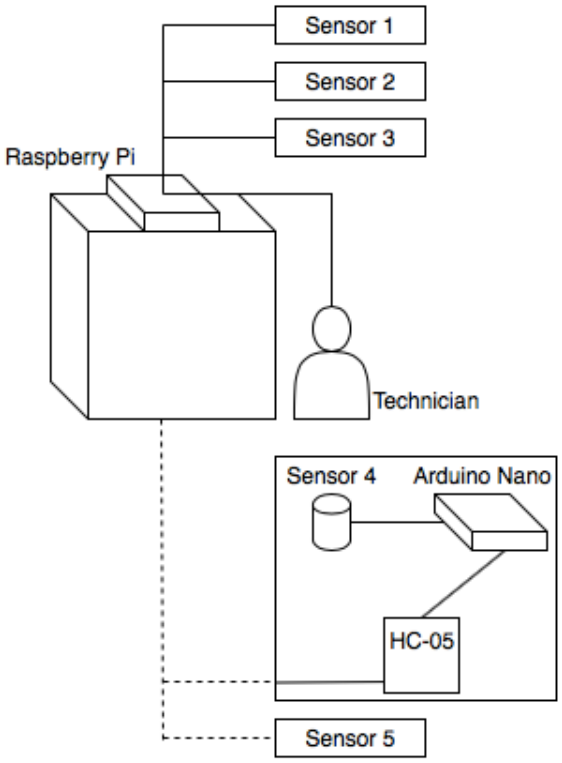


Figure 16: Sample System Network Diagram

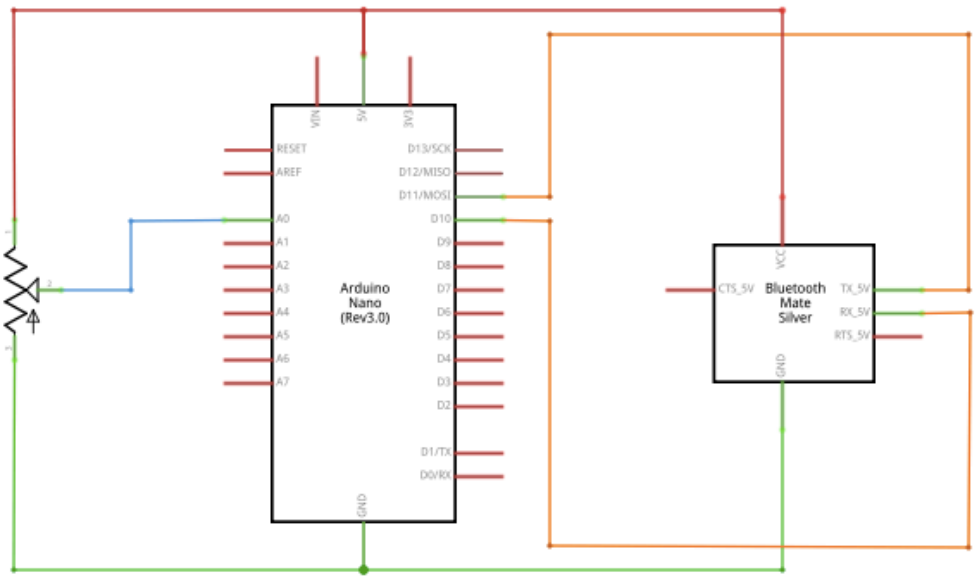


Figure 17: Linear Sensor Circuit Diagram

The circuit diagrams for each sensor setup, with the Arduino Nano microcontroller and HC-05 Bluetooth module, are shown in Figure 17, Figure 18, and Figure 19. The TT Electronics linear sensor diagrammed in Figure 17 is a spring loaded linear potentiometer, similar to a slide potentiometer. The resistance value between ground and +5v is read as a voltage into pin A0, converted to a distance, and then transmitted through the HC-05 Bluetooth module. The sensor here was stated to have a theoretically infinite resolution and is limited by the amount of bits given by the Arduino to the measurement. With the minimum 0-1023 bits, the sensor had a resolution of approximately 0.02 mm which is less than the 0.05 mm required by the maintenance manual.

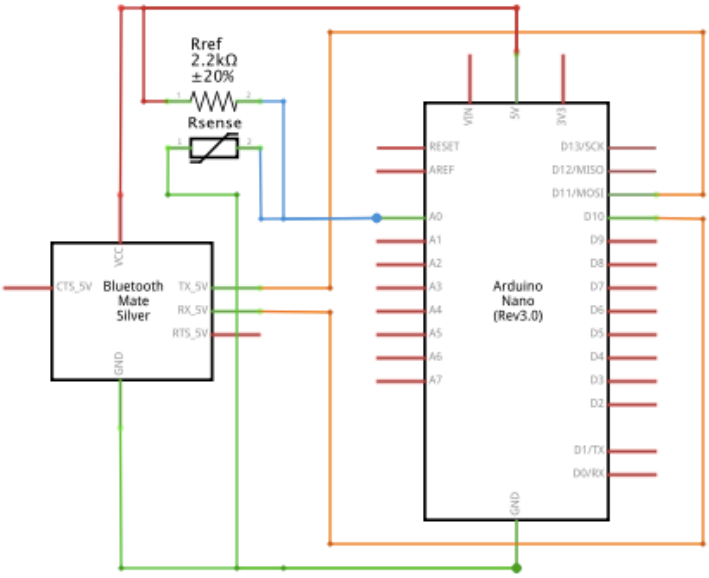


Figure 18: Fluid Level Sensor Circuit Diagram

The Milone eTape sensor in Figure 18, when purchased as the 4-pin option, has an internal reference resistor of about 2k Ohms and a variable resistor that are used to measure the fluid height. Both resistors connect to the node at A0 and as the fluid level increases, the voltage output from the voltage divider will change accordingly. The HC-05 setup is the same as the linear sensor. The DS18B20 temperature sensor when in the 3-wire

configuration appears as in Figure 19. If in 1-wire configuration, only the DQ pin would be connected to the D2 pin on the Arduino Nano. The HC-05 setup is the same as the other circuit diagrams.

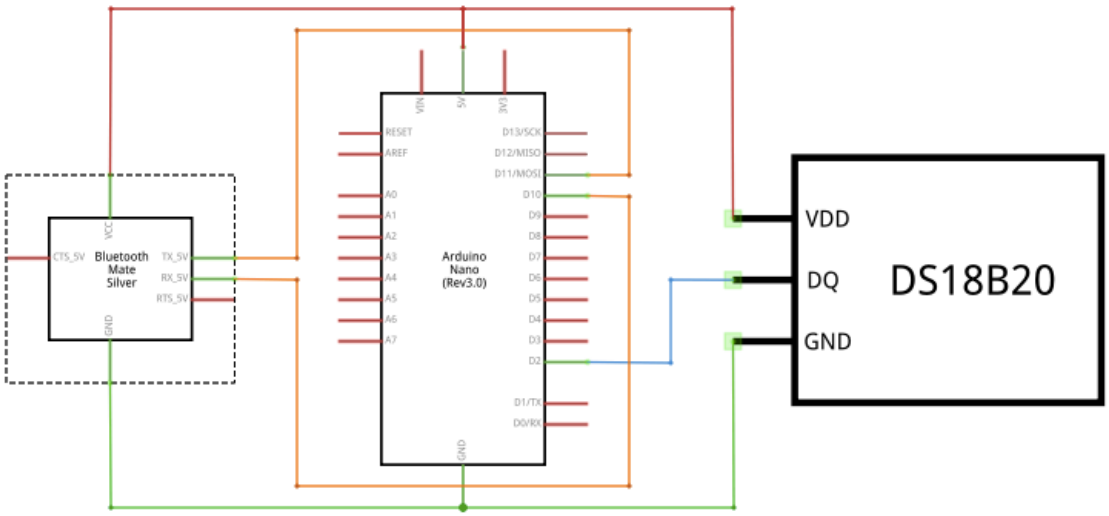


Figure 19: Temperature Sensor Circuit Diagram

In order to show the feasibility of this system, sample sensor data was collected and transmitted to a Raspberry Pi where it was stored until the iPhone mobile phone application (app) made a connection. The test showed that the Nano would store data until in range with the Raspberry Pi. Once a sensor was in range of the Pi, the Nano would transmit the data via the HC-05 module and wipe the recorded values. The Pi would then store the data with a timestamp until the iPhone app made a request. Once the request was made, the Pi burst transmitted the data to the phone. The app did not have any data processing coded into it and only displayed the transmitted values in the spaces provided. The processing in the app would come from one or some combination of the sensor fusion methods discussed in chapter 4. The main storyboard for the app is shown in Figure 20. The app also displayed

whether or not the phone had activated Bluetooth and the name of a Bluetooth device when one connected. It is important to note that this sample setup was purely a proof of concept and not intended to be a final iteration. For example, this sample had no security protocol or high detail data processing. It was developed only to show that the idea was feasible and could be expanded in the future. Now that access to the MAX data acquisition and transmission hardware is available, the network is less necessary but could still have some application on the elevators not yet covered under a MAX system.

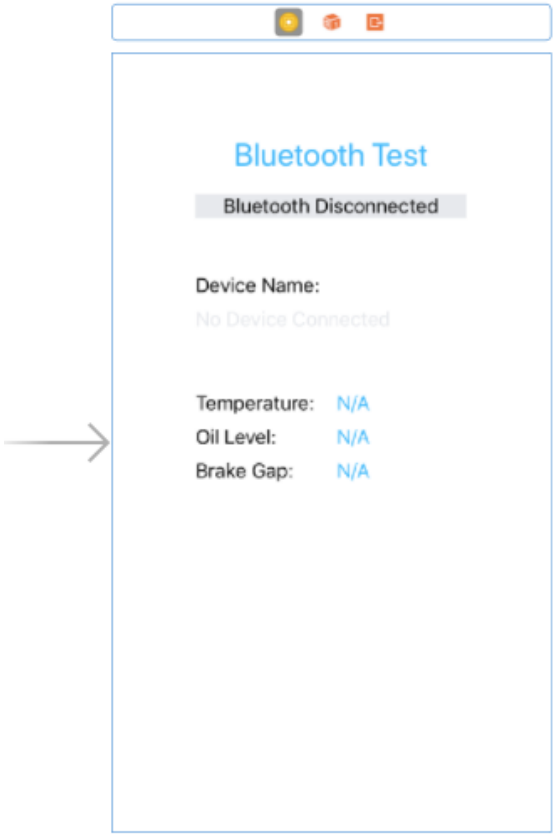


Figure 20: Proof-of-Concept Bluetooth App

3.3 Summary

The sensors and communication network described in section 3.2 have the potential to meet multiple aspects needed in a network for the scenarios chosen in section 3.1. The sensor specs are more than adequate to measure in the range and resolution required for each subsystem and the Arduino, Raspberry Pi, and iPhone network connecting via Bluetooth has the storage and bandwidth necessary to do local short term storage on the Arduino, longer storage on the Raspberry Pi, and transmit all data between network nodes. Without access to the actual spaces that the sensors and network would be installed, minimal testing was done. However, for an installation, some pieces will need to be determined individually for each elevator. Some shafts might require Bluetooth relays or signal boosters, while some might need more short term storage, and others might need different sensors entirely. The present system is a starting point for individual installations and provides the framework for any future networks.

CHAPTER 4. SENSOR FUSION APPLICATIONS

4.1 Sensor Fusion Methods

There are multiple sensor fusion methods, all of which have different benefits and use-cases. However, each method has the same goal of combining data and generating a result that is more accurate or intelligent than the data could provide separately. These methods include: Bayes Probabilities, Dempster-Shafer theory of evidence, Kalman-Filtering, and Artificial Neural Networks. All of these methods are discussed below with some level of detail, and detailed examples showing the benefit of using the Dempster-Shafer and the Kalman Filter methods are also discussed. This paper discusses only these four methods, and implements two of them, because while there are more algorithms available, they might only be slight variations of the ones above or have niche utility.

4.1.1 Bayes' Probability

Many sensor fusion methods stem from probability theory, so the first method examined in this paper is Bayes' Theorem. This concept of probability was first proposed by Thomas Bayes in the 18th century [25]. While the first use of probability was clearly not sensor fusion, the method is still relevant. The overarching idea is that a user can input various possibilities and determine the probability of an event occurring. The concept is extremely similar for sensor fusion. A user inputs various data points and receives a combined datum that ideally has more knowledge or worth than the individual data points. From this concept, the Bayes' Theorem, seen in equation 1, was applied to sensor fusion.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} \quad (1)$$

Relating Bayes' Theorem to the topic of this study, a user could predict the probability of Fault A occurring given that sensor data B is recorded. The method is simple, but it could still be applicable in some basic situations. There are multiple variations of this method that arose as an attempt to expand or improve the effectiveness of the theorem. For example, a Bayesian Network is one such expansion which is utilized in Ref. [26] to establish an occupancy grid for the exterior of a mobile robot.

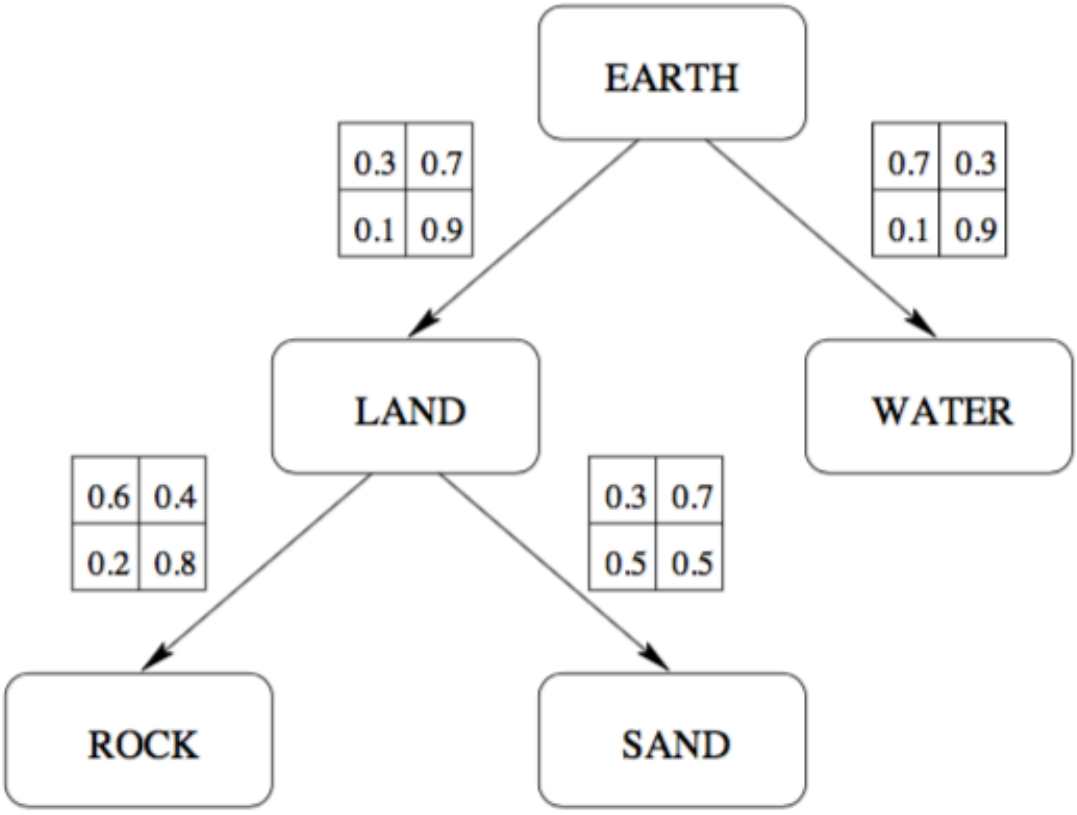


Figure 21: Sample Bayesian Network [26]

The network has pre-determined probabilities of certain scenarios occurring based on other scenarios occurring and can determine the appropriate result from a given input. The robot in this case uses a network of this type and sensor data to make guesses about its environment as it moves.

