INFRASTRUCTURE ENABLED AUTONOMY ACTING AS AN INTELLIGENT TRANSPORTATION SYSTEM FOR AUTONOMOUS CARS

A Thesis

by

AUSTIN JESS BURCH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Srikanth Saripalli
Committee Members,	Swaminathan Gopalswamy
	Paul Carlson
Head of Department,	Andreas Polycarpou

May 2018

Major Subject: Mechanical Engineering

ABSTRACT

Autonomous cars have the ability to increase safety, efficiency, and speed of travel. Yet many see a point at which stand alone autonomous agents populate an area too densely, creating increased risk - particularly when each agent is operating and making decisions on its own and in its own self-interest. The problem at hand then becomes how to best implement and scale this new technology and structure in such a way that it can keep pace with a rapidly changing world, benefitting not just individuals, but societies. This research approaches the challenge by developing an intelligent transportation system that relies on an infrastructure. The solution lies in the removal of sensing and high computational tasks from the vehicles, allowing static ground stations with multi sensor-sensing packs to sense the surrounding environment and direct the vehicles safely from start to goal. On a high level, the Infrastructure Enabled Autonomy system (IEA) uses less hardware, bandwidth, energy, and money to maintain a controlled environment for a vehicle to operate when in highly congested environments. Through the development of background detection algorithms, this research has shown the advantage of static MSSPs analyzing the same environment over time, and carrying an increased reliability from fewer unknowns about the area of interest. It was determined through testing that wireless commands can sufficiently operate a vehicle in a limited agent environment, and do not bottleneck the system. The horizontal trial outcome illustrated that a switching MSSP state of the IEA system showed similar loop time, but a greatly increased standard deviation. However, after performing a t-test with a 95 percent confidence interval, the static and switching MSSP state trials were not significantly different. The final testing quantified the cross track error. For a straight path, the vehicle being controlled by the IEA system had a cross track error less than 12 centimeters, meaning between the controller, network lag, and pixel error, the system was robust enough to generate stable control of the vehicle with minimal error.

DEDICATION

To the many talented, intelligent and patient role models who have taken their time to help me over the years.

ACKNOWLEDGMENTS

I would like to thank the Texas A&M University for providing a positive educational experience throughout my time here. I would also like to extend thanks to my research advisor, Dr. Saripalli, and my entire committee for their guidance and time put into this research. Finally, I would like to thank the Unmanned Systems Lab and CAST Team members at Texas A&M for bringing an enthusiam to learn and grow each day.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a thesis committee consisting of Dr. Saripalli and Dr. Gopalswamy of the Department of Mechanical Engineering, and Dr. Carlson of the Department of Civil Engineering.

For scaled testing purposes, a smaller, secondary Muti-sensor Sensing Unit was constructed and assembled by George Chustz.

All other work conducted for the thesis was completed by the student independently.

Funding Sources

This graduate study was supported by Texas A&M's Department of Mechanical Engineering funding for the Unmanned Systems Lab.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION AND OVERVIEW

1.1 Introduction

This thesis begins with the introduction of a problem that comes along with immersing technology in todays car industry in the form of autonomous functionalities. Artificial intelligence and autonomous vehicles are proving to be the next generation of transportation. Currently companies are releasing vehicles with enhanced technologies such as collision avoidance, adaptive cruise control, and lane correction. While these advances aid the human driver and make transportation faster, safer, and more efficient, there remains a tremendous capacity for technological growth in the industry.

This thesis paper aims to discuss the use of an Infrastructure Enabled Autonomy (IEA) approach to an Intelligent Transportation System (ITS). The solution lies in the removal of sensing and high computational tasks from the vehicles, allowing static ground stations with multi sensor-sensing packs (MSSPs) to sense the surrounding environment and direct the vehicles from start to goal. This approach allows for a distribution or computational power, and offloaded expensive hardware from each individual vehicle to a static set of MSSPs on a connected network that provides the vehicles in the environments with the wireless commands to achieve safe transportation.

This introductory section defends the motivation to conduct research pertaining to the use of autonomy, and more specifically, the need for an infrastructure based autonomous vehicle platform. The Introduction then objectively reviews the related work and a literature review. Finally, this section ends with an overview of the remaining paper, to understand the fluidity of moving from one section to the next.

1.1.1 Motivation

Currently, the race is on to produce a fully autonomous car. Autonomous cars have many positive benefits, yet many see a point at which autonomous agents populate an area too densely, creating increased risk - particularly when each agent is operating and making decisions on its

own and in its own self-interest. The problem at hand then becomes how to best implement and scale this new technology and structure in such a way that it can keep pace with a rapidly changing world, benefiting not just individuals.

A stand alone autonomous car requires an expensive array of sensors and computational hardware. Each vehicle must be fully outfitted with the sensors needed to analyze the vehicle's environment, the computational power to process, fuse, and provide intelligent decisions for the sensor feeds on each vehicle, and also the power to drive all of these electronics. A stand alone autonomous car utilizes the sensors on the vehicle to know its surrounding environment. Relying on autonomous cars proprioception has the potential to limit its FOV, which in turn may require more cameras, which utilize more computation and power. Also, the viewpoint of a ground vehicle is not always the most optimal perspective, and in the presence of large obstacles or other vehicles, can cause blind spots which hinder path planning and require shorter reaction times.

Different from a standalone autonomous car, Intelligent Transportation Systems (ITS) have various approaches to implementing connected vehicle models which utilize wireless communications to relay information for decision making. Varying ITS configurations can include combinations of vehicle to vehicle communications, infrastructure communications, and infrastructure to vehicle communications. A stand alone autonomous car typically operates on a confined network within the vehicle, which may not utilize all advantageous functionalities of connected vehicles. The literature review section will focus on many of these ideas, and compare and contrast the Infrastructure Enabled Autonomy approach to the respective work. The next section will build the foundation of the problem at hand, and outline the benefits and goals of this research.

1.1.2 Development of Problem and Problem Statement

An Infrastructure Enabled Autonomy approach to operating vehicles autonomously is defined uniquely as a network of distributed sensing units working cohesively to detect and plan for the vehicles in the IEAs environment. This approach differs greatly from the typical autonomous car that carries its own sensing and computational hardware. This approach to an Intelligent Transportation System not only sets the foundation for autonomous operation, but also pushes cars to

operate under a connected vehicles model.

Areas densely populated with autonomous vehicles can more efficiently be controlled through a limited number of statically mounted sensing units working cohesively as Infrastructure Enabled Autonomy (IEA). This critical mass happens when the number of Multi-sensing Sensor Packs (MSSPs) required to thoroughly visualize the activity in an environment falls below the number of vehicles operating within the environment. Confined spaces with high levels of traffic, such as parking lots, or even urban areas would best represent these situations.

Another superior application of an IEA approach to a stand alone autonomous platform is when in use for GPS denied environments. Technology has become increasingly reliant on GPS for many applications that require localization. However, the new implementation of autonomy to the automotive industry requires a high level of accuracy to ensure safety for its passengers. It is currently very difficult and expensive to achieve fast, accurate, and reliable GPS data. Furthermore, GPS need 'visibility' of satellites, and the more the better. This presents a critical issues in applications underground, obstructed by large features like mountains, buildings, and in particular, metal structures such as bridges. Therefore, need for a local frame transportation system that does not rely on GPS enables these autonomous cars in more use cases, and provides accurate and reliable localization in GPS denied and unstable environments.

A large portion on the economy's delivery architecture relies on the use of many vehicles (specifically large rigs carrying heavy payloads) to travel repetitive routes for long distances. Along with an infrastructure, enabling autonomous functionalities, specified IEA lanes may reduce the complexity of autonomous vehicle's environment, and increase safety of the vehicles within it. Furthermore, there would be a large reduction in power consumed on board the vehicle over the long repetitive drives due to the offloading of sensing and computing.