The major downside to this method, and to Bayes' Theorem in general, is the large amount of prior data that is needed to generate a probability network like the one shown in Figure 21, or just general *P(B|A)* probabilities. The relations between each state need to be known in advance in order for the algorithm to propagate up the network. This would mean that the robot already has a vast knowledge of its potential surroundings. For the context of this paper, knowledge would be required for all the different fault types and how they relate to the sensors attached to the system. There are certainly benefits, but for this study there are better alternatives.

4.1.2 Dempster-Shafer Theory of Evidence

Arthur Dempster, and later Glenn Shafer, expanded and altered the original Bayes' Theorem as many others have also done. Instead of using statistical probability functions, Dempster uses functions, that are referred to as "Belief Functions", "Mass Functions", or "Levels of Evidence", which are simply the levels of belief in a piece of evidence [21].

$$\Omega(\theta) = \{a, b, [a, b], \emptyset\} \tag{2}$$

Given a set of outcomes, there is a power set, 2^n, of mass functions for n potential outcomes as seen in equation 2. Where in this case, $n = 2$, and the possibilities are *a*, *b, a or b*, or the null set. Each possibility is then assigned a belief, *m*, such that

$$\sum_{A \subseteq \Omega} m(A) = 1 \tag{3}$$

From these mass functions, each outcome has an associated belief and plausibility function.

$$bel(A) = \sum_{\beta \subseteq A} m(\beta) \tag{4}$$

$$pls(A) = \sum_{\beta \cap A} m(\beta) \tag{5}$$

This "Belief Interval", [*bel(A), pls(A)*], is used to make the decision of which outcome is present. There a few ways to make the decision. For example, the highest belief, the highest plausibility, the smallest belief interval, or some combination thereof. The choice of which decision metric to use can come from the user or from whichever gives the most accurate results.

The main benefit for sensor fusion in the Dempster-Shafer method is the combination of evidence. Once different outcomes have been assigned a mass function, they can be combined using the following equation.

$$m(A) = \frac{1}{1-K} \sum_{B \cap C = A} m_1(B) m_2(C) \tag{6}$$

where $$K = \sum_{B \cap C = \emptyset} m_1(B) m_2(C)$$

The *K* value is a measure of conflict between the two data sources and used to eliminate discrepancies. This combination method generally allows for more accurate belief and plausibility values and a reduced Belief Interval. For example, suppose that an elevator has two temperature sensors, 3 accelerometers, and 2 noise sensors. Each sensor records a value that is used to generate a mass function for each fault type. All of these readings can

then be combined using equation 6 to get an overall view of the system and make a more intelligent assessment of which fault is occurring.

Dempster-Shafer is more robust than the Bayes' Theorem but also has its own shortcomings. There is no widely recognized method of determining the mass functions and can range from "Expert Input", such as a meteorologist predicting the chance that it will rain or be sunny, to using prior knowledge and calculating the distance between the sensor data and different states. Dempster-Shafer also has problems with strongly conflicting data as seen in Ref. [26]. Exaggerated sample data exemplifying this conflict is multiplied in Table 9 and then combined using equation 6 in Table 10.

Table 9 – Conflicting Evidence

		$m_1(A)$	$m_1(B)$	$m_1(C)$
	m(Ω)	0.01	0.0	0.99
$m_2(A)$	0.01	0.0001	0.0	0.0099
$m_2(B)$	0.99	0.0099	0.0	0.9801
$m_2(C)$	0.0	0.0	0.0	0.0

Table 10 – Conflicting Result from Evidence in Table 9

	Dempster-Shafer
m(A)	1.0
m(B)	0.0
m(C)	0.0

The combined values strongly favour option A even though both data sources had very small mass functions for A. Because the two data sources were strongly in favour of conflicting sources, the algorithm eliminates the conflict and chooses the decision where there is any overlap, however small it may be. This problem can generally be avoided by combining a higher number of data points, but it could still arise.

Despite this shortcoming, the Dempster-Shafer method is a valid sensor fusion algorithm and can generally deal with disagreements between data sources. Multiple data sources can be shown to be sufficient to deal with wide variations in the data itself. For a scenario that does not require time-varying analysis and is limited to low amounts of computing power, Dempster-Shafer is a relevant method and was strongly considered as a solution for this study.

4.1.3 Kalman Filter

A Kalman Filter differed from the previous two methods in that the algorithm specifically includes time as a factor. This method treats the system similar to a controls problem and models the system as a discrete-time state-space representation [20]. Using this state space representation, the algorithm makes a prediction and estimate of the current state given information of the current and previous states.

$$x(t) = A(t)x(t-1) + B(t)u(t) + v(t) \tag{7}$$

$$y(t) = C(t)x(t) + w(t) \tag{8}$$

Equations 7 and 8 are used to discretely model the system. Where $x(t)$ is the state vector at time t, $u(t)$ is the input at time t, and $v(t)$ and $w(t)$ are Gaussian noise with covariance of $R(t)$ and $Q(t)$ respectively [20]. The time t can also be replaced with $k\Delta t$, where k is an integer for the current iteration, and Δt is the step size of the system. From this model, the Kalman Filter algorithm estimates the current state of the system using the following equations from Ref. [20].

Prediction:
$$\hat{x}(t|t-1) = A(t)\hat{x}(t-1|t-1) + B(t)u(t) + v(t) \quad (9)$$
$$P(t|t-1) = A(t)P(t-1|t-1)A^T(t) + Q(t) \quad (10)$$

Estimation:
$$K(t) = P(t|t-1)C^T(t)[C(t)P(t|t-1)C^T(t) + R(t)]^{-1} \quad (11)$$
$$\hat{x}(t|t) = \hat{x}(t|t-1) + K(t)[y(t) - C(t)\hat{x}(t|t-1)] \quad (12)$$
$$P(t|t) = [I - K(t)C(t)]P(t|t-1) \quad (13)$$

The main deviations of the Kalman filter from other sensor fusion methods, are the calculation of the Kalman gain *K(t)* as shown in equation 11, and the recursive nature of the estimation. The algorithm calculates the error between the predicted values in equation 9 and the measured values *y(t)*. The Kalman gain, which is a combination of the covariance matrices of the estimation and the inherent noises, acts as a step size adjustment to better predict the state of the system. The estimation covariance is then updated in equation 13 and all of the values are fed into the following time step of the algorithm. There are many variations of the Kalman Filter, for example the Extended Kalman Filter, that each have uses in different areas.

Like Bayes' Theorem and Dempster-Shafer, the Kalman Filter method also has many variations all of which have usefulness in some area. In general, the Kalman Filter works well with time varying systems unlike Bayes' Theorem and Dempster-Shafer. Some downsides to the Kalman Filter include high computational cost especially as the number of sensors increase and the covariation matrices become larger and more complex. Despite this, it has use in scenarios where the state of the system needs to be estimated or cannot be measured.

4.1.4 Artificial Neural Network

Artificial Neural Networks (ANN) are probably the most well-known of the algorithms discussed here and could be considered as having the broadest application. The concept is derived from the functionality of neurons in a brain hence the name, "Neural Network". The algorithm structure is comprised of multiple layers as seen in Figure 22. Only two layers, the two outer layers, are technically required and the middle "hidden layers" are hidden neurons connecting the inner and outer layers. They are called hidden only because the values calculated in each node are never seen by the end user. The user puts data into the input layer and receives and output in the output layer. The number of these hidden layers and number of nodes per layer is not pre-defined and could be chosen by the user or by another algorithm designed to optimize the network for a specific application. The number of input nodes typically comes from the type of data (i.e. number of data points available) and the nodes in the output layer depend on the type of outcome required. For example, the output could be akin to binary where each node will be a 1 or 0 and will signify a specific outcome. Each path between layers, or between nodes, has an associated weight that is tuned as the algorithm is trained. The training method is generally the most complex step in creating a neural network. The method of training can be back or forward propagation or involve higher level deep-learning methods.

Once the ANN is trained, a user or a script inputs data values into the first layer. The ANN will propagate these values through each layer and eventually make an overall decision or assessment. For example, in Ref. [23] and as seen in Figure 22, researchers used sensors to determine the wear of a cutting tool. They used a number of features in the first layer and the last layer output the predicted average wear of the cutting tool. For this

paper, the sensor data would be fed into the first layer and the ANN would decide what the possible fault state is.

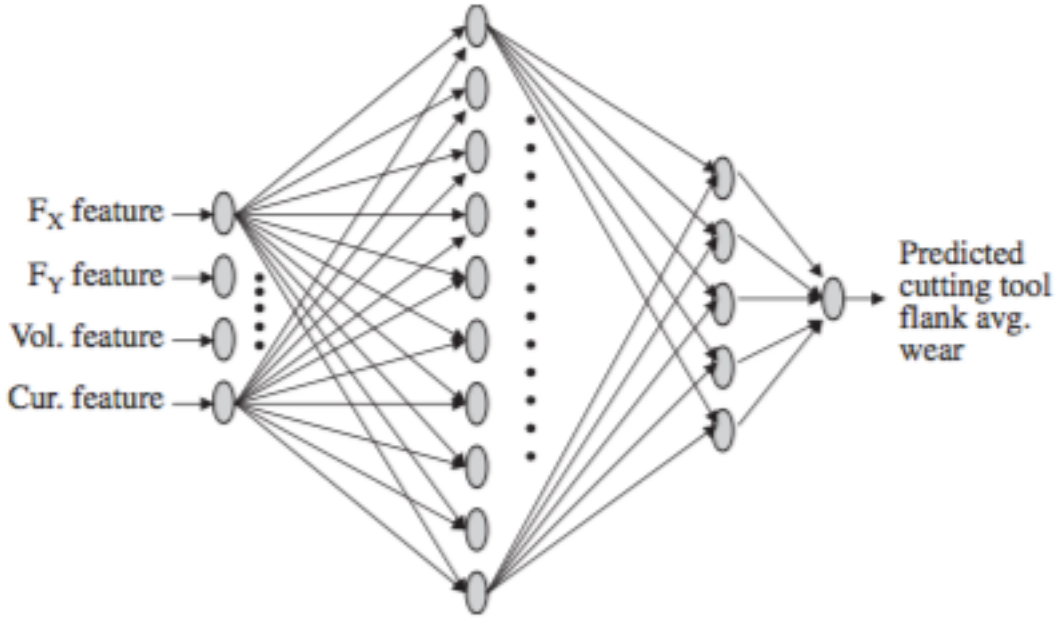


Figure 22: Sample ANN [23]

Neural Networks can be extremely accurate but can also require a lot of data to train the weights. This training requirement, as with the other methods, means that there is a need for prior knowledge of the system before the algorithm is useful. There is also a large amount of variety in the structure of an ANN and this can be useful or a hindrance to the user. For example, they are trained either through a feed-forward or feed-back method, the number of layers and number of nodes per layer is not predetermined, and how to weight the paths is also decided by the user. However, the method has been shown to have clear benefits in accuracy and the broad applications to which it can be applied.

4.2 Algorithm Results

The Dempster-Shafer and Kalman Filter methods were tested to determine the efficacy when used for a small number of sensors. Each algorithm can of course be expanded to include any number of sensors, but for exemplary purposes, and to keep calculations to a minimum, 5 data types were used for the Dempster-Shafer, and two sensors were used for the Kalman Filter. The Bayes' Theorem was not tested because it has been eclipsed by the Dempster-Shafer and other methods and the ANN was not tested because the Dempster-Shafer and Kalman Filter methods fit the scenarios better.

4.2.1 Dempster-Shafer Theory of Evidence

The Dempster-Shafer algorithm was tested in a maintenance scenario. Multiple sensors collect data across subsystems which is then used to estimate if and which fault occurred. Unfortunately, due to a lack of recorded data, the data had to be simulated with a Monte-Carlo Simulation. This lack of available data for the chosen sensor types and fault types required mock data to be generated. The types of faults come from the maintenance log data shown previously and could be measured with the chosen sensors. Mock fault states are shown in Table 11 and the mock standard deviations for each fault state are shown in Table 12. These values were used along with the MATLAB *normrnd* function to generate 10000 samples for each fault according to a Gaussian distribution.

Table 11 – Means of Mock Fault Data

	Temp (F)	**Oil Level (in)**	**Vibration (mG)**	**Frequency (Hz)**	**Noise (dB)**
No Error	100	3	10	10	30
High Temp	130	3	15	25	50
Low Oil	100	9	25	25	55
Rail Guide	100	3	20	40	55
Brake Pick	100	3	15	10	70

Table 12 – Standard Deviations of Mock Fault Data

	Temp (F)	Oil Level (in)	Vibration (mG)	Frequency (Hz)	Noise (dB)
No Error	18.1	3.5	4.7	4.8	12.5
High Temp	19.0	3.8	5.7	5.8	14.7
Low Oil	12.2	3.5	4.7	6.5	17.5
Rail Guide	18.3	3.3	6.7	5.1	13.2
Brake Pick	17.3	4.3	4.8	4.7	12.6

Each sample was used to calculate a mass function to plug into the algorithm. The mass functions, as mentioned before, do not have a widely accepted method of determination. For this study, and similar to Ref. [28], the mass functions were calculated using the Minkowski distance, shown in equation 14, between the sample, x_i, and the means, $\bar{\bar{x}}$, of each of the faults. For this specific case where the exponents in equation 14 are $\rho = 2$, and subsequently $1/\rho$, the equation is also known as the Euclidean distance.

$$d(x_i, \bar{\bar{x}}) = \left(\sum_{i=1}^{n} |x_i - \bar{\bar{x}}|^2 \right)^{1/2} \tag{14}$$

$$p_i = {}^{1}/_{d(x_i, \bar{\bar{x}})} \tag{15}$$

$$m_i = \frac{p_i}{\sum_{i=1}^{n} p_i} \tag{16}$$

Each distance calculated using equation 14 was then inverted as seen in equation 15 and normalized to be between one and zero using equation 16. An example of the mass function generation is shown in Table 13.