Beyond reducing the complexity of an environment for more reliable decision making, analyzing overlapping and sensing static environments is a simpler task, that has advantages for increased redundancy, reliability, and accuracy. Because different nodes of sensing units can combine their respective perceptions, vehicles now become capable of both seeing and planning around blind

spots and turns and adapting for obstacles at further distances. Multiple MSSPs, in overlapping areas, can collaboratively decide the best MSSP command to be sent to the vehicle controller based on FOV, detection characteristics, and a priori knowledge. This confidence level and selector functionality allows for a more robust localization. This can be beneficial in difficult road layouts, where more sensing and FOV is desired. Furthermore, the daisy chaining of MSSPs vision together allows for the construction of a nodal network of connected MSSPs that collectively generate a map of the environment. These MSSPs can each determine the conditions of their respective environment, communicating the obstacles and vehicle positions.

Due to the simplification of the requirements for developing the basis of what consists of an autonomous vehicle, commonly found drive-by-wire vehicles with minimal firmware can simply connect to the IEA network to operate autonomously in the IEA system. This lack of hardware adjustment greatly reduces the push back of the implementation of new technology. Using simple drive by wire setups allow for efficient developing, quick testing, and a platform for others to collaborate on at a reasonable cost.

By utilizing a decentralized, distributed sensing and control approach, this research presents the situation where an IEA system can increase reliability, efficiency, and most importantly, safety. This platform has the potential to integrate all autonomous agents in a particular area into a connected vehicle model. When agents operate on the same network, there is a consistent dissemination from MSSPs across all vehicles. Intelligent Transportation Systems aim to implement algorithms to increase predictability, which can lead to a safer transportation platform. Not only can vehicles be lead remotely by the network of MSSPs, but the autonomous agents, as a whole, can be led in such a way as to minimize risks, in turn increasing safety.

The main objective of this research is to develop a platform in which distributed sensing and computing can successfully navigate cars autonomously, through use of a network of statically mounted Multi-Sensor Sensing Packs (MSSPs). Furthermore, this research hopes to develop methods for transmitting data, and determining vehicle control commands for a network of units operating cohesively for safe and reliable transportation.

1.1.3 Research Requirements

This research aims to investigate and test a proof of concept, using static sensors and a network of distributed computing in the environment can be used to control a vehicle safely and reliably. In order prove or disprove the IEA hypothesis, a set of research requirements guided the overarching goals to achieve defensible outcomes.

The foundational hardware utilized for this IEA system is the Multi-sensing Sensor Pack (MSSP). This unit must contain the sensors for accurate data collection in order to develop a virtual environment in which the CPU can make intelligent decisions for controlling the vehicle. Furthermore, the CPU making these decisions is required to operate at speed that enable sufficient data collection and processing in order to control the autonomous cars. The large components of the CPU's processing is due to detection, tracking, and classification tasks required for the computer vision localization of the vehicles. Another high computational task can be attributed to the distributed path planning - too slow of a processor would induce too low of a vehicle control frequency for stable control, or too slow path planning such that the vehicle could not react to obstacles in its path.

In order to address the network questions pertaining to the operation of an infrastructure based approach to autonomy, the system developed should consist of at least two static sensing units. This insures there is a fused environment where the MSSPs can collaborate to provide redundant localization in overlapping areas, and extend the line of sight or look ahead distance of a vehicle's sensing capabilities. For the vehicle control, a distributed network will test the speed and accuracy of a hand off from one MSSP to another. It is important the network bandwidth can support these functions including sending control commands to the vehicle, and sharing processed sensor data to build and analyze a fused environment for the MSSPs to plan and operate the vehicles within. This successful hand off is both time dependent and accuracy dependent. Another requirement of this research's system is that the IEA should be able to intelligently used shared control of the vehicle. This means both the time and accuracy requirements must be met and reliable for consistent use. The IEA architecture should consist of distributed computing, each MSSP unit

containing its own computing, and performing paralleled functionalities when it comes to computer vision, path planning, and vehicle control.

This system also requires a drive by wire vehicle. Electronic signals collected wirelessly from the MSSP will control the steering and throttle of the vehicle. Beyond being outfitted with a drive by wire setup, this demands that the vehicle wireless connectivity, and a small processor interpreting the wireless commands to electrical signals driving the drive by wire actuators. At this point, a vehicle has the functionalities required to operate on IEA network.

This requires will ensure a level of assessment for the Infrastructure Enables Autonomy system. Once this system is operational, the core can be scaled and expanded to contain any number of MSSPs operating cohesively.

1.1.4 Contributions

Throughout the process of this research, and the development of the IEA system, multiple operational units and methods have originated from my individual contributions.

- Designed, tested and iterated on the Multi-Sensor Sensing Pack (MSSP).
- Developed architecture for acquiring, processing, and performing functions on sensor data in real-time.
- Implemented path planning and tracking algorithms, computer vision functionalities, and an optimal architecture for data workflow.
- Developed confidence building and MSSP selection logic for fused control.
- Designed a multi-unit system capable of local and remote data fusion, distributed computing, and confidence building.
- Implemented proof of concept for multi MSSP and vehicle testing environment.

1.2 Related Work

This section will provide an in depth discussion on the relevant work to this research. Starting with a detailed literature review, various papers will present different approaches to managing numerous autonomous vehicles. The discussion on related work will then move to an infrastructure specific topic on the spectrum of autonomy in such systems.

1.2.1 Literature Review

Addressing this issue of autonomy integration with the current makeup of human drivers is vital to the beginnings of fully autonomous systems. Even more powerful than the proficiency of technology, the acceptance into society through trust is just as important in the introduction of a new technology to the public. Autonomous systems must operate in such a way to provide comfort and protection to the passenger, as well as optimizing their goals such as traffic mitigation and route efficiency. The papers in this literature review will be analyzed based on method to solve the traffic control problem for fully autonomous systems, the technical feasibility of integration into a real world use case in the near future, and its parallel to an IEA approach.

The discussion on the varying approaches to relevant intelligent transportation system approaches will be segmented by paper to provide a simple and structured literature review process.

A. Prioritized Path Planning

Plessen in Multi-automated vehicle coordination using decoupled prioritized path planning for multi-lane one- and bi-directional traffic flow control aims to promote a multi-agent coordination scheme with communications between each car. Plessen states that a car-2-car infrastructure communication and subsequent real-time coordination is perceived to be the main enabling technology for maximized road safety, lane throughput, and congestion avoidance due to its anticipative nature and potential for deterministic traffic flow planning. This approach proposes an discretized algorithm that looks at both one-directional and bi-directional traffic flows. For each case, current vehicle path and goal deviations are analyzed for each vehicle, then each vehicle independently checks their own trajectory with the a priori knowledge of the surrounding vehicles. This paper

discusses the advantage of increased knowledge base, through sharing data with other cars. This idea is similar to the IEA approach providing a different, and in many cases, larger perspective of the surrounding environment. While this paper focuses on autonomous platooning, the approach is confined to vehicle to vehicle communication. This varies from IEA due to the lack of remote sensing and computing. Figure 1.1 illustrates one aspect of the coordination between vehicles through wireless communication [1].

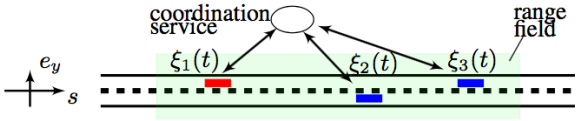


Figure 1.1: Diagram of Plessen's coordination service method.