Table 13 – Example Mass Function Generation

	Temp (F)	Oil Level (in)	Vibration (mG)	Frequency (Hz)	Noise (dB)
High Temp	130	3	15	25	50
Sensor Data	118	2	17	23	42

$$d(x_2, \bar{\bar{x}}) = \left(|118 - 130|^2 + \cdots + |42 - 50|^2\right)^{1/2} = 14.7309$$

$$p_2 = {}^{1}/_{d(x_1, \bar{\bar{x}})} = 0.0679$$

$$m_2 = \frac{p_2}{\sum_{i=1}^{n} p_i} = 0.3226$$

Using the means of each fault state and the data from the sensors, the mass function for a High Temperature fault in this example was calculated to be 0.3226.

Each of these 10000 samples for each fault was run through the Dempster-Shafer algorithm to ensure that the predicted fault matches the fault from which the data was generated. The accuracy was then calculated as a ratio of successes out of the 10000 samples. The initial accuracies for varying numbers of each sensor type are shown in Table 14. For example, 1 sensors means one of each of the 5 sensor types, and 2 sensors means 2 of each of the 5 sensor types.

Table 14 – Accuracies from Dempster-Shafer (# of Sensors is Coupled)

	No Error	High Temp	Low Oil	Rail Guide	Brake Pick
1 Sensor	90.6	77.8	86.7	84.8	83.6
2 Sensors	98.4	92.7	93.1	95.5	95.7
3 Sensors	99.8	97.0	97.4	96.9	97.9

It was clear that having more of each sensor increases the accuracy of the algorithm. The algorithm works best when having multiple sources of data, as it reduces sources of conflict between sensors, resulting in the improved accuracies in Table 14. However, one issue with coupling the number of sensors was that, in practice, it might not be necessary to add a

sensor of each type and could be an additional expense. To remedy this, the Monte-Carlo Simulation was run again, but adjusted such that sensors were added sequentially. For example, one temperature measurement was added and an updated accuracy was determined. The temperature measurement was then removed and an oil level measurement was added. This continued for each measurement type. The accuracy for each addition was recorded and the sensor addition that caused the largest change to accuracy was held, while the rest remained at their previous values. The sensor amounts were updated similar to a gradient descent method in that sensors were added wherever there was the greatest increase in accuracy. Following this method, sensors were added until all of the accuracies were above some threshold, in this case 95 percent.

Table 15 – Average Number of Sensors for 95% Accurate Results

	Temp (F)	**Oil Level (in)**	**Vibration (mG)**	**Frequency (Hz)**	**Noise (dB)**
Calculated	2.05	3.35	2.15	2.3	2.1
Rounded	2	4	3	3	2

The results from running this method 30 times are shown in Table 15. The data for Vibration and Frequency can come from the same sensor and thus the higher of the two numbers was required and would be equal for both data types. The numbers for all the sensors will vary for each application, but the algorithm would be able to determine the number of sensors required provided that adequate data is available for each fault type.

Table 16 – Accuracies of Each Fault using Number of Sensors from Table 15

	No Error	**High Temp**	**Low Oil**	**Rail Guide**	**Brake Pick**
Calculated	98.99	95.58	97.60	97.19	97.10
Rounded	99.01	96.37	98.50	98.23	97.84

After it was shown in Table 14 that the method was capable of achieving a minimum accuracy of 95 percent, that value was used as the threshold for generating the values in Table 15. The accuracies for the calculated and rounded number of sensors in Table 15 are shown in Table 16. The accuracies of the rounded number of sensors are slightly higher due to most of the amounts of sensors being rounded up to the nearest whole number. Not only can the algorithm be used to correctly identify a number of faults, it can also be used to determine a minimum number of sensors that would be needed for an accurate result and therefore minimizing cost while maximizing efficiency.

It should be noted that only one set of means and standard deviations were tested due to computation times increasing as the number of simulations inside of simulations increased. The mean of each fault state was somewhat arbitrary, but the standard deviation could greatly influence the results. Therefore, the large standard deviations used were chosen to simulate uncertain knowledge of fault states. The standard deviations ranged from 10-47 percent of each data types' maximum value ensuring a wide variation in test data. With this variation, the algorithm had some problems with lower numbers of sensors but rapidly increased in accuracy as more were added. The Dempster-Shafer method proved that it quickly becomes more effective with an increasing number of sensors and it could also determine the minimum number of sensors required to achieve a desired level of accuracy.

4.2.2 Kalman Filter

The previous project that used the data from the IMUs shown in section 3.2.1.4 used data to estimate the vibrations experienced at head height by a passenger. The project only used the C_u sensor attached to the cabin as seen in Figure 13 and the data from the other

three sensors was unused. Due to the system being a prototype and therefore still uncertain about the internal parameters and due to the data available, it made sense to use sensor fusion in an attempt to estimate the parameters of a model of the prototype and to compare the estimated parameters against the values used in the previous project.

4.2.2.1 Problem Formulation

From the model shown in Figure 13, the goal was to estimate the parameters between the sledge, where sensor S_u measured, and the bucket. Due to the rigidity of the supports between the weights simulating the cabin and the bucket, these two pieces were treated as one rigid body in both the previous MULTI project and this paper. From this assumption, the S_u and C_u sensors could be used in the fusion algorithm. This was beneficial due to them being the same type of sensor and therefore having the same sample rate. The model was simplified to a spring-mass-damper system as shown in Figure 23 where m_B is the bucket/cabin rigid body, m_S is the sledge, u is the input to the system, and the other variables are the associated acceleration, velocity, and position of each body. The K and C are the spring constant and damping coefficient respectively and the parameters to be estimated to see if this model fits with the recorded data. From this model, the following state space representation of the bucket was formed.

$$\begin{bmatrix} \dot{x}_B \\ \dot{v}_B \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{K}{m_B} & -\frac{C}{m_B} \end{bmatrix} \begin{bmatrix} x_B \\ v_B \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{K}{m_B} & \frac{C}{m_B} \end{bmatrix} \begin{bmatrix} x_s \\ v_s \end{bmatrix} \tag{17}$$

$$y(t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_B \\ v_B \end{bmatrix} \tag{18}$$

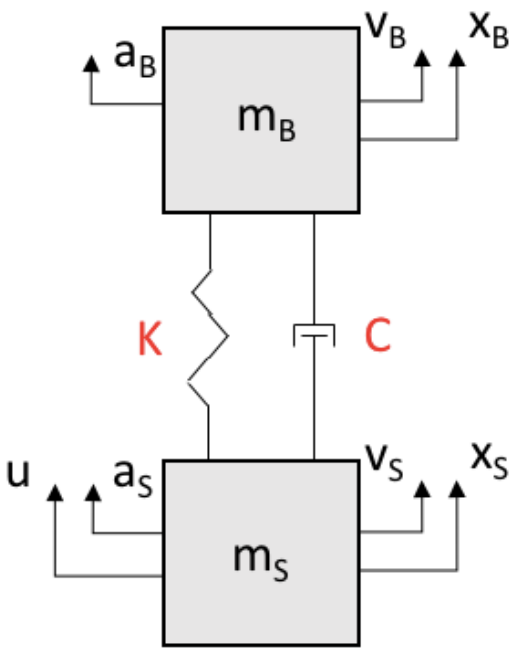


Figure 23: Proposed System Model

The *K* and *C* values are the unknown parameters, the positions and velocities are calculated from the acceleration data recorded and therefore considered as known, and m_B is the combined weight of the bucket and weights used to simulate the cabin. The output *y(t)* is the calculated values of position and velocity from the recorded acceleration data. The state-space representation is then converted to a discrete model using the MATLAB function *c2d* and shown as follows:

$$\begin{bmatrix} x_B(k+1) \\ v_B(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ -\frac{K * \Delta t}{m_B} & 1 - \frac{C * \Delta t}{m_B} \end{bmatrix} \begin{bmatrix} x_B(k) \\ v_B(k) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ \frac{K}{m_B} & \frac{C}{m_B} \end{bmatrix} \begin{bmatrix} x_s(k) \\ v_s(k) \end{bmatrix} \quad (19)$$

$$y(k) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_B(k) \\ v_B(k) \end{bmatrix} \quad (20)$$

The data used in this model is acceleration data of the sledge, S_u, and the cabin, C_u. The acceleration data was converted into velocity and position values by first subtracting

the mean of the data, and then integrating using the MATLAB function *cumtrapz*. One of the rides recorded is shown in Figure 24.

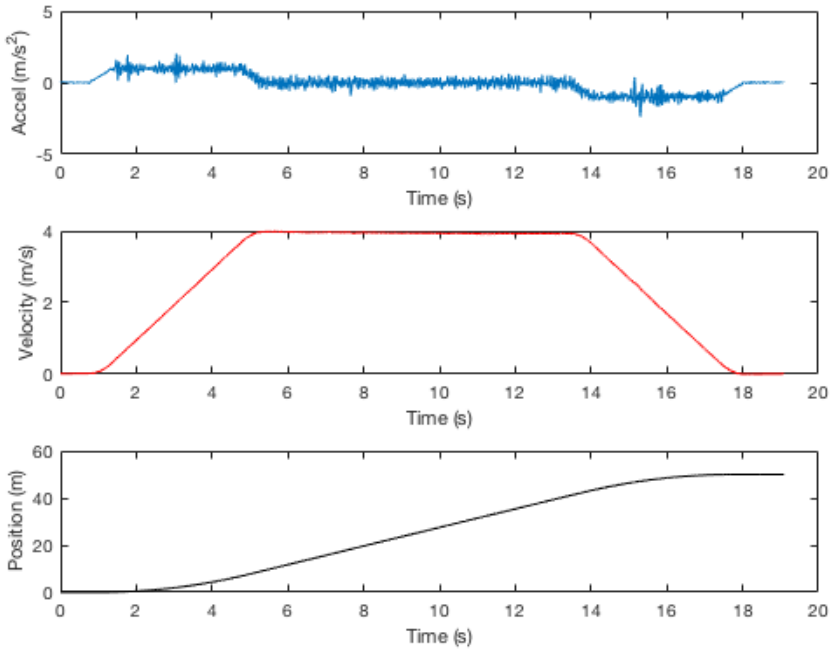


Figure 24: Ride Profile (Vertical, 4 m/s, 50 m)

The data for both the Sledge and the Cabin are in this form. Rides were recorded at three different speeds (3, 4 and 5 m/s). The 4 and 5 m/s rides accelerated at 1.0 m/s^2 and with a travel distance of 50 m. The 3 m/s rides accelerated at 0.8 m/s^2 with a travel distance of 20 m. Rides of each type were used for robustness and accuracy.

After some consideration, due to the ability to work with time varying state space models, the Kalman Filter fit the requirements the best. Generally, the algorithm takes a discretized state space representation of a system and estimates the state of the system at the next time step. However, some versions are capable of simultaneously estimating the internal parameters of the system if they are time-varying or simply unknown. Some of these variants are the Extended Kalman Filter [29], Unscented Kalman Filter [30], and the

Dual Extended Kalman Filter [31]. The Dual Extended Kalman Filter was the variant chosen due to its simplicity and its similarity in concept to other methods such as the Dual Ascent optimization method [32].

The Dual Extended Kalman Filter effectively nests two Kalman filters inside of each other. The outer filter updates the estimates of the parameters, and the inner filter updates the estimate of the state. The formulation is similar to equations 9-13. The equations from Ref. [31] are shown as follows, but using notation similar to equations 9-13, and are based off the state space representation in equations 7 & 8.

Prediction (Paramter):

$$\hat{x}_p(t|t-1) = \hat{x}_p(t-1|t-1) \tag{21}$$

$$P_p(t|t-1) = P_p(t-1|t-1) + Q_p(t) \tag{22}$$

Prediction (State):

$$\hat{x}_s(t|t-1) = f\left(\hat{x}_s(t-1|t-1), u(t), \hat{x}_p(t|t-1)\right) \tag{23}$$

$$P_s(t|t-1) = A(t)P_s(t-1|t-1)A^T(t) + Q_s(t) \tag{24}$$

Estimation (State):

$$K_s(t) = P_s(t|t-1)C^T(t)[C(t)P_s(t|t-1)C^T(t) + R_s(t)]^{-1} \tag{25}$$

$$\hat{x}_s(t|t) = \hat{x}_s(t|t-1) + K_s(t)[y(t) - C(t)\hat{x}_s(t|t-1)] \tag{26}$$

$$P_s(t|t) = [I - K_s(t)C(t)]P_s(t|t-1) \tag{27}$$

Estimation (Parameter):

$$K_p(t) = P_p(t|t-1){H_p}^T(t)[H_p(t)P_p(t|t-1){H_p}^T(t) + R_p(t)]^{-1} \quad (28)$$

$$\hat{x}_p(t|t) = \hat{x}_p(t|t-1) + K_p(t)[y(t) - C(t)\hat{x}_s(t|t-1)] \quad (29)$$

$$P_p(t|t) = \left[I - K_p(t)H_p(t)\right]P_p(t|t-1) \quad (30)$$

Here, $\hat{x}_p$ and $\hat{x}_s$ are the vectors of parameters and state vectors to be estimated, P_p and P_s the covariance matrices of the estimation error, *A(t)* is the Jacobian of the state vector, *C(t)* is the Jacobian of the output vector, *u(t)* the input vector, $Q_p(t)$ and $Q_s(t)$ are noise covariance matrices that can be specified by the user, and the $R_p(t)$ and $R_s(t)$ are the output noise covariance matrices [31]. From the discretized model in equations 19 and 20 the Jacobians are

$$A(t) = \begin{bmatrix} 1 & \Delta t \\ -\dfrac{K * \Delta t}{m_B} & 1 - \dfrac{C * \Delta t}{m_B} \end{bmatrix} \quad (31)$$

$$C(t) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (32)$$

The noise covariance matrices come from the standard deviation of the output readings

$$R_p(t) = R_s(t) = \begin{bmatrix} {s_x}^2 & 0 \\ 0 & {s_v}^2 \end{bmatrix} \quad (33)$$

and are the same because the output vectors for both the parameters and the state are the same. The $Q_p(t)$ and $Q_s(t)$ matrices are calculated as in the method in Ref. [31] as

$$Q_p(t) = \begin{bmatrix} {s_k}^2 & 0 \\ 0 & {s_c}^2 \end{bmatrix} \quad (34)$$

$$Q_s(t) = \begin{bmatrix} 10000 & 0 \\ 0 & 10000 \end{bmatrix} \quad (35)$$

where the variables s_k and s_c are set to approximately 1% of the expected parameter values. The Jacobian for the parameters is calculated using the method described in Ref. [31] and Ref. [33]

$$H_p(t) = H_s \frac{\partial f(\hat{x}_s, \hat{x}_p)}{\partial \hat{x}_p} = \begin{bmatrix} \frac{\partial x}{\partial K} & \frac{\partial x}{\partial C} \\ \frac{\partial v}{\partial K} & \frac{\partial v}{\partial C} \end{bmatrix} \tag{36}$$

H_s in this case is the Jacobian of the state space equation, or *A(t)* in equation (31. The discretized state space model and the associated parameters could then be simulated in a Kalman Filter.

4.2.2.2 Simulation Results

Using the initial matrices from 4.2.2.1 and the recorded data, a simulation was run with MATLAB. The Dual Extended Kalman Filter generated a spring constant *K* of 1,763 N/m and a damping coefficient *C* of 410 N/m*s^{-1}. With these values, a basic Kalman Filter, as shown in equations 9-13, was run to check against the recorded output data. The results from this method for the velocity and position are shown in Figure 25 and Figure 26 respectively. The estimated velocity results are extremely accurate with very little drift while the position estimate is almost exact. The errors for both velocity and position are shown in Figure 27 and Figure 28.

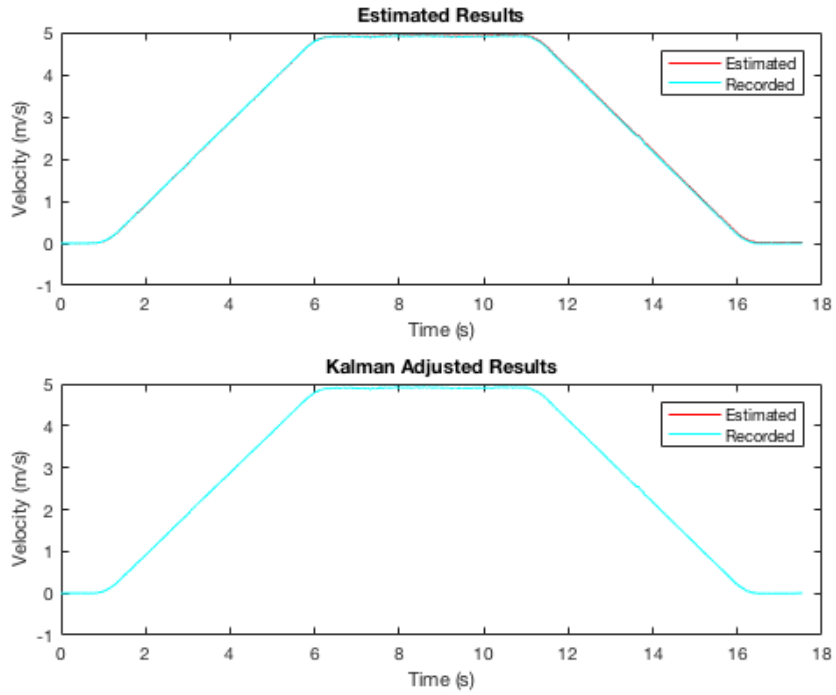


Figure 25: Dual Extended Kalman Filter Parameter Estimation Results – Velocity

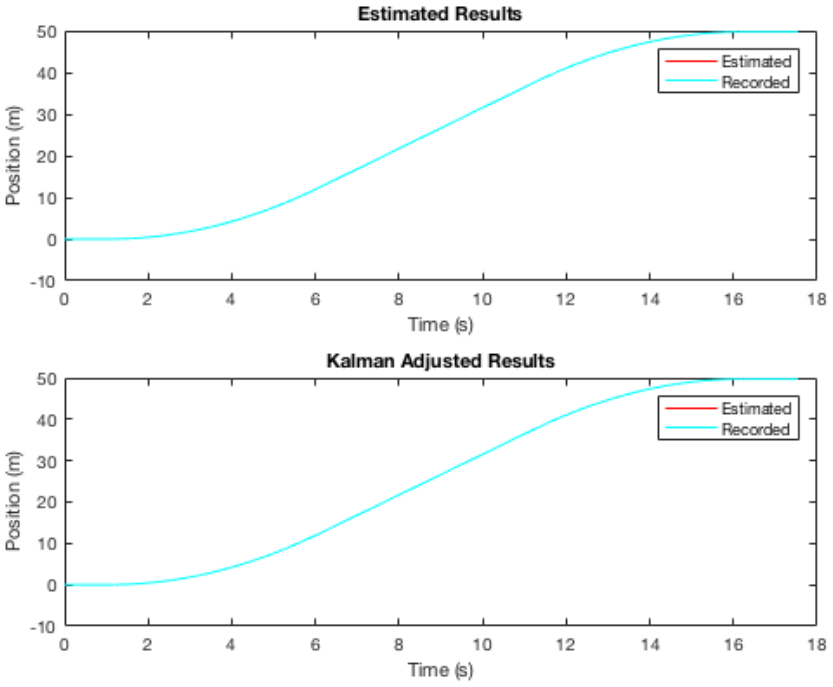


Figure 26: Dual Extended Kalman Filter Parameter Estimation Results – Position

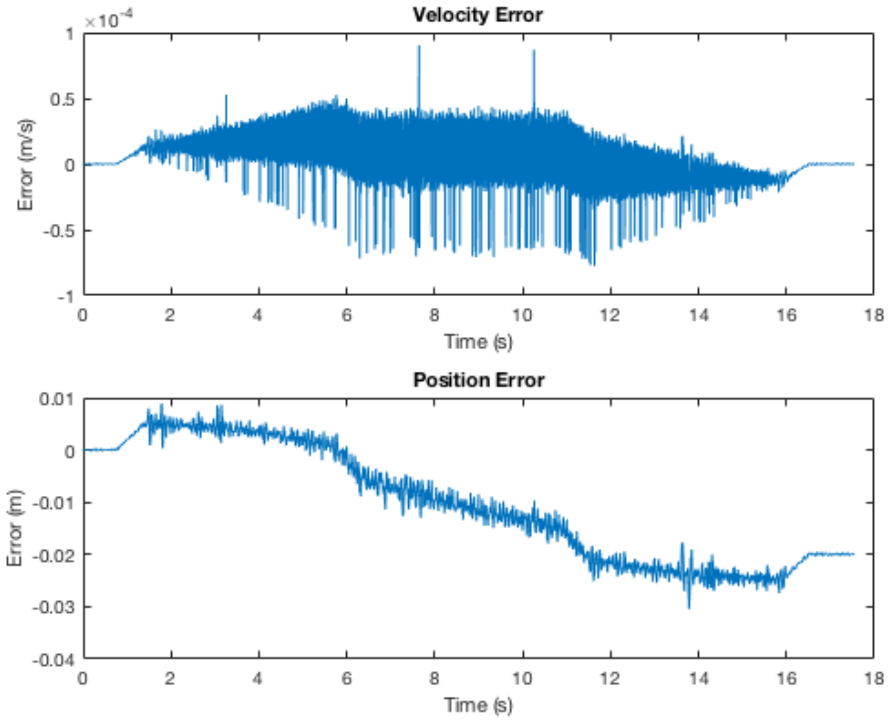


Figure 27: Error Between Recorded and Estimated Output States

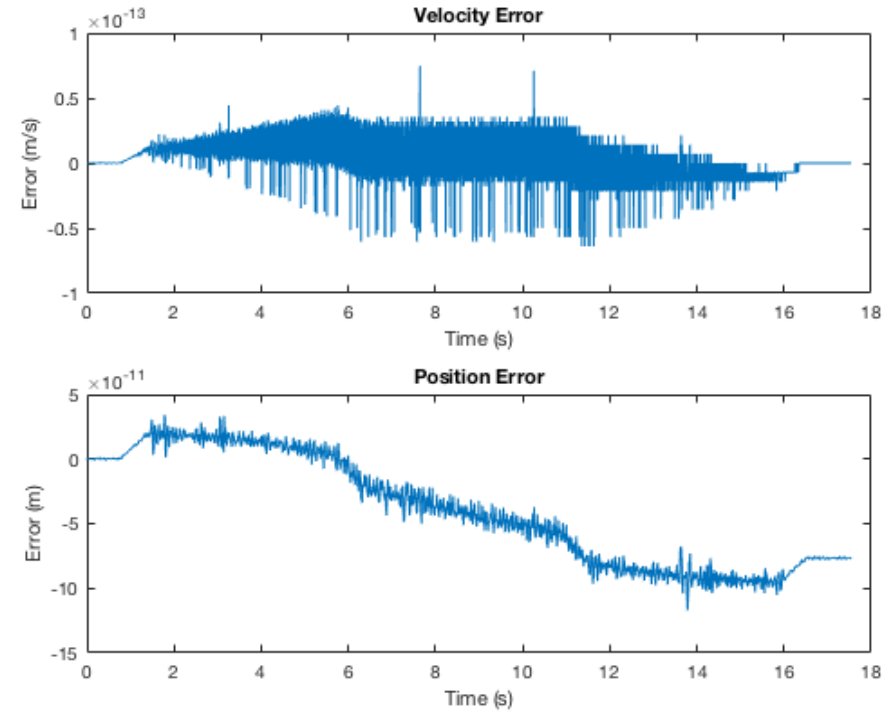


Figure 28: Error Between Recorded and Kalman Adjusted Output States

While this method seems to be extremely accurate, there were some problems with some of the data sets. For example, when a set of data recorded at 4 m/s moving downward was singularly run through the Dual Extended Kalman Filter, the algorithm estimated physically impossible parameters. The spring constant K was estimated to be $1.736*10^6$, and the damping coefficient C was estimated to be $-4.147*10^4$. In this regard, a negative damping coefficient is inconsistent with a realistic response. Using these parameters with a basic Kalman Filter generates the estimates shown in Figure 29 and Figure 30. Interestingly, the position estimation is still very accurate, while the velocity estimate, before the Kalman adjustment, exhibits extreme drift. The errors for both the velocity and position are shown in Figure 31 and Figure 32.