B. Pheromone Model Traffic Control

The Pheromone model, presented by Masutani, is a predictive traffic model. This implementation is in reference to a larger scale than of Park and Lees infrastructure approach, as it looks at the prediction and avoidance of highly congested areas, as opposed to conflict resolution at highly congested intersections. This approach models social insects that excrete pheromones to communicate in a decentralized fashion. This application will make use of pheromones as releasing a electronic density cloud proportional to the congestion in that particular area. Parameters such as density, directional velocity, and pheromones evaporations allow for the dynamic process to evolve with time. This approach, was simulated using real world data, and saw a significant increase in the shortest route selection by the autonomous agents. This model is attractive as a feasible implementation, because the pheromone model does not conflict with adding humans to the road. While the main implementation would be for use in vehicle to vehicle communications, this paper notes that other agents could tap into this network to provide a larger set of data. Similar to the IEA approach, this model attempts to build a map of vehicle activity and attempts to provide intelligence with the

intelligence to reach its goal along an optimal path. However, there are much greater differences such as the lack of an infrastructure to aid in the computing, much less a static nodal network of sensing units sending commands to the environment's vehicles [2].

C. Information Dissemination

In attempt to address the next step in ITS systems, Hager in Vehicular Ad Hoc networks: Multi-hop information dissemination in an urban scenario discusses the approach to disseminate information. Vehicular ad hoc networks (VANETs) requires communication between individual vehicles, not just a broadcasting as previously used in digital audio broadcasts (DAB) or satellite radio systems. Hager proposes a VANET delay based solution, where each node waits a specific time until it tries to forward information. If a node receives the information during this waiting period from another node, it will terminate the forwarding process. This approach frees up bandwidth and ensures the minimization of redundant communication, which is vital in multi-agent systems. This process was tested in simulation on multiple events of interest, and the efficiency of communication and information dissemination amongst a grouping of autonomous vehicles was analyzed. Hager suggests that no one multi-hop broadcast protocol is capable of reacting efficiently for all scenarios, and a more robust even classification module would need to be developed in order for implementation of this model. This connected vehicle model is similar to the IEA approach such that it utilized a wireless network to manage the environment data through use of a multi-agent system. Hager states that the main challenges of this VANET approach is the, determination of the corresponding protocol parameters with respect to the scenario and the decision process performed by the vehicles to detect these scenarios. This paper provides further insight on the discussion that for a successful intelligent traffic mitigation approach will require a variety of modules, preparing the autonomous agents for as many different events that they may be subjected to. Furthermore, we see that a single infrastructural approach cannot fully equip the autonomous agent with all the tools it need to react to each and every spontaneous situation. This is a direct use case for the IEA sensing units. Their job would enable the environment processing for autonomous control within the environment [3].

D. Dynamic Traffic Control Systems

Wietholt presents an approach to dynamics traffic control through use of infrastructure elements. Dynamic Traffic Control Systems attempt to harmonize traffic flow. In this application, variable signs that are posted on roadways give indications of dynamic speed limits, no passing for heavy vehicles and warning of congestion, work zones or weather conditions. The main functionality, the dynamic speed limit, is varied according to traffic flow, mean speed, or traffic density on the motorway. This indication system provides a low tech way to implement traffic aid to human drivers. Both autonomous and humans drivers may access and interpret this data in the same way. This is an advantageous approach, because it is an platform that can allow different algorithms to be tested and output through the manipulation of lanes through these dynamic indicators. However, due to the low level of intelligence, the use functionalities are limited. The lack of sensing does not enable the Dynamic Traffic Control System to aid in real time, low level control it the vehicles. This papers focus is on the grander scaled of reducing and predicting traffic [4].

E. IEEE 802.11 DCF/PCF Mechanisms at Intersections

Intelligent Traffic Control Based on IEEE 802.11 DCF/PCF Mechanisms at Intersections by Park and Lee discuss an infrastructure approach to traffic control systems. This approach aims to eliminate the need of traffic lights through access points available for autonomous cars approaching intersections. This algorithm attempts to minimize the congestion of lanes at intersections, by giving longer lines priority. Two basic components make up this approach, the contention period, and non-contention period. The contention free period involves the congested lanes determining which lanes have the most congestion, and therefore receives priority. The contention period is the opposite, but provide a unique workout around. Because it is unfair to make a single car wait until it is the busiest lane, the contention period ensures that lane overlap, allowing some flow in all directions. Park and Lee discuss that this balance of contention free periods lead to an optimization problem of fairness with increase contention priority, and efficiency with increased non-contention priority. While this algorithm shows vast improvement to the unintelligent timing systems around

us today, this approach requires fully autonomous agents. This approach is similar to the IEA intelligent transportation system in the fact that it uses an infrastructure to process localization data in attempt to plan and coordinate vehicles within the environment. The differences between the IEA approach and this paper are that Park relies on the vehicle to communicate information to the infrastructure via an access point. While both use a connected network, the IEA system is able to localize vehicles within its own environment without receiving data from the vehicles. Figures 1.2 and 1.3 illustrate both the complexity and flow patterns in typical traffic, as well as would be in the proposed ITS system in this paper [5].

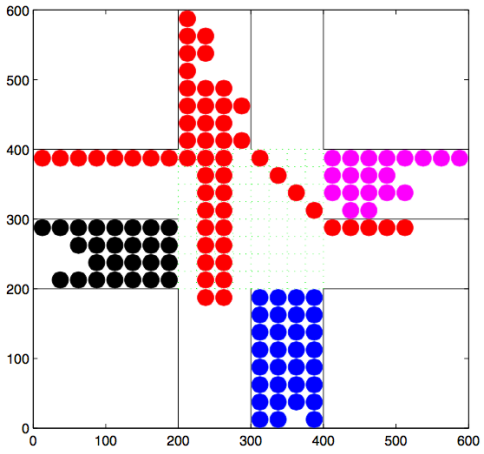


Figure 1.2: Typical traffic flow pattern in current human driver applications.

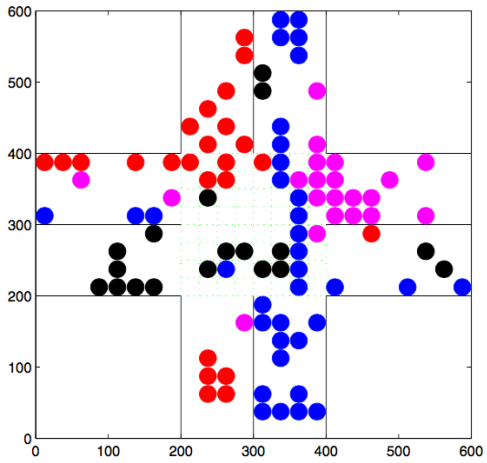


Figure 1.3: Potential autonomous flow pattern through use of intelligent traffic control.

F. Internet of Vehicles

Gerla in Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds discusses in depth the advancement in ITS and their capabilities due to new technologies in autonomous car functionalities. Gerla suggests that the next step is to integrate these intelligent cars into an 'Intelligent Vehicle Grid'. This network of vehicles would share storage, intelligence, learning, and other autonomous benefits. This Internet of vehicles is similar to IEA such that it aims to use an infrastructure based network to pass information from one unit to the next. This paper discusses an implementation where vehicles, road side units, and servers all communicate with each other. While this research investigates the infrastructure taking command of all sensing, processing, and control, this approach attempts to solve the problem in a similar way to IEA. This approach may fall onto the Spectrum of Autonomy discussed in the following section. While it does vary from this research architecture, less infrastructure dependent approaches, are similar to Gerlas approach. Figure 1.4 below shows how connected various units would communicated in an environments utilizing Gerla's internet of vehicles [6].

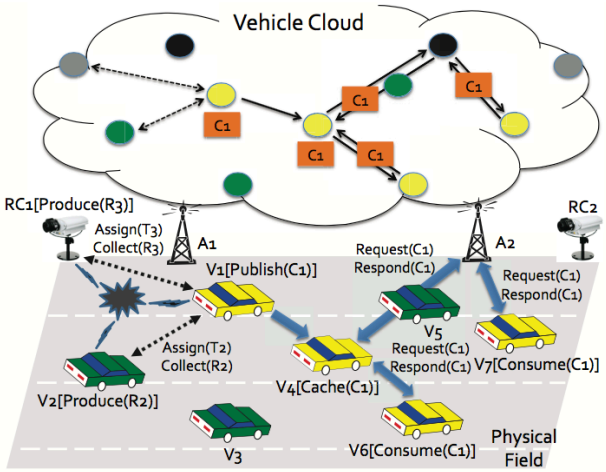
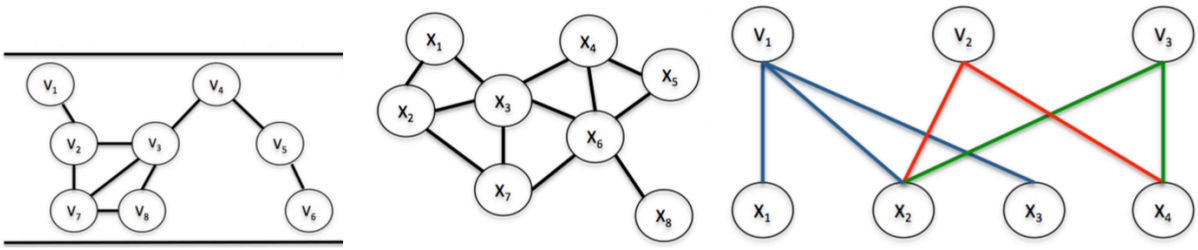


Figure 1.4: Gerla's connected network diagram of cars and road side units.

G. Distributed Ranking of Popular Smart Vehicles

Khan in An Information-Centric Autonomous and Distributed Ranking of Popular Smart Vehicles defends the ability of autonomous vehicle to increase road intelligence and aim in decision making for safer travel. This paper utilizes the autonomous car sensing platform to collect and share environment data with surrounding vehicles over a distributed VANET system. This approach to connected vehicles lacks an infrastructure, and relies on vehicle to vehicle communication. This approach, however, does discuss the problem of scaled network overloading, and the importance of consistent wireless communication. While this is a connected vehicle model, it lacks the remote sensing and computing, and instead relies on the hardware residing on the vehicle. Figure 1.5 is a diagram of a potential nodal network of connected vehicles, and their hierarchy of communication.



(a) Vehicuar graph: $G(\mathbb{V}(t), \mathbb{E}^v(t))$ (b) Location graph $G(\mathbb{X}, \mathbb{E}^x)$ (c) Information Association Graph $G(\mathbb{V}, \mathbb{X}, \mathbb{E})$

Figure 1.5: Vehicular graph for vehicle to vehicle communication and connected vehicle dissemination.

H. Traffic Models for Connected Cars

Pawe's Traffic Models for Self-driving Connected Cars states the use cases of connected vehicles, whether it be vehicles communicating with one another, or an infrastructure component. The scope of this paper is to simulate traffic intersections, and specific vehicle congested scenarios. This paper is similar to IEA in the way it aims to collaborate multiple data streams to plan and predict the safest route for vehicles within the environment. In this model, the vehicle communicates and clears commands before performing maneuvers. While IEA also sends instructions to the vehicle, the level of reliance on the infrastructure is minimized, the computing and sensing remain on the vehicles to aid in perception. This paper also does not expand beyond simulation, nor utilizing remote sensing units not only for perception, but for planning and control. [7]

I. D-S Theory Based Credibility Map for Perception

Zhao's paper shows close parallel to the IEA sensing component. Zhao utilizes road side units to gain perception on the environment, and communicate to the vehicles. This approach, different from the IEA approach, uses V2I (vehicle to infrastructure) communication. The on board GPS of the vehicle is responsible for providing the road side units with localization information. The road side unit consists of sensors enabling it to detect vehicle within its FOV, similar to the IEAs sensing unit. The focus of this paper is to develop an accurate fusion method to handle uncertainties from

the two localization techniques. Zhao's approach ends here. The IEA system not only localized vehicle and obstacle in its environment, but continues with path planning, and communicating commands to the vehicle. This paper showed a similar use case to that of the IEA project, but a much smaller scope. The IEA system aims to encompass the full stack solution from detecting for outputting commands. Figures 1.6 shows the components that make up the V2I, and Figure 1.7 shows a credibility map identifying the location of vehicles within the scene [8].

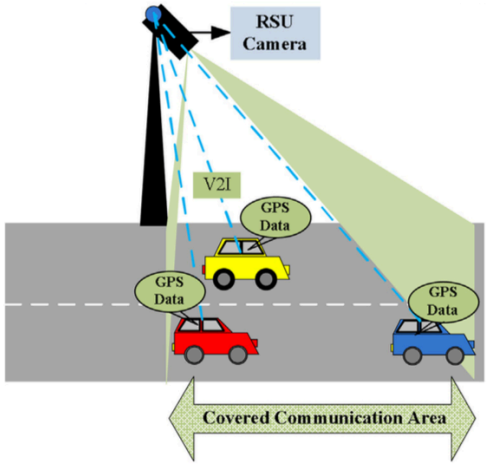


Figure 1.6: RSU and V2I diagram.

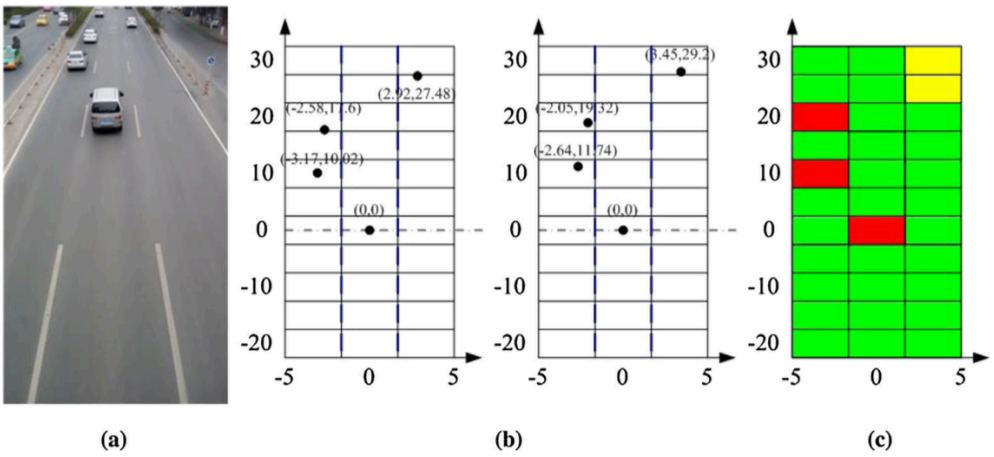


Figure 1.7: Zhao's vehicle to infrastructure outputs displaying vehicle localization.