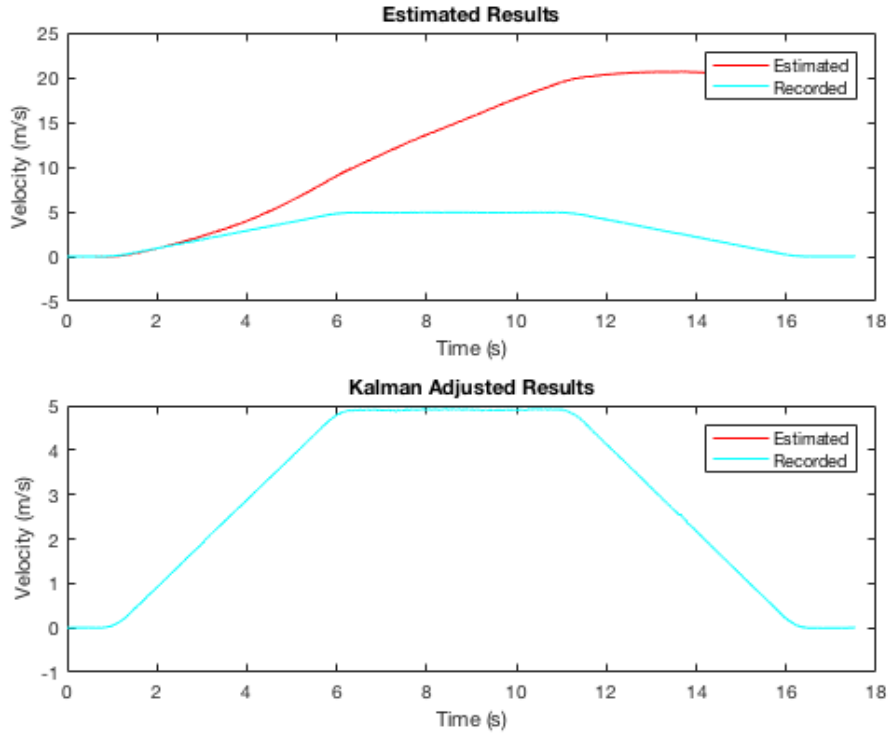


Figure 29: Velocity Estimation using Physically Impossible Parameters

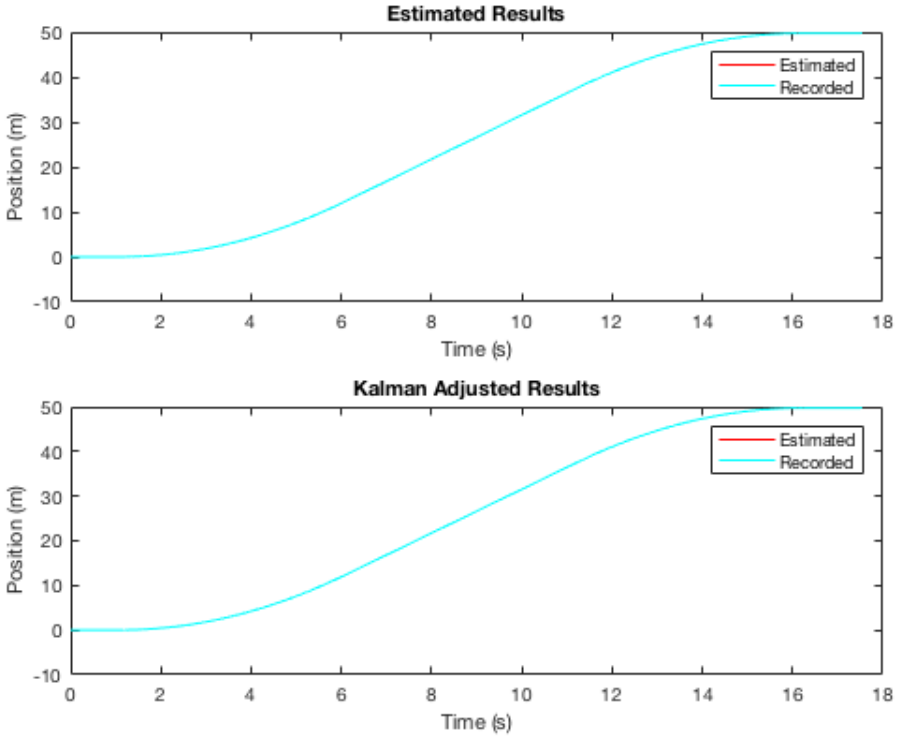


Figure 30: Position Estimation using Physically Impossible Parameters

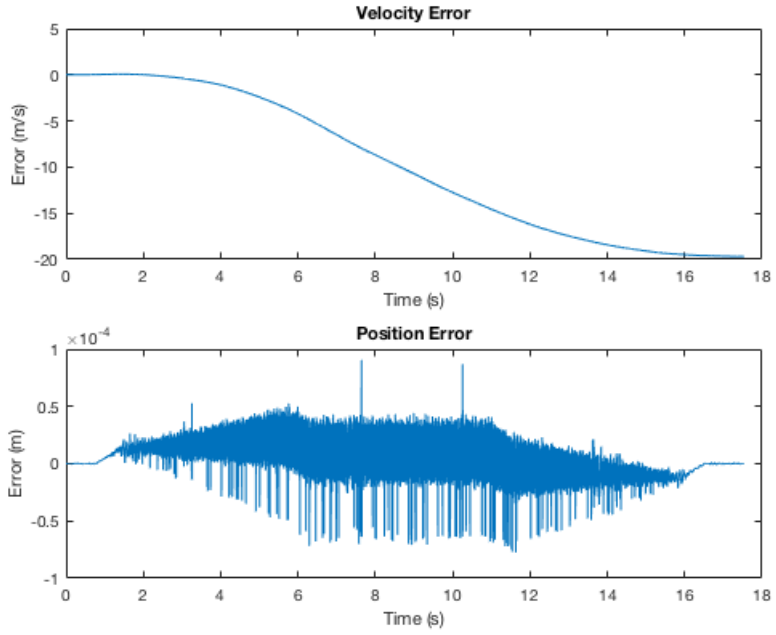


Figure 31: Error Between Recorded and Estimated Output States

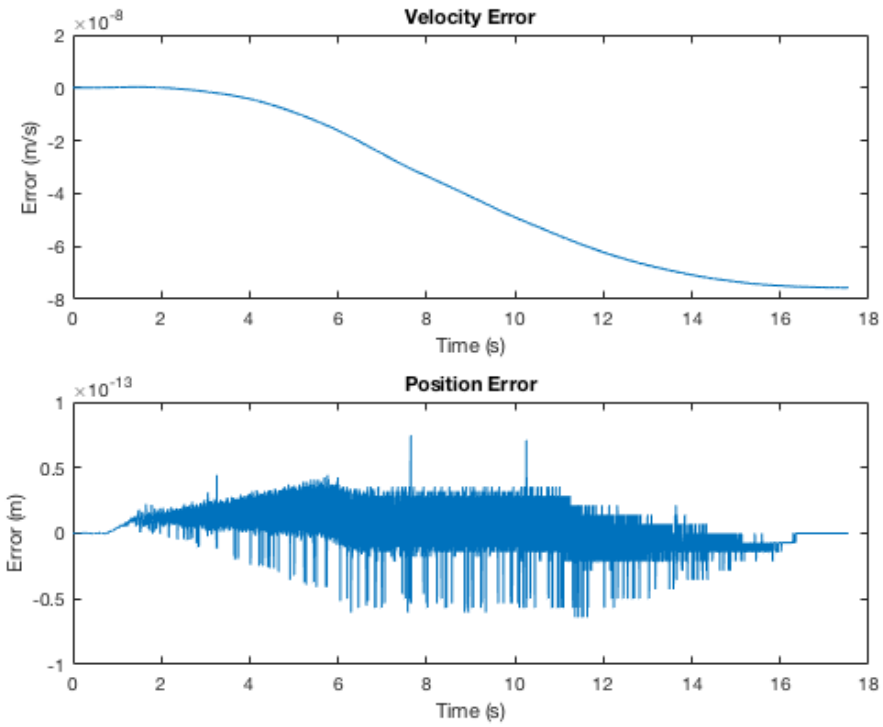


Figure 32: Error Between Recorded and Kalman Adjusted Output States

The error between the estimated and recorded velocity is clear and expected based on the large differences visible in Figure 29. The position estimation, surprisingly, is still extremely accurate despite the parameters. Due to the Dual Extended Kalman Filter giving such obviously inaccurate results for some data sets, another verification method was needed.

4.2.2.3 Alternative Parameter Estimation

While the Kalman Filter does have variants that allow for simultaneous parameter estimation, the common way of estimating parameters is through a form of regression applied to recorded data. With this in mind, the *fminsearch* MATLAB function could be used to estimate the parameters. This function uses the Nelder-Mean simplex algorithm and it is built in to MATLAB making it preferable over creating or coding a new algorithm.

The function takes in an equation to be optimized and specifies what variables are to be estimated for optimization. The function also needs an initial guess for the parameters in question as a starting point. The equation could be the time-varying or discrete versions of the state-space model. In order to use the raw data, the left side of the time-varying equation 17 was subtracted so that the minimum should be zero as seen in equation 37.

$$\frac{K}{m_B}(x_s - x_B) + \frac{C}{m_B}(v_s - v_B) - \dot{v}_B = 0 \tag{37}$$

Most of the data sets available were used to update the estimation of the parameters, while some were reserved for testing in the Kalman Filter. Because the optimization function is run for every time step of each dataset, this method took much longer than the Dual Extended Kalman Filter.

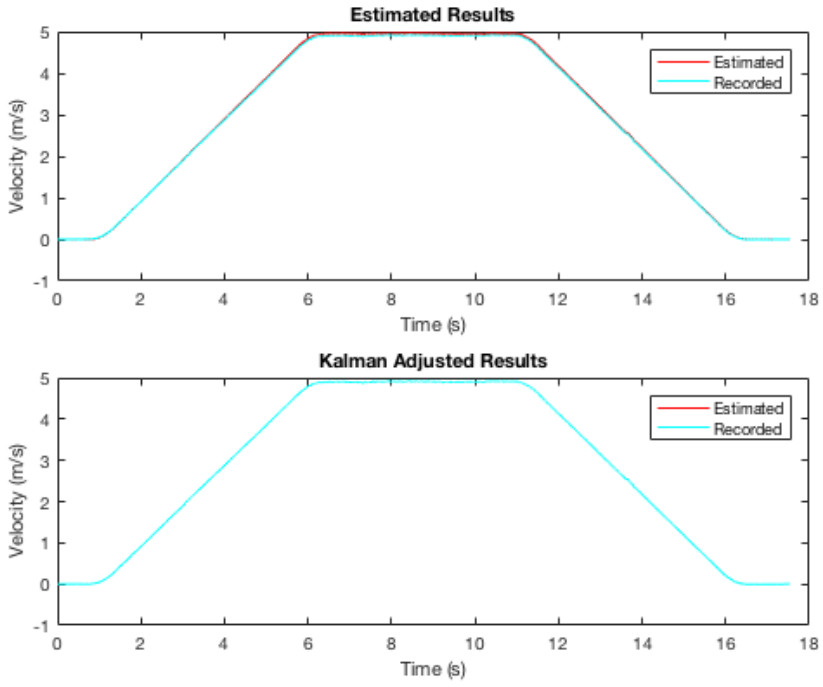


Figure 33: Velocity Estimation using Parameters from *fminsearch*

After running the datasets through the *fminsearch* function, the spring constant K was estimated to be 4.86 N/m and the damping coefficient C as $4.32*10^4$ N/m*s^{-1}. These values were inputted to a basic Kalman Filter and generated the results in Figure 33 and Figure 34. The errors for the estimated and Kalman adjusted values are then shown in Figure 35 and Figure 36. The errors for both velocity and position are similar in magnitude to the successful tests of the Dual Extended Kalman Filter, except in this case every dataset was run through the parameter estimation without generating impossible parameters. As with the previous attempt, some datasets were reserved for testing. There is some error, more noticeable in the velocity estimation, but again the Kalman adjustment step reduces the error to be negligibly small.

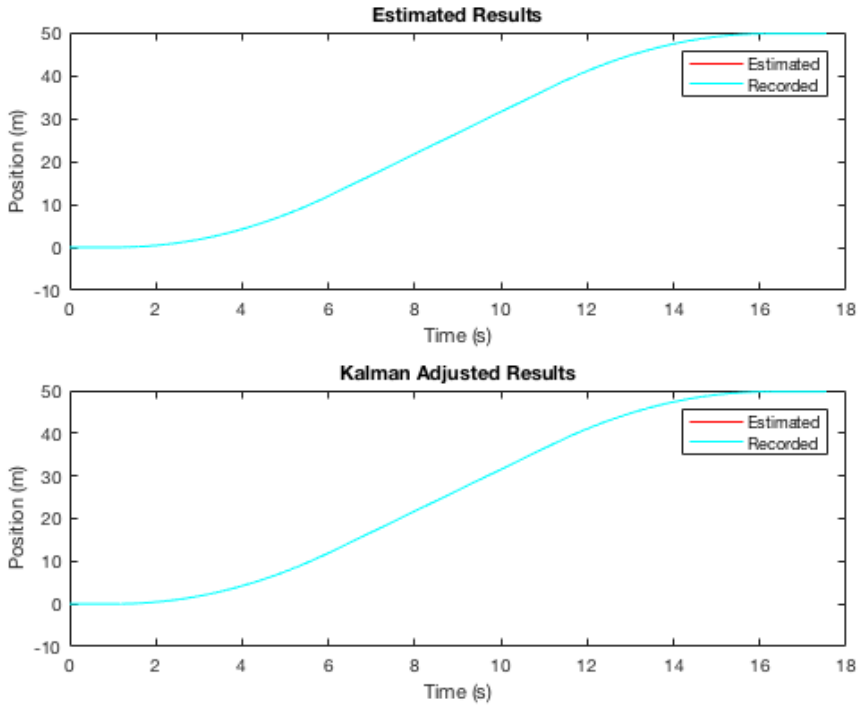


Figure 34: Position Estimation using Parameters from *fminsearch*

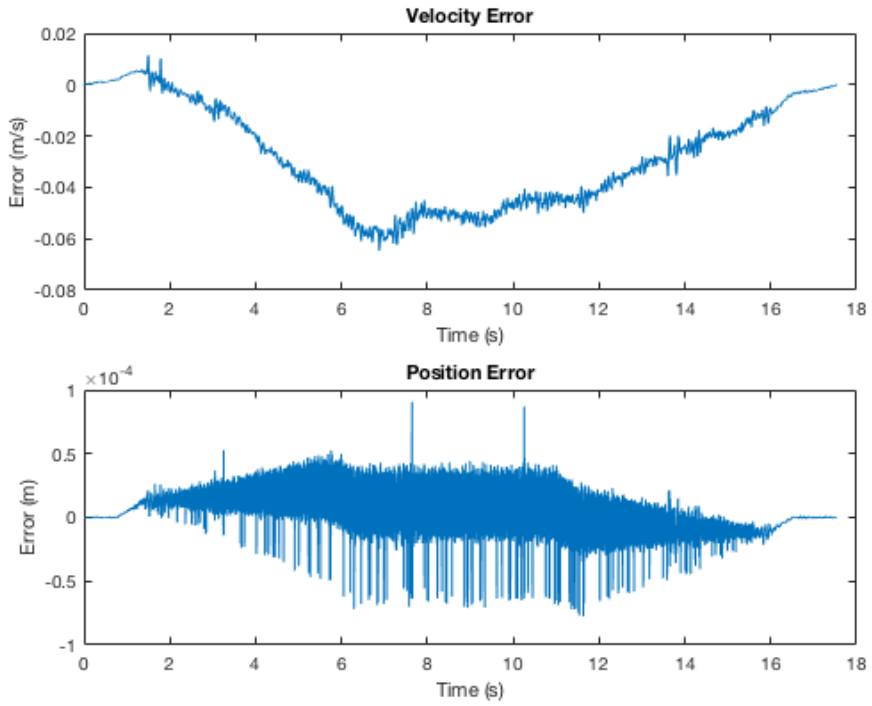


Figure 35: Error Between Recorded and Estimated Output States

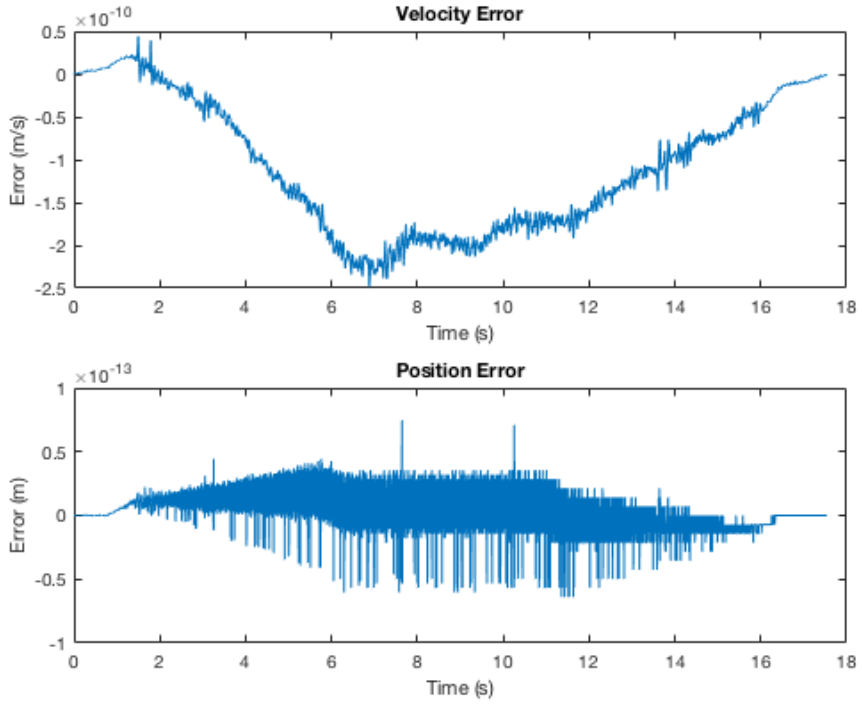


Figure 36: Error Between Recorded and Kalman Adjusted Output States

4.2.2.4 Kalman Filter Analysis

While the estimated parameters from the Nelder-Mead function and the successful Dual Extended Kalman Filter sets are physically possible, and generate extremely accurate results, there are still some elements that may suggest they are not accurately representing the data. The damping ratio and natural frequency of the system can be calculated from the spring constant and damping coefficients from equations 38 and 39.

$$\zeta = \frac{1}{2} * \frac{C}{\sqrt{K * m_B}} \tag{38}$$

$$\omega_n = \sqrt{K/m_B} \tag{39}$$

For the Dual Extended Kalman Filter, the estimated spring constant and damping coefficient generate a damping ratio of 0.1941 and natural frequency of 1.67 Hz. For the Nelder-Mead estimation method, the damping ratio is 389.4339 which would suggest that the system is extremely overdamped. The natural frequency was calculated to be 0.0876 Hz. The damping ratio from the Nelder-Mead method is extremely large and the frequency is extremely small. However, every dataset was run through the optimization function without generating impossible results. The damping ratio calculated from the Dual Extended Kalman Filter method is between zero and one which would suggest the system is underdamped which could make sense if the some of the datasets were not generating impossible physical parameters.