J. Parked Cars As Virtual Network Infrastructure

Hagenauer in Parked Cars As Virtual Network Infrastructure, utilizes an infrastructure of road side units to communicate with a number of vehicles residing in parking lots. The use case described is a connected vehicle network to share data and act as network gateways for the surrounding area. While this paper introduces the challenges of scaling vehicle to infrastructure or vehicle to vehicle communications, there is no end goal of controlling autonomy from the communication platform. Furthermore, the roadside units are merely data centers for the vehicles to interact with, they do not provide the processing and intelligence the IEA system has to offer. [9]

K. Vehicle-to-infrastructure Communication over Multi-tier Heterogeneous Networks

Ndashimye and others in Vehicle-to-infrastructure Communication over Multi-tier Heterogeneous Networks, explores another intelligent transportation system that takes advantage of the connected vehicles approach. Utilizing and infrastructure and wireless communications, this paper focuses on the Vehicle-to-infrastructure (V2I) communication with use of multi-tier networks. While this paper does expand beyond the idea of stand alone autonomous vehicles, it relies on the infrastructure for data collection and server processing. This differs from IEA due to the lack on sensing and distributive computing. This paper does not utilize a sensing infrastructure, but uses

the infrastructure as a wireless hub for vehicles to be connected on. [10]

1.2.2 Spectrum of Autonomy

Moving on from a literature review of relevant work, this section describes the Infrastructure Enabled Autonomy (IEA) system as one end of a spectrum from shared to stand alone autonomy. One of the most interesting aspects of an IEA approach is how much sensing and computing is removed from the vehicle, and placed under the responsibility of the IEA network. At one end of the spectrum, stand alone autonomous cars carry a variety of sensors, and a large amount of computational processing power. Not only are these both heavy and power consuming, they are very expensive. The IEA system sits at the other end of the spectrum. All that the IEA system requires outside of the MSSPs is a drive-by-wire vehicle with wireless connectivity. Because the MSSPs are statically distributed, the coverage can be optimized for the least amount of hardware while maintaining successful tracking of objects in the environment. The impact of this approach is the adjustment of responsibilities including control, liability, and cost onto the infrastructure rather than the car manufacturer.

Texas A&M is also pursuing a middle ground, where the IEA is utilized as a service alongside stand alone autonomy to aid the vehicle in localization. This scenario requires full sets of hardware on both sensing agents, the stand alone vehicle and the IEA system. Each system publishes localization data to the vehicle, and the vehicle is responsible for interpreting, fusing, and selecting a position to act on. This system aims to be expanded to encompass obstacle detection and other autonomous functionalities [11]. Currently this approach's framework is being implemented in simulation. A 'Distributed Hybrid Hardware-In-the-Loop Simulation framework for Infrastructure Enabled Autonomy' is being developed in attempt to enable a platform for larger scale simulation environments to run such a system. MSSP and autonomous vehicle processing are done in a distributed hardware architecture, and communicated wirelessly between them. This simulation environment will expedite development and testing for a large integrated system such as IEA [12].

1.3 Overview

This remainder of this paper will detail this research's specifics from the development of the operations system, to a discussion of testing outcomes. This IEA development focuses on the proof of concept for real world implementations. Developed around distributive networks or nodes handling the sensing and computing, these areas under most speculation will be the shared control between MSSPs, the wireless control commands delivered to the vehicle for real time control, and the accuracy of the MSSP's fused sensing network. This introductory section ends with a concise statement on the scope of this research, and a short overview of the remaining chapters.

1.3.1 Scope of Research

The IEA system has the capacity to be scaled up to a huge network of connected systems, vehicles, and computing parallels. While there are explicit areas of the control architecture that are necessary for an operational system, there are various methods of implementation that each have respective advantages and disadvantages. The descriptions of the system architecture and methods will address the down selection process - however, in order to validate design decisions, the scope of this research was setup addressing the project time line of one year. The last chapter will discuss future work, and will focus on the capacity for increased accuracy and robustness.

In order to ensure a fully operational system, the hardware setup and requirements for sensing, computing, and network capabilities all had to be met. This research utilizes two MSSPs operating together on a single network with the vehicle. This nodal communication could be expanded to any number of MSSPs. Within each MSSP, and beginning with the development of a single-unit system, this research focused on the real time implementation of a plethora of algorithms including path planning algorithms, computer vision functionalities, and an architecture and logic methods for optimal data process flow. The implementation of a multi-unit system focuses on local and remote data fusion, network management, and confidence building. These are the key components of an operational IEA system, and define the scope of this research.

1.3.2 Document Overview

This paper is divided into five chapters, the Introduction and Overview, System Methods and Description, Results and Discussion, and Summary and Conclusions. The introduction established the basis upon which this research was derived. The IEA system was discussed in a broader context of autonomous vehicles and the potential benefits behind such a system. The System Methods and Description will breakdown the development and design of this system within the scope of this research. This section will focus on the steps taken toward an operational IEA system capable of further testing for analysis. The Results and Discussion section will develop metrics for success of such a system, describe the testing process and outcomes. Finally, the Summary and Conclusions will correlate this proof of concept back to the large scale implementation discussion, and deliver suggestions, future work, and conclusions on how the system operates, can be improved, and can be best organized for success in a real world implementation. The back matter will conclude this report's content.

2. SYSTEM METHODS & DESCRIPTION

This section first presents the selected methodologies utilized for the implementation and design of the IEA system.

In order to construct a DAQ unit capable of capturing and processing all the required streams of data, a large processing unit is necessary. The Multi-Sensor Sensing Pack contains a computer, Visual Spectrum Camera, Thermal Camera, LIDAR, GPS, and battery. The computer system is run on Ubuntu and uses the ROS framework. For the multi-unit testing, a simpler MSSP that contained only a Visual Spectrum Camera for sensing was designed. The housing for both units were designed in CAD software, and 3D Printed. Each MSSP is mounted to a tripod. In order to determine the orientation, and distortion of each camera, a checkerboard is used on the ground plane of the environment to align the MSSPs projected view. The drive-by-wire system required for operation with the IEA system is an identical architecture on both the small scale RC car, and the full sized car. Aside from the drive-by-wire functionalities, each vehicle consists of a single node connecting to the network, and communicating steering and throttle commands to the controller via MavLink. These few items span all that is required to operate within the IEA framework. A visualization of the IEA implementation can be seen below in Figure 2.1.

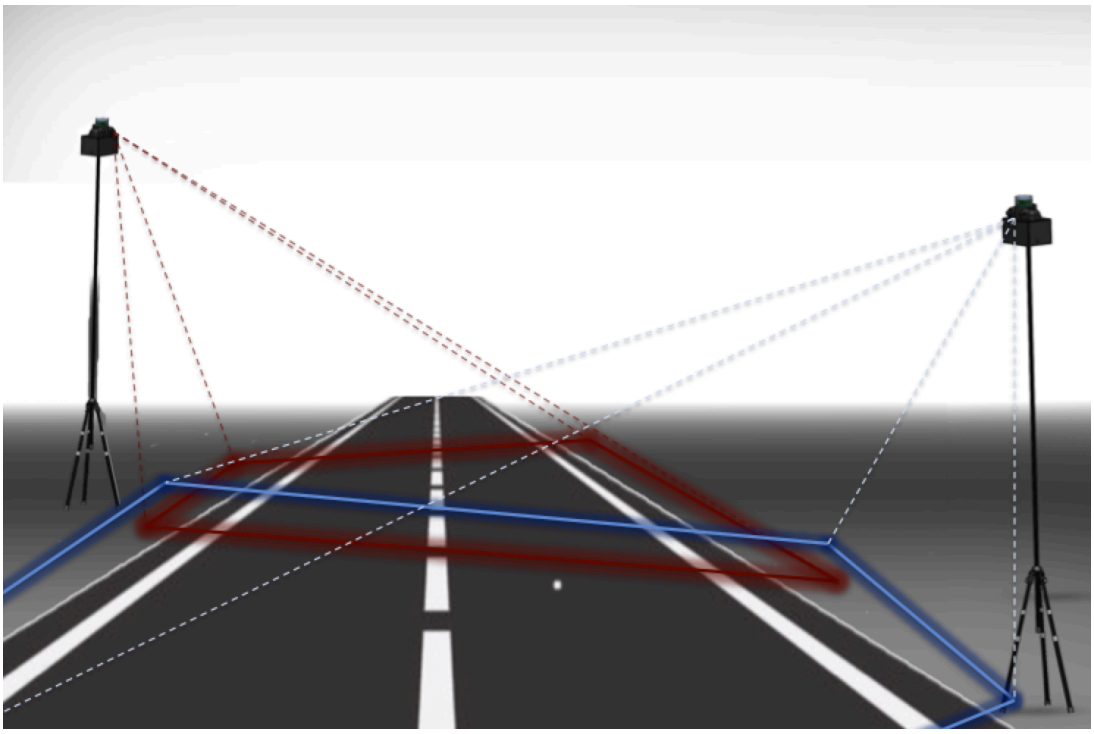


Figure 2.1: Scene detailing an IEA system with two MSSPs with overlapping perceptions.

Within each MSSP, many vision processing functionalities are being developed in order for the MSSP to be able to detect objects. In order to complete this task, this research utilizes the advantages of each MSSP remaining static. One avenue of detection algorithms is based on change detection of a background environment. Since any one particular MSSP is constrained to the same environment day in and day out, this seems like the most reliable approach. Once detected, the objects need to be classified as vehicles or as obstacles. These two lump sum categories are obviously simplified, but meet the needs and scope of this research topic. In order to successfully classify each object, color based segmentation and filtering is used. This is a simpler, but robust approach. While it may need to be tuned, the consistent nature of the IEA setting swayed positively toward this implementation. These functionalities make up the vision processing of the MSSP.

With a simple User Interface, a user can select a goal on the map of the environment, and arm the vehicle to send it on its way. The approach of selecting a single goal, or series of waypoints can be input simply with the users clicks seemed like the most versatile. With a goal sent to the MSSP

network from the user, the MSSPs can begin to plan the vehicles route throughout the map. Each occupancy grid is greatly down sampled and shared on the network so that global path planning can initiate. A binary image of the environments object is used as the path plannings occupancy grid. A global planning algorithm determines a series of waypoint for avoiding obstacles. Because the IEA system is capable of detecting moving obstacles, a fast path planning update rate is important for obstacle avoidance. A local planner based off a pure pursuit approach, is the final process before each MSSP sensing the vehicle calculates a confidence, target speed, and target angle. These values are then input to the MSSP selector, and converted to command velocities, which are in turn sent to the vehicle for throttle and steering. In order to determine a confidence level, metrics for the pixel size of the detected vehicle, past command trends and confidence levels, and position in the cameras FOV are all fused into a single metric for confidence. This confidence ensures the MSSP with the most visibility of the scene is in charge of commanding the agent.

2.1 System & Architecture Overview

This section begins by describing milestones set in order to complete an operational system described above. The first phase of this research consists of focusing on the functionalities of a single Multi-Sensor Sensing Pack. Once a single MSSP is operational, the multi-unit architecture can be developed. The application is then implemented at full scale, providing a real-world testing platform in a controlled environment. Furthermore, the testing metric will be briefed such that the system discussion may remain focused on the outcomes at hand. An overview of the IEA architecture is introduced below. The discussion on this architecture will continue through this section, and be referenced throughout.

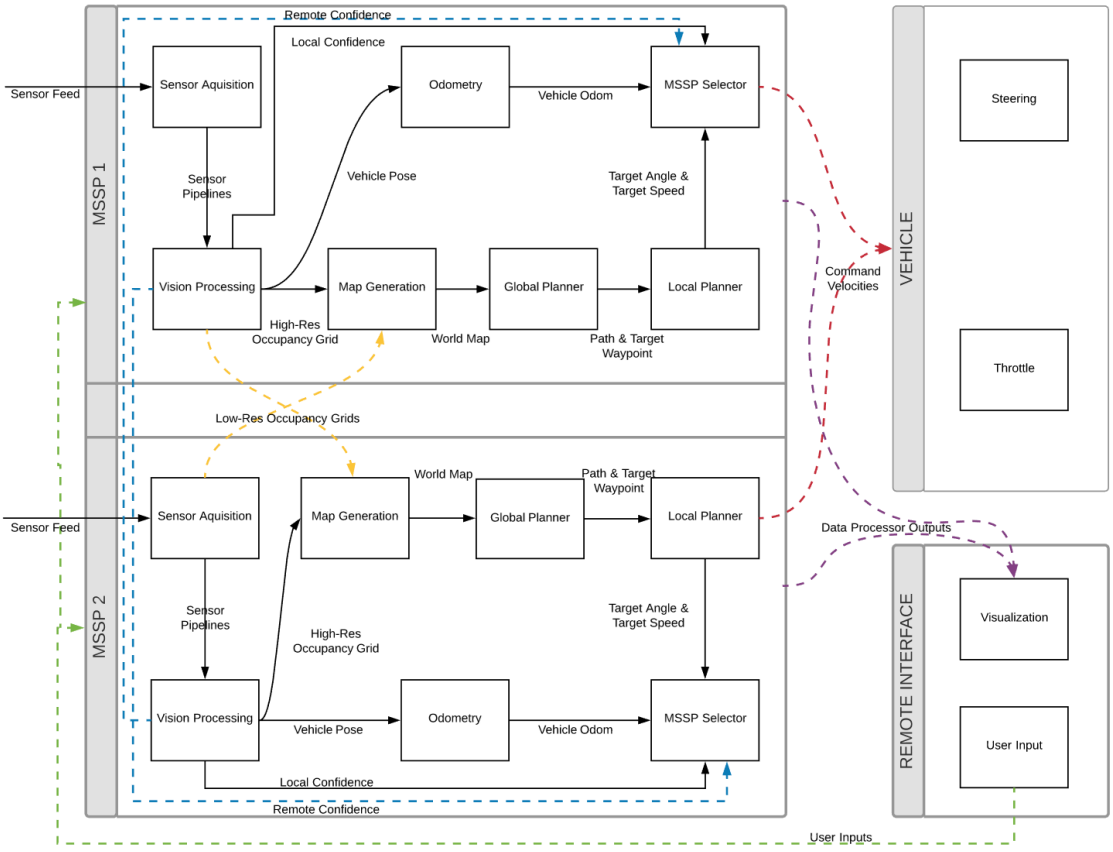


Figure 2.2: Overall system architecture of the IEA system implementation.

This figure contains many of the internal working components of the software implemented on each hardware system, and illustrates show each node or topic interacts within the system. It is important to note that the dotted lines are wireless transmission, and solid are wired transmissions. This is a main focus of the IEA development, and will be discussed throughout the paper. each hardware item is labeled in a gray band, making a total of four different components. The remainder of this section details the milestone of development for this architecture, both hardware and software, and the design considerations for developing an operational system to be tested.

2.1.1 Development Milestones

The Multi-Sensor Sensing Pack is designed to provide an enclosed set of sensors and cameras capable of capturing and processing the surrounding environment. The first task is to construct

an enclosure to house the computing, power distribution, and sensing required for an MSSP unit. Once the housing has been designed to enable the capturing of raw data streams, the MSSP is tasked with detecting obstacles and vehicles. After each object is detected and classified, the MSSP then determines the best path for the vehicle to reach its goal, while avoiding the obstacles. Finally, a robust and consistent wireless communication from the MSSP to the vehicles controller is added for continuous control.