Some other aspects that could be affecting the algorithms include disparity in the datasets themselves or that the model is too simplified and does not accurately reflect how the system behaves. The disparity in the data is visible in Figure 37 and Figure 38.

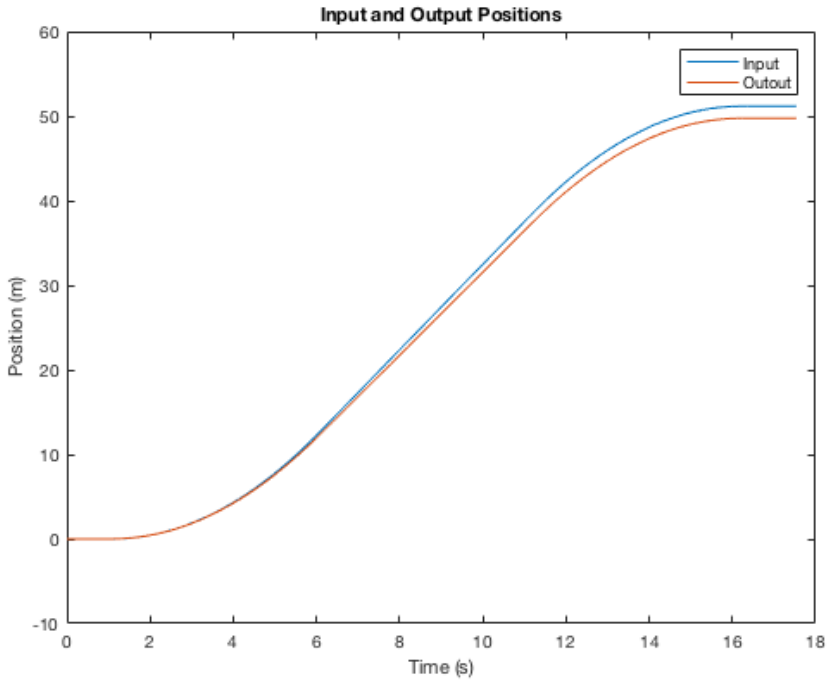


Figure 37: Discrepancy between Input and Output Positions

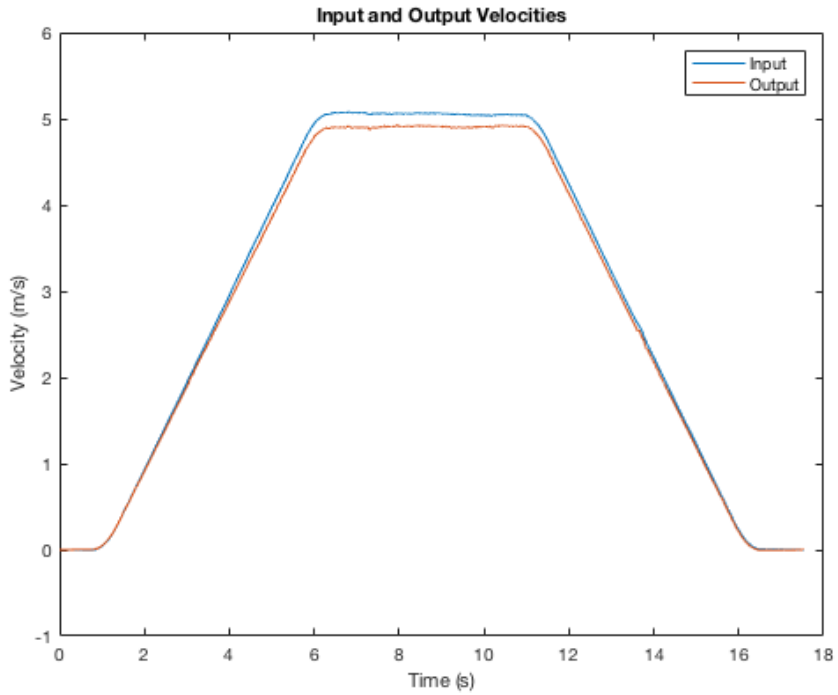


Figure 38: Discrepancy between Input and Output Velocities

The two bodies, the sledge and the cabin/bucket, constitute the input and output values respectively and are shown in the previous figures. They are attached in a way such that

the difference of 1m shown in Figure 37 is not possible without the system breaking. The velocity difference shown in Figure 38 also does not fit the system. The two bodies should be traveling around the same speed, with only minor fluctuations due to the dynamics present between them. While it is possible that the model is oversimplified or does not accurately describe the system, it is more likely that there was a problem generating the velocity and position data from the recorded acceleration data. One final issue could be with the types of data being recorded. Because the exact state vector, containing position and velocity, is being recorded, the Kalman Filter update step, shown in equations 12, 26, and 29, is correcting the state estimate to the exact value. Typically, in a Kalman filter scenario, information about the state vector is only partly known meaning that the rest of the state must be estimated. Having access to the exact values reduces, if not eliminates, the need to estimate the state of the system. This could potentially be affecting the accuracy of the parameter estimation but would require further testing using different measurement vectors.

4.3 Algorithm Discussion

4.3.1 Dempster-Shafer Theory of Evidence

The Dempster-Shafer method showed that even when the exact fault states are unknown or have large uncertainty, it can still generate accurate results. The inaccuracies inherent from the large standard deviations could account for uncertainty in the fault states themselves or theoretically could be from the uncertainty in a sensor. Either way, it was shown to be able to determine the number of sensors required for each fault state the be accurately predicted. Due to the form of the evidence combination, it is also possible for

many fault states to be checked at once without severely increasing the computation time. Due to this, the algorithm would be relevant to the upcoming internal thyssenkrupp project titled MAX 2.0. Now that the data transmission interface is in place from MAX 1.0, the next step of the project is to determine which sensors can be added to an elevator that will generate the most value. The sensors value will likely be determined on a basis of maintenance, but whether it will be reactive, predictive, or preventative is currently unknown. For many cases, the Dempster-Shafer method could be a valuable tool in implementing a maintenance-focused monitoring algorithm.

4.3.2 Kalman Filter

The Kalman Filter has many applications further than the one examined in this paper. There are many other potential uses for the filter on the MULTI project, but potential projects outside of this include real time monitoring of the wear of a carbon fiber belt. For the elevator applications with multiple shafts, the Dual Extended Kalman Filter could be used as a real time parameter estimator to allow for active damping to minimize vibrations experienced by passengers and therefore maximizing ride quality. The algorithm could also be used on the other connection, between the bucket and the cabin, if the assumption that they are one rigid body changes. For the carbon fiber belt, a set of steady state measurements can generate the parameters for a state space representation of the system and then the algorithm can monitor for deviations. These are but a couple of examples, and of course there are many more possibilities.

CHAPTER 5. CONCLUSIONS AND NEXT STEPS

5.1 Results Interpretation

Elevators are still in a nascent state of intelligent monitoring, but there are many opportunities to improve. Companies such as thyssenkrupp have begun transitioning elevators into cyber-physical systems, but there are many more sensors that can be added and algorithms to implement that are necessary to reach the top of the CPS hierarchy. The sensors discussed in this paper, temperature, fluid level, distance, and acceleration, all can record data that can be used to generate new information about the elevator. Ongoing technology deployment projects include health check monitoring and others related to ubiquitous sensing and fiber belt monitoring also pose significant opportunities.

The sensors chosen in sections 3.2.1.1 - 3.2.1.3, temperature, oil level, and distance, can be used in conjunction with the Dempster-Shafer method for maintenance. Initially the sensors and the algorithm fulfil the bottom two sections of the CPS hierarchy, the Smart Connection and Data-to-Information Conversion levels. However, this is an important first step to moving up the hierarchy and creating a smart device. With the data that the sensors provide, and the information that a sensor fusion algorithm can generate, multiple algorithms could be combined for predictive or even preventative maintenance. If the system proposed and analysed in this paper is implemented, it could complement ongoing sensing projects and be expanded to include many more sensors giving a comprehensive health metric of the entire elevator system. Additionally, the Kalman Filter and its variants are widely used in multiple industries and could be very beneficial to the elevator industry as discussed in the above sections.

5.2 Next Steps and Future Work

There are next steps for both the Dempster-Shafer algorithm and the Kalman Filter work done in this paper, as well as work that can be done in the future. The Dempster-Shafer analysis in this paper was done with theoretical data and a Monte-Carlo simulation. The next steps would include recording data for the fault states chosen in section 3.1.2 and generate a look up table that describes the fault states and healthy states. Another step would be to install the network proposed in section 3.2.2 on an elevator to collect data and test if the algorithm can accurately determine when a fault occurs. Should any problems arise while testing, further changes and steps could be required.

Due to the problems occurring with the Kalman-Filter parameter estimation, the next steps would be to determine the source of the error in the estimation. First, different methods of preprocessing the data could improve the discrepancies seen in Figure 37 and Figure 38 and subsequently improve the effectiveness of the parameter estimation. If the issues still persist, one option would be to record the position and velocity directly instead of calculating them from the recorded acceleration data. Another option would be to change the measurement vector to be different from the state vector so that the algorithm cannot rely on an exact estimate of the state and must rely on the parameter estimation to improve accuracy. Lastly, if none of these changes are effective, then the model might require adjusting as it either is simplified too much from the actual system or does not accurately describe the system.

Future work, as mentioned previously, would be to implement the findings from this paper on other projects that thyssenkrupp is currently pursuing or planning to pursue. These include the carbon fiber belt real time monitoring and the MAX 2.0 projects. Both projects

will start in the Atlanta research center in Fall 2018 and provide a great opportunity for thyssenkrupp to move its elevators up the CPS hierarchy towards the cognition and even configuration levels.

APPENDIX A. SPEC SHEETS FOR SENSORS

A.1 TT Electronics Linear Position Sensor

A.2 Milone Technologies eTape Fluid Sensor

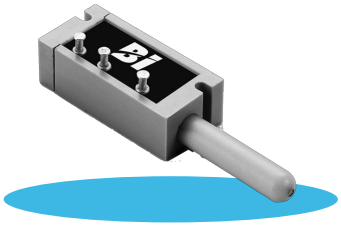
A.3 DS18B20 Temperature Sensor

A.4 iMAR IMU

A.5 Sensor-Technik Wiedemann GmbH IMU

Model 404 Series

Position Sensor



Features:

- Short travel (0.5 inches), linear motion with spring return
- Accurate position feedback
- Compact design for small spaces
- Conductive plastic technology
- Long life (5 million actuations)

Applications:

- Robotics
- Automotive
- Heavy equipment
- Industrial automation
- Wing flap position
- Pedal position
- Satellite dish
- Electro surgical equipment

Electrical

Resistance range	1K to 150K Ohms
Standard resistance tolerance	±10%
Minimum practical resistance tolerance	±5%
Independent linearity	±1%
Minimum practical independent linearity	±0.5%
Input voltage	400 VDC maximum, not to exceed power rating
Dielectric strength	1,000 V rms
Insulation resistance	1,000 Megohms minimum
Output smoothness	0.1% maximum at 10” to 18” per minute
Actual electrical travel	0.50” ±0.015” (12.7 mm ±0.38 mm)
Electrical continuity travel	Within mechanical travel
End voltage	Maximum 0.5% of input voltage
Resolution	Essentially infinite
Temperature coefficient of resistance	-400 ppm/°C typical
Temperature coefficient of output voltage	±10 ppm/°C typical

Mechanical

Torque mechanical travel	0.56” ±0.015” (14.2 mm ±0.38 mm)
Actuating force	14 oz. maximum, internal spring to return slider to extended position
Backlash	0.003” maximum
Static stop strength	20 lb. minimum
Body style	Rectangular
Termination style	Turret terminations

BI Technologies
4200 Bonita Place, Fullerton, CA 92835 |Ph: +1 714 447 2300
www.bitechnologies.com | www.ttelectronics.com

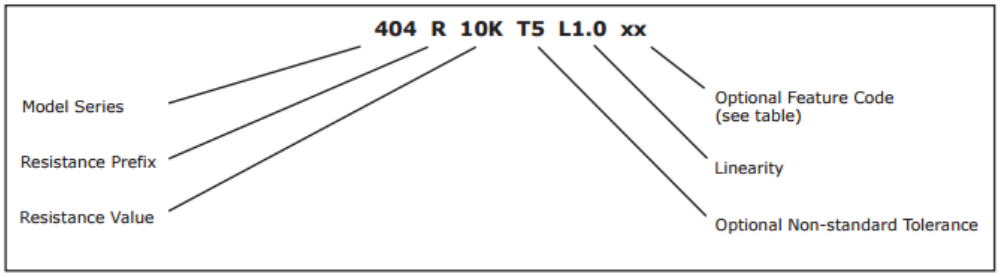
Model 404 Series

Position Sensor

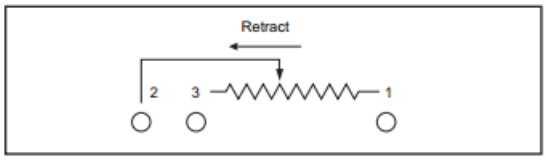
Environmental (MIL-PRF-39023)

Operating temperature range	-40°C to +125°C dynamic, -55°C to +125°C static
Load life	1 million shaft cycles at 0.25 Watts & 70°C, maximum 10% ΔR

Ordering Information



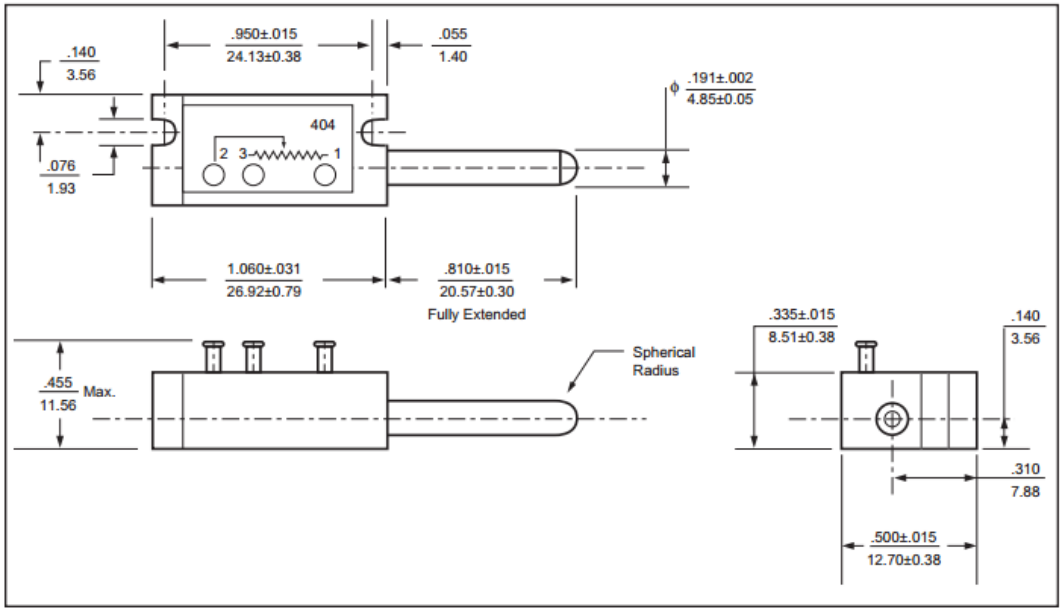
Circuit Diagram



Feature Codes

Linearity Tape	LT

Outline Drawing



BI Technologies
4200 Bonita Place, Fullerton, CA 92835 |Ph: +1 714 447 2300
www.bitechnologies.com | www.ttelectronics.com

eTape™ Continuous Fluid Level Sensor PN-12110215TC-8

Description

The eTape sensor is a solid state, continuous (multi-level) fluid level sensor for measuring levels in water, non-corrosive water based liquids and dry fluids (powders). The eTape sensor is manufactured using printed electronic technologies which employ additive direct printing processes to produce functional circuits.