The set of milestones described above lead to the operation of a single MSSP. However, in order to present a comprehensive solution, it is necessary to development a multi-unit system. This allows further research into the scaled application of an IEA system, and how to best perform hand offs from one MSSP to another. The two main areas of focus for the development of the multi-unit system are the architecture for data transfer, and a reliable and fast hand off protocol. Because of limited bandwidth, sending data between vehicles and MSSPs can quickly slow the system. A goal for the multi-unit system is to minimize the data transfer from one MSSP to another. This IEA system inherently minimizes data sent to the vehicle, and so the remaining factor is the communication between MSSPs. Techniques such as background detection, map building, and occupancy grid resolution can all affect the size of these data transfers. The last milestone of the multi-unit implementation is a robust hand off protocol that acts as the MSSP selector, based off of each MSSPs calculated confidence.

2.1.2 Design and Testing Considerations

Infrastructure Enabled Autonomy presents an interesting approach on how to best control a network of autonomous agents over a large continuous operating space. The decentralized nature of IEA removes as much computing and intelligence as possible from the vehicle itself, and can plan continuously through each nodes perception of the environment. This setup provides a striking advantage. The goal of these novelties is to determine how to best setup the communication protocols from MSSP to vehicles and between MSSPs, whether for sending vehicle commands, orientation, confidence, or map building. Also, this research hopes to implement and test a robust hand off method when switching control from one MSSP to another.

The testing of this research took place indoors, in a scaled environment with scaled 'buildings' and road lines. Windows shed real light into the environment, changing it over time. However, this setup reduced the dynamic level of the environment, and simplified the implementation for testing. It should be noted that these simplifications took place in the testing environment, and the design of the IEA system was tailored to utilizing this testing space.

In order to determine the success of the IEA system, testing and validation metrics must be determined. Testing this system should determine the successfulness of how the system operates, can be improved, and can be best organized for success in a real world implementation. The metrics of interest will be focused on three topics, and will be discussed in dept in the Results and Discussion section. These topics include quantifying the pixel error of the vehicle detection during the shared control implementation of the MSSP and map building for path planning and obstacle avoidance, the time delay induced from utilizing wireless command protocols, and the cross track error on straight paths due to all the factors that make up the vehicle's shared control system. With these tests, milestones, and requirements in mind, the scope of this IEA system begins to form. The remainder of System Methods and Description section details each item of the IEA system and the methods in place for the intelligent transportation system.

2.2 System Configuration Description

Various levels of architecture for this full systems had to be implemented and integrated to operate together. The entire IEA system required both hardwired and wireless connectivity, and within each unit (both vehicle and MSSPs), firmware, system configuration, and network setups had to be initialized for use. This section introduces the architectures of the system and its underlying components.

2.2.1 System Architecture

The IEA system, as introduced in the requirements section, should consist of at least two static sensing units. This enables fused sensing of multiple perspectives, and the ability to generate a map for the sensor feed from each MSSP unit. For the vehicle control, a distributed network will

enable the vehicle to operate under shared control from the MSSPs. It is important the network bandwidth can support these functions including sending control commands to the vehicle, to sharing processed sensor data, to be able to build and analyze a fused environment for the MSSPs to plan and operate the vehicles within. This successful hand off is both time dependent and accuracy dependent. The diagram below lays out the overall architecture of the IEA system. This figure will be referred to multiple times throughout the remainder of this chapter, the focus and level of depth of discussion will proceed along with the chapter. Presenting the diagram now is to introduce the overarching system design, and the units that reside within it.

2.2.2 Network Architecture

One vital component to the success of this research was the network configuration. To insure robust operation, the network needs to support bandwidth for real time control commands being sent to the vehicle from both MSSP units. Furthermore, the network needs to be able to allow the MSSPs to share low resolution occupancy grids with one another. These occupancy grids will, in turn, become the C-space, and are vital to the path planning algorithms, and to the map generation with multiple perspectives from the MSSPs.

To meet all of the needs stated above, and optimize the headroom for growth, a high power router with a 5Ghz band was utilized for the entire system. Long (200ft) cat6 cables connected each MSSP to the router for Local Access Network transfer between the two MSSPs. The router also utilized its wireless capabilities to connect to the vehicle through a hosted network. At this point each unit could operate on the same network.

Finally, a remote Graphical User Interface enabled a user to input commands to the vehicle, and visualize real time outputs. This can be performed wirelessly, as the wireless network load is minimized for the GUI.

2.2.3 ROS Framework

The Robot Operating System (ROS), is an open source collection of work that enables multi threaded applications and passing data between devices in a standardized format commonly uti-

lized in control and robotics applications. ROS is a powerful and flexible solution for collaborating various levels of hardware and intelligent agents. For this research, ROS is utilized across all three hardware platforms (two MSSPs and one scaled electric vehicle) to manage the network communication and to run each system's source code. ROS is flexible in that it can compile both C++ and Python. For this research, a majority of the code was developed in C++. When hosting a ROS Network, each node can be connected to the network, and a single master serve the passing of data. For this network implementation one MSSP operated as the ROS master. The other MSSP and electric vehicle connected to this hosted ROS network as slaves. The benefit of utilizing a ROS Network is that each node is able to publish and subscribe to topics on the network. these topics consist of various formats for data transfer between nodes, and any unit on the ROS network and communicate with any of these topics.

For this research, any data transfer from one hardware unit to another was performed over the ROS Network. As shown in Figure 2.1 above, by the dotted lines, the wireless transfers consist of three items: the low resolution occupancy grids shared to each MSSP for map generation, the vehicle control commands sent to the vehicle, and the vitals sent to the remote GUI for the operator. In order to ensure reliable and consistent control commands, the resolution and frequency of other topics were adjusted such that they utilized as little band width as possible.

Within each MSSP, the ROS network is capable of much larger data transfers. Publishing and subscribing to topics within a single hardware unit is similar to the structure of sharing data in multi threading applications. This ROS framework enables all the levels of intelligence to be computed within a single CPU. The four CPUs used in this research can be seen by the solid box definitions in Figure 2.1. A simplified version of the published and subscribed topics are shown as solid lines transferring topic data between nodes.

2.2.4 Remote Interface

In order to send commands to the vehicle, and visualize processed data from the MSSP, a remote user interface needed to be developed to operate the IEA system. To complete this task, there needed to be visual processing, and graphical input interpretations, such as mouse click and

key interpretations. OpenCV is an open sourced computer vision library which enables all of these functions. The simple GUI can be seen below in 2.3.

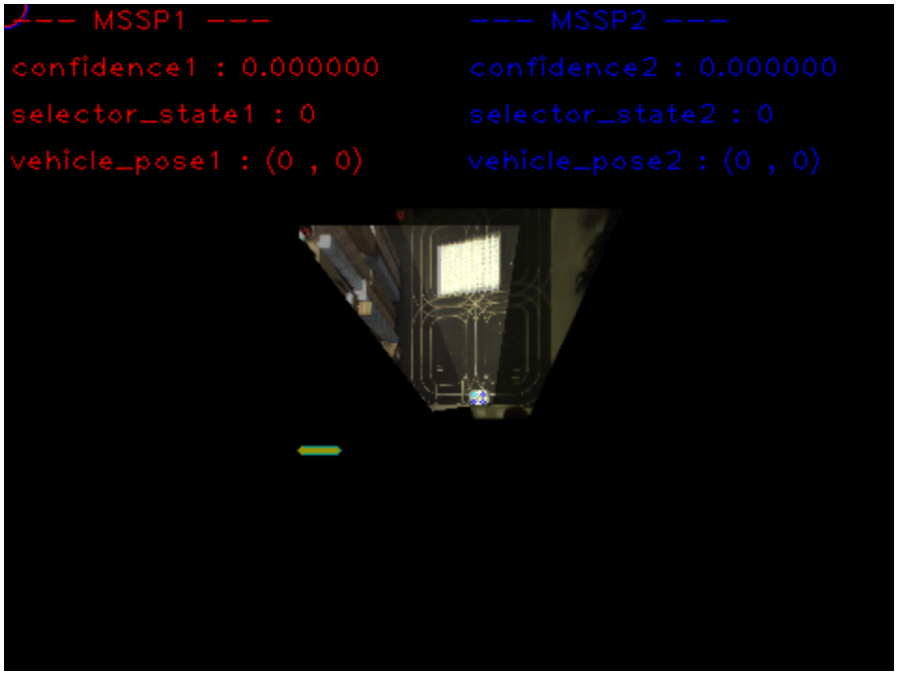


Figure 2.3: Graphical User Interface for remote monitoring and visualization of IEA system.