Theory of Operation

The eTape sensor's envelope is compressed by hydrostatic pressure of the fluid in which it is immersed resulting in a change in resistance which corresponds to the distance from the top of the sensor to the fluid surface. The eTape sensor provides a resistive output that is inversely proportional to the level of the liquid: the lower the liquid level, the higher the output resistance; the higher the liquid level, the lower the output resistance.

Specifications

Sensor Length: 10.1" (257 mm)

Thickness: 0.015" (0.381mm)

Width: 1.0" (25.4 mm)

Active Sensor Length: 8.4" (213 mm)

Sensor Output: 1500Ω empty, 300Ω full, ±10%

Resistance Gradient: 140Ω /inch (56Ω/cm), ±10%

Resolution: < 0.01"(0.25 mm)

Actuation Depth: Nominal 1" (25.4 mm)

Reference Resistor (Rref): 1500Ω, ±10%

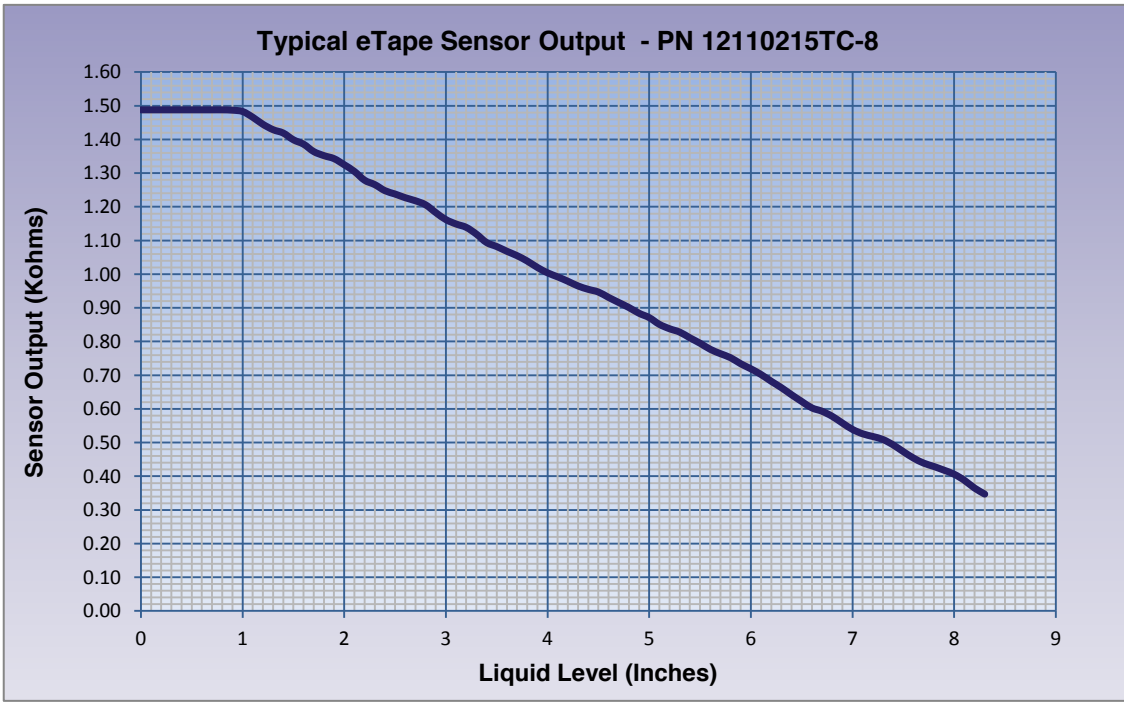
Connector: Crimpflex Pins

Temperature Range: 15°F - 150°F (-9°C - 65°C)

Power Rating: 0.5 Watts (VMax = 10V)

Sensor Output

The eTape can be modeled as a variable resistor (300 – 1500 Ω ± 10%). The typical output characteristics of the eTape sensor are show in the figure below:

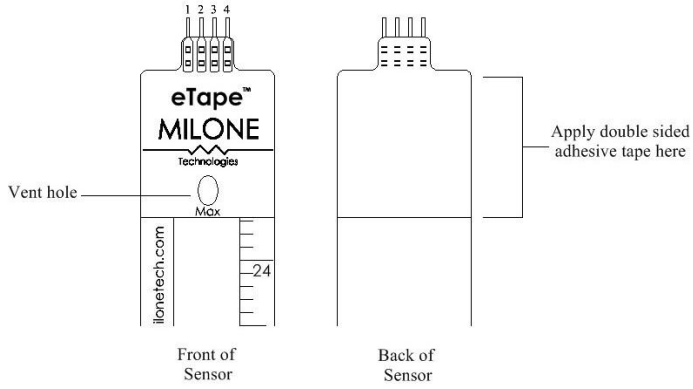


Milone Technologies, Inc - 17 Ravenswood Way - Sewell, New Jersey 08080 - Phone: (856) 270-2688
Email: info@milonetech.com Web: www.milonetech.com

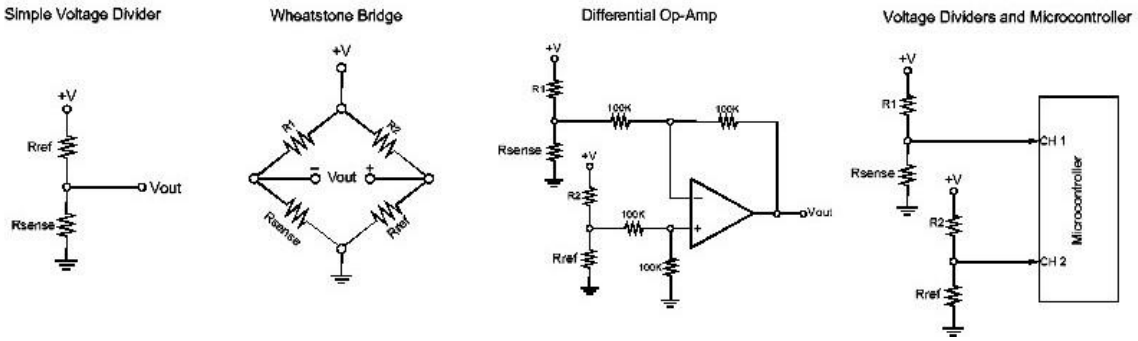
eTape™ Continuous Fluid Level Sensor PN-12110215TC-8

Connection and Installation

Connect to the eTape by attaching a 4 pin connector with pre-soldered wires to the Crimpflex pins. Do not solder directly to the Crimpflex pins. The inner two pins (pins 2 and 3) are the sensor output (Rsense). The outer pins (pins 1 and 4) are the reference resistor (Rref) which can be used for temperature compensation. Suspend the eTape sensor in the fluid to be measured. To work properly the sensor must remain straight and must not be bent vertically or longitudinally. For best results install the sensor inside a section of 1-inch diameter PVC pipe. Double sided adhesive tape may be applied to the upper back portion of the sensor to suspend the sensor in the container to be measured. However, the liquid must be allowed to interact freely with both sides of the sensor. The vent hole located above the max line allows the eTape to equilibrate with atmospheric pressure. The vent hole is fitted with a hydrophobic filter membrane to prevent the eTape from being swamped if inadvertently submerged.



Sample Circuits



Custom Applications

The eTape sensor can be manufactured in custom lengths to fit any application. Contact Milone Technologies if you have an application that requires specific length, configuration or output characteristics.

Technical Support

If you require technical support for the eTape liquid level sensor, please contact our technical support department by email at: techsupport@milonetech.com.

Innovative Fluid Sensing

MILONE Technologies

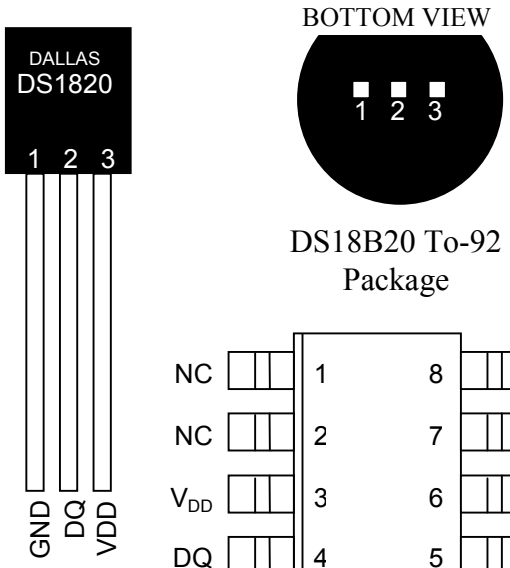
Milone Technologies, Inc - 17 Ravenswood Way - Sewell, New Jersey 08080 - Phone: (856) 270-2688
Email: info@milonetech.com Web: www.milonetech.com

DS18B20
Programmable Resolution 1-Wire® Digital Thermometer

FEATURES

- Unique 1-Wire interface requires only one port pin for communication
- Multidrop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Zero standby power required
- Measures temperatures from -55°C to +125°C. Fahrenheit equivalent is -67°F to +257°F
- ±0.5°C accuracy from -10°C to +85°C
- Thermometer resolution is programmable from 9 to 12 bits
- Converts 12-bit temperature to digital word in 750 ms (max.)
- User-definable, nonvolatile temperature alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

PIN ASSIGNMENT



DS18B20 To-92 Package

DS18B20Z
8-Pin SOIC (150 mil)

PIN DESCRIPTION

GND - Ground
DQ - Data In/Out
V_{DD} - Power Supply Voltage
NC - No Connect

DESCRIPTION

The DS18B20 Digital Thermometer provides 9 to 12-bit (configurable) temperature readings which indicate the temperature of the device.

Information is sent to/from the DS18B20 over a 1-Wire interface, so that only one wire (and ground) needs to be connected from a central microprocessor to a DS18B20. Power for reading, writing, and performing temperature conversions can be derived from the data line itself with no need for an external power source.

Because each DS18B20 contains a unique silicon serial number, multiple DS18B20s can exist on the same 1-Wire bus. This allows for placing temperature sensors in many different places. Applications where this feature is useful include HVAC environmental controls, sensing temperatures inside buildings, equipment or machinery, and process monitoring and control.

GESELLSCHAFT FUER INERTIALE MESS-, AUTOMATISIERUNGS- UND REGELSYSTEME MBH
WWW.IMAR-NAVIGATION.DE

iμIMU-01/ iμVRU-01

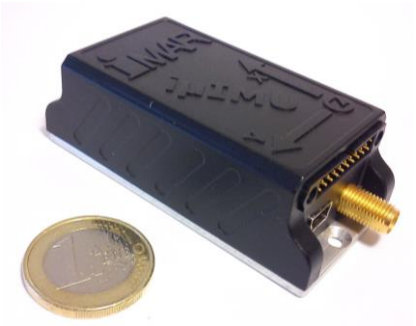
Micro IMS with integrated GPS, Magnetometer, Barometer, Odometer

The iμIMU-01 is a MEMS based low-cost IMU consisting of 3 MEMS gyro axes and 3 MEMS accelerometer axes, baro, 3D magnetometer, GPS and odometer interface. The iμVRU-01 additionally provides attitude, velocity, position and true heading (AHRS).

- Calibrated sensor data
- Up to 1'000 Hz data rate with calibrated data
- Filtered power supply via USB
- iμVRU: Attitude Heading Reference, Surveying, UAV & missile Guidance & Control Applications
- Integrated L1 GPS, magnetometer, barometer (altimeter) and odometer interface
- Precise UTC referenced output
- RS232, RS422, USB, CAN, CANaero
- SYNC input and output for time stamping
- Compatible to external iDAGOS 2-antenna GPS heading reference and up to RTK GNSS aiding

The iμVRU is delivered with fully calibrated gyro and acelerometer axes. The IMU is designed for ruggedized industrial applications on autonomous guided vehicles, land vehicles, marine vessels and aircrafts. The iμVRU-01 and the iμIMU-01 can be operated at an unregulated wide range power supply (7-34 V DC and via USB 5V) and is protected against wrong polarity.

The iμVRU contains an integrated AHRS processor which provides roll, pitch, heading, position and velocity information.



iMAR's iμVRU is delivered with iMAR's PC based iXCOMsoftware for data collection, parameter adjustment and command interface.

The iμIMU / iμVRU are neither covered by any export control nor by ITAR restrictions. The iμVRU is used in unmanned aerial vrehicles (UAV) as well as in manned and unmanned naval, aerial and automotive vehicle applications.

Technical Data of i*μ*IMU-01 / i*μ*VRU-01 (typical 1 sigma values):

	Angular Rate[1]	Acceleration[1]	Altitude (Baro)	Magnetometer
Sensor Range:	± 250 °/s [2'000 °/s]	± 4 g [16 g]	300…1100 hPa -500…9000 m	± 8 Gauss ± 0.8 mT
Bias (OTR):	< 1 °/s (typ. < 0.2 °/s)	< 15 mg (typ. < 8 mg)	< 1 hPa	< 2 mGauss / 0.2 μT
Bias Stability (AllanVar):	< 10 °/h (@ const.temp)	2 mg	< 0.1 hPa	
Resolution [@range]:	< 0.003 °/s [@ 250 °/s]	< 0.1 mg [@ 4 g]	0.01 hPa (0.1 m)	< 1 mGauss / 0.1 μT
Linearity / Scale error:	< 0.2 % / < 1 %	0.5 % / < 0.5 %		0.1 % / 5 %
Angular random walk, Noise:	0.015 °/s/√Hz (@ 10 Hz)	< 1 mg/√Hz	0.01 hPa/√Hz	
g dependent Drift (comp.):	< 0.01 °/s/g			
Data Rate / Bandwidth:	up to 1'000 Hz / 200 Hz		40 Hz	75 Hz
GPS:	2.5 m CEP, 27 sec cold start, 3 sec aided start; WAAS/EGNOS/MSAS supported			
Output iμIMU-01:	calibrated rate & acceleration, pressure, magnetic field vector, odometer counts, GPS			
iμVRU-01:	additional: **roll/pitch 1.5° static, 0.5° dyn.; INS/GPS heading < 0.5°** [2]; mag. heading < 1°			
Inertial Axis Misalignment:	< 2 mrad between all inertial sensor axes (calibrated)			
Digital Interface:	RS232, RS422, USB, CAN; INS/GNS data up to 200 Hz, calibrated sensor data 1 kHz ARINC-825-light / CANaero-light			
Connector:	Harwin M80, 20 pin; SMA for GPS antenna; Micro USB			
Data rate:	up to 200 Hz all navigation data, or up to 1'000 Hz for calib. raw data			
SYNC:	Option: RS422 level SYNC input to reset internal package counter			
Temperature:	-40...+71 °C (operating, case temperature); magnetometer: -30…+85 °C -45...+85 °C (storage)			
Shock, Vibration:	6 g, 20 ms ½ sine saw-tooth; 10...2'000 Hz 2 g rms (operation) 6.3 g rms (endurance); shock and vibration may affect performance			
Environment / MTBF/ MTTR:	IP54 / > 25.000 hrs (estimated) / 2 minutes			
Size, Weight:	approx. 73.5 x 23 x 34 mm (plus connector), approx. 50 gr			
Power, Start-up-Time:	7...34 V DC ; approx.. < 2 W @ 34 V, < 1 W @ 7 V; < 4 sec; reverse-voltage protection			

iMAR Navigation GmbH • Im Reihersbruch 3 • D-66386 St. Ingbert / Germany
Phone: +49-(0)-6894-9657-0 • Fax: +49-(0)-6894-9657-22
www.imar-navigation.de • sales@imar-navigation.de

[1] Systems with other (higher and lower) sensor performance available on request
[2] depends on applied dynamics (typical values)

Pioneering new technologies

Sensor-Technik Wiedemann GmbH
Mobile Controllers and Measurement Technologies

Technical data

Inclinometer and Gyroscope Sensor NGS2

www.sensor-technik.com

3-axis Inclinometer and Gyroscope Sensor

Combined sensor for measurement of angular velocity and inclination in 3 axes. The acceleration in each axis is also available.
The measured values are available on the CAN bus and optionally on three analog outputs, current or voltage.

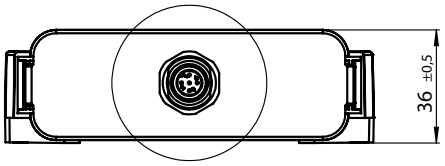
Technical data

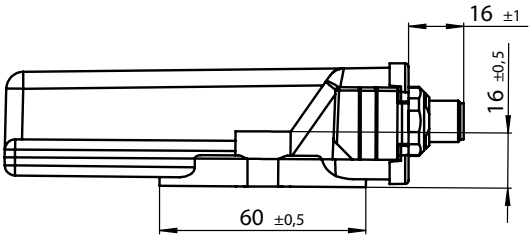
Gyroscope Sensor	
Measuring range	± 50 °/s
Band width (3 dB)	40 Hz
Resolution	0,25 °/s
Accuracy Offset (–40 °C...+85 °C)	± 0,5 °/s*
Accuracy Span (–40 °C...+85 °C)	± 0,5 % FS*
Non linearity	0,1 % FS
Influence acceleration	(0,1 °/s)/g

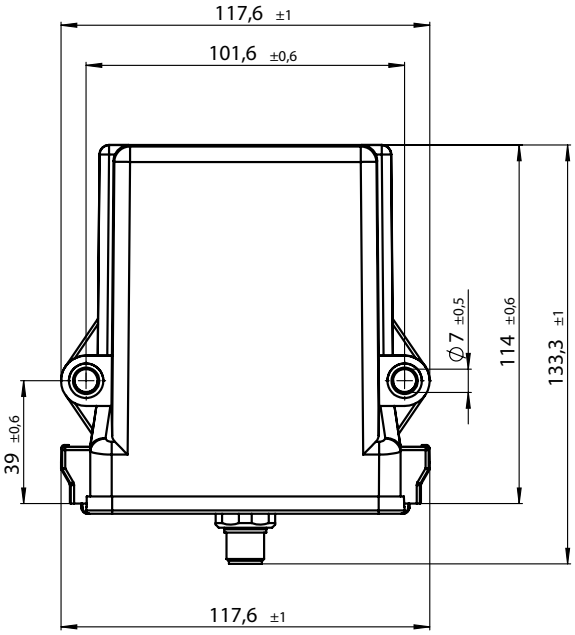
Inclinometer	
Measuring range	± 180 ° (analogue ± 90 °)
Band width (3dB)	15 Hz
Resolution	0,01 °
Accuracy (–40°C...+85°C)	± 1,5 ° (typ. ± 0,5 °)*

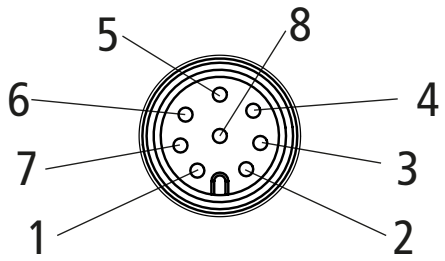
Collective data	
Output signal digital	CAN, baudrate 50 to 1000 kBit/s
Output signal analogue	0 ... 20 mA or 0 ... 10 V *refered to CANopen-Interface, analogue outputs can differ
CAN-Interface	CANopen
Temperature range	–40 °C ... +85 °C
Excitation voltage	9 … 36 VDC resp. 14 … 36 VDC (0 … 10 V voltage output)
Current consumption	120 mA @ 12 V / 60 mA @ 24 V
Connector	5-pole M12-plug (CANopen), 8-pole (CANopen and analogue output)
Protection class	IP67 / IP69k
EMV, mechanical and climatic requirements	According to automotive, agricultural and construction industry standards, CE-conformity
Chassis	PBT-GF30
Weight	approx. 250 g

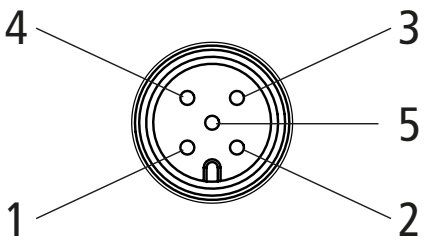
3-axis Inclinometer and Gyroscope Sensor











Pin assignment for 8-pole connector	
Pin	Signal
1	CAN_H, CAN+
2	V+, Supply
3	ANALOG_OUT1
4	ANALOG_OUT2
5	ANALOG_OUT3
6	ANALOG_GND
7	CAN_L, CAN-
8	GND, Ground

Pin assignment for 5-pole connector	
Pin	Signal
1	n.c.
2	CAN_V+, Supply
3	CAN_GND, Ground
4	CAN_H, CAN+
5	CAN_L, CAN-

REFERENCES

[1] H. Lasi, F. Peter, T. Feld, M. Hoffmann and H.-G. Kemper, "Industry 4.0," *BISE,* vol. 6, no. 4, pp. 239-242, 19 06 2014.

[2] J. Lee, B. Bagheri and H.-A. Kao, "A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems," *ScienceDirect,* pp. 18-23, 10 Decemeber 2014.

[3] T. Arampatzis, J. Lygeros and S. Manesis, "A Survey of Applications of Wireless Sensors and Wireless Sensor Networks," in *Intelligent Control, 2005. Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation* , 2005.

[4] R. Schmidt, M. Möhring, R.-C. Härting, C. Reichstein, P. Nuemaier and P. Jozinović, "Industry 4.0 - Potentials for Creating Smart Products: Empirical Research Results," 2015.

[5] L. D. Xu, W. He and S. Li, "Internet of Things in Industries: A Survey," *IEEE Transactions on Industrial Informatics,* vol. 10, no. 4, pp. 2233-2243, November 2014.

[6] Z. Bi, L. D. Xu and C. Wang, "Internet of Things for Enterprise Systems of Modern Manufacturing," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS,* vol. 10, no. 2, pp. 1537-1546, May 2014.

[7] M. S. Hossain and G. Muhammad, "Cloud-assisted Industrial Internet of Things (IIoT) – Enabled framework for health monitoring," *Computer Networks,* vol. 101, pp. 192-202, February 2016.

[8] J. A. G. Ibáñez, S. Zeadally and J. Contreras-Castillo, "IntegrAtIon chAllenges of IntellIgent trAnsportAtIon systems wIth connected VehIcle, cloud computIng, And Internet of thIngs technologIes," *IEEE Wireless Communications,* vol. 22, no. 6, pp. 122-128, December 2015.

[9] B. R. Upadhyaya and E. Eryurek, "Application of Neural Networks for Sensor Validation and Plant Monitoring," *Nuclear Technology,* vol. 97, pp. 170-176, February 1992.

[10] F. Tao, Y. Cheng, L. D. Xu, L. Zhang and B. H. Li, "CCIoT-CMfg: Cloud Computing and Internet of Things-Based Cloud Manufacturing Service System," *IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS,* vol. 10, no. 2, pp. 1435-1442, May 2014.

[11] D. Bradley, D. Russel, I. Ferguson, J. Isaacs, A. Macleod and R. White, "The Internet of things – the future or the end of mechatronics.," *Mechatronics,* vol. 27, pp. 57-74, 2015.

[12] W. Y. Wong, M. S. Wong and K. H. Lo, "Clinical applications of sensors for human posture and movement analysis: A review," *Prosthetics and Orthotics International,* vol. 31, no. 1, pp. 62-75, March 2007.

[13] V. M. Rohokale, N. R. Prasad and R. Prasad, "A Cooperative Internet of Things (IoT) for Rural Healthcare Monitoring and Control," in *Proceedings of the 2nd*

International Conference on Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), 2011.

[14] M. Napolitano, D. Windon and J. Casanova, "Kalman filters and neural-network schemes for sensor validation in flight control systems," *IEEE Transactions on Control Systems Technology,* vol. 6, no. 5, pp. 596-611, September 1998.

[15] Y. R. Rao, "Automatic Smart Parking System using Internet of Things (IOT)," *International Journal of Engineering Technology Science and Research,* vol. 4, no. 5, pp. 225-228, May 2017.

[16] H. O. Al-Sakran, "Intelligent Traffic Information System Based on Integration of Internet of Things and Agent Technology," *International Journal of Advanced Computer Science and Applications,* vol. 6, no. 2, pp. 37-43, 2015.

[17] A. McAfee and E. Brynjolfsson, "Big Data: The Management Revolution," *Harvard Business Review,* pp. 1-9, October 2012.

[18] A. Gandomi and M. Haider, "Beyond the hype: Big data concepts, methods, and analytics," *International Journal of Information Management,* vol. 35, no. 2, pp. 137-144, April 2015.

[19] W. Elmenreich, "An Introduction to Sensor Fusion," Institut fuer Technische Informatik Vienna University of Technology, Austria, Vienna, Austria, 2002.

[20] Q. Gan and C. J. Harris, "Comparison of Two Measurement Fusion Methods for Kalman-Filter-Based Multisensor Data Fusion," *IEEE Transactions on Aerospace and Electronic Systems,* vol. 37, no. 1, pp. 273-280, January 2001.

[21] O. Basir and X. Yuan, "Engine fault diagnosis based on multi-sensor information fusion using Dempster-Shafer evidence theory," *Information Fusion,* vol. 8, no. 4, pp. 379-386, October 2007.

[22] P. Pinheiro and P. Lima, "Bayesian Sensor Fusion for Cooperative Object Localization and World Modeling," in *Proc. 8th Conf. IAS*, Amsterdam, 2004.

[23] N. Ghosh, Y. B. Ravi, A. Patra, S. Mukhopadhyay, S. Paul, A. R. Mohanty and A. B. Chattopadhyay, "Estimation of tool wear during CNC milling using neural network-based sensor fusion," *Mechanical Systems and Signal Processing,* vol. 21, no. 1, pp. 466-479, 2007.

[24] Crown Elevator & Lift Company, "Hydraulic Elevators," [Online]. Available: https://www.crownelevator.com/hydraulic-elevators/. [Accessed 7 May 2018].

[25] T. Bayes, "An Essay towards solving a Problem in the Doctrine of Chances," 1763.

[26] A. Singhal and C. Brown, "Dynamic Bayes Net Approach to Multimodal Sensor Fusion," University of Rochester, Rochester, NY, 1997.

[27] J. Dezert, P. Wang and A. Tchamova, "On the Validity of Dempster-Shafer Theory," in *International Conference on Information Fusion*, Singapore, 2012.

[28] J. Zhou, L. Liu, J. Guo and L. Sun, "Multisensor Data Fusion for Water Quality Evaluation Using Dempster-Shafer Evidence Theory," *International Journal of Distributed Sensor Networks,* vol. 2013, 2013.

[29] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 3. State and parameter estimation," *Journal of Power Sources,* vol. 134, pp. 277-292, 2004.

[30] E. A. Wan, "The Unscented Kalman Filter for Nonlinear Estimation," Oregon Graduate Institute of Science & Technology, Beaverton, Oregon, 2000.

[31] T. A. Wenzel, K. J. Burnham, M. V. Blundell and R. A. Williams, "Dual extended Kalman filter for vehicle state and parameter estimation," *Vehicle System Dynamics,* vol. 44, no. 2, pp. 153-171, 2006.

[32] A. F. Bumb and W. Kern, "A simple dual ascent algorithm for the multilevel facility location problem," University of Twente, Enschede, 2001.

[33] E. A. Wan and A. T. Nelson, "Neural dual extended Kalman filtering: applications in speech enhancement and monaural blind signal separation," Department of Electrical Engineering, Oregon Graduate Institute, Portland, 1997.

[34] A. Hać, Wireless Sensor Network Designs, Honolulu, Hawaii: John Wiley and Sons, Ltd, 2003, pp. 31-98, 141-231.