The visualization of data from the MSSPs consist of some vitals such as vehicle position, confidence level, and MSSP selector state from each MSSP unit. These values are organized in the top of the screen, and shown in red and black. The center of the GUI displays the down sampled image feed from each MSSP, and their transformations. Checkerboard initializations discussed in the computer vision section, vehicle's planned and taken path, command and current heading angles, and look ahead radius for path tracking are all also visualized during operation of the MSSP. These values enable the remote user to understand how the system is operating, and visualize the effectiveness of control.

The functionalities provided by this developed GUI include, HSV detection palette, initializations of the MSSP perspectives, goal setting for the vehicle, and arm and disarm command for the

vehicle. The HSV palette as shown in the Computer Vision section, enables the user to configure detection for any vehicle of interest. The initialization of MSSP perspectives, re detects the checkerboard, and undistorts the image to a top down view for control. This step also reinitializes the homogeneous transformation from one MSSP to another. Finally, the last step in initialization, is to reinitialize the environment. This is perform through background subtraction and is discussed further in the Computer Vision section. In order to control the vehicle, the user simply can click on the screen to set a blue dot, acting as a goal for the vehicle. These goals can be a single point, a set of points, or a looping array of points for continuous operation. The right mouse click will clear all set goals. The final user input is to arm and disarm the vehicle. This is a boolean setting which enables a safety feature for the operator to have final control over the vehicle.

2.2.5 Vehicle Control

The scaled RC car operates as an electric drive-by-wire vehicle with only two outputs (steering position and throttle). This requires two commands to be sent to the vehicle from the MSSPs. Each MSSP is capable of publishing these commands to the vehicle, and MSSP selector logic described in the MSSP Selection section determines which MSSP published commands. This segmentation requires wireless control commands to be sent. This in turn, creates a focus to ensure consistent transmission of commands. The control system diagram below in Figure 2.4 demonstrates how the vehicle control system operates.

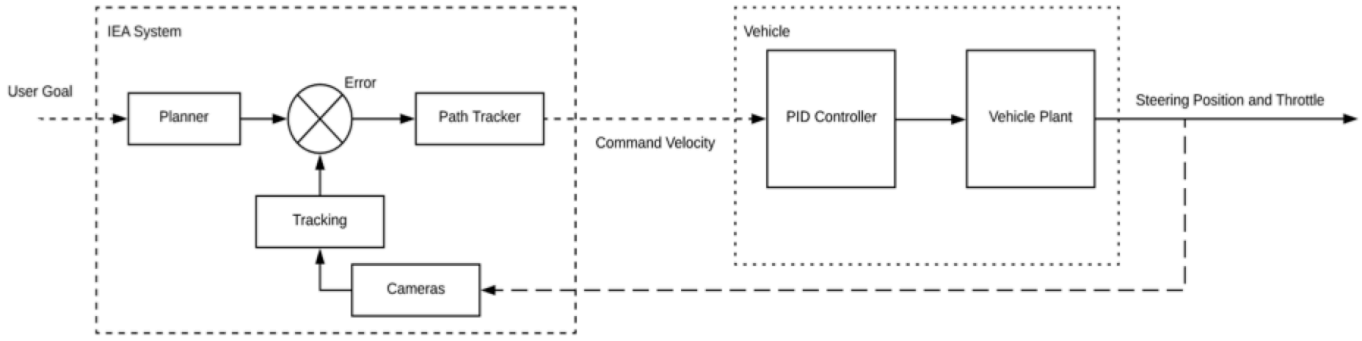


Figure 2.4: IEA system control diagram showing both the IEA and vehicles boundaries of responsibilities. The IEA system is distributed amongst MSSPs.

The figure above illustrates two separate components of the system (IEA infrastructure and Vehicle) working cohesively as the vehicle's control system. Once the reference inputs from the user are processed as a goal, a path, the desired speed, and heading can be used to calculate the error of operation. The MSSP's sensors are used to feedback the raw data for vision processes, which output the position of the vehicle. This is described more in the Tracking section below. After the error is computer, the path tracker calculates the control commands for heading and speed. This makes up the command velocity vector sent to the vehicle. Once the vehicle receives the wireless commands from the IEA infrastructure, the on board firmware takes care of converting the vehicle commands to outputs for the low level controller of the scaled electric vehicle. This segmentation allows each vehicle to posses its own tuned controller values, while the generalized command velocity can be sent to any vehicle. Finally the outputs within the vehicle's processor send drive commands to the steering and throttle actuators of the vehicle.

2.3 Hardware Design Description

As derived in the research requirements section above, this IEA system's hardware contained two MSSP units and a single vehicle. This sections focuses on the hardware implementation of the system, including the sensors and computing for the MSSP, and the network connectivity and actuators of the vehicle. The MSSP hardware was designed to be mobile, lightweight, and an all in one sensing and computing unit. The MSSP should be mountable and portable, meaning it should contain its own power source. The vehicle must be capable of carrying the added components for wireless connectivity to the ROS network, and sufficient drivers for stable control of the vehicle. With these hardware requirements in mind, the following subsections outlines the hardware within the MSSPs and vehicle.

2.3.1 Multi-Sensor Sensing Unit (MSSP) Design

The Multi-sensor Sesing Unit or MSSP is the unit that brings the IEA system to life. The MSSP has two sections, the internal power distribution and computing in the bottom, rectagular portion of the MSSP, and the form fitting sensor mounts on the top half. The CAD model can be seen in

the rendering below in Figure 2.5.

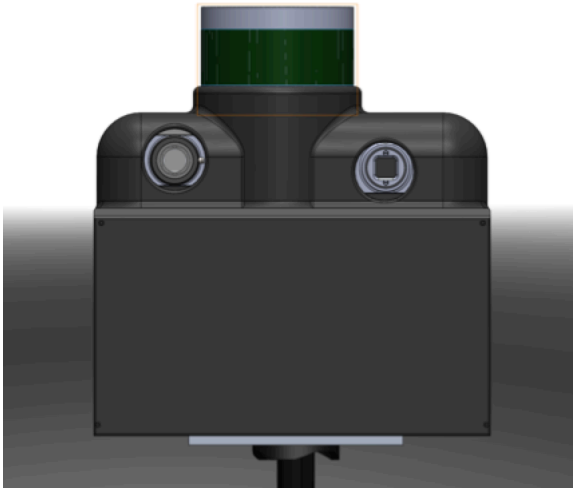


Figure 2.5: Finalized CAD model rendering of a single MSSP unit.

To meet the power requirements, a 12V bus from a 50000mAh battery powers the various components, as well as the CPU outputs providing direct power to its cameras and sensors via USB and ETH connections. This sensor pack is designed to have the capabilities of completely sensing what is happening in its surrounding environment. To achieve this, a variety of sensors were selected.

This MSSP unit consists of visual, LIDAR and thermal sensors along with a GPS, IMU and an embedded computer. The computing required to perform all the operations necessary for the IEA system is high, and therefore an Intel NUC i7 was chosen. Not only does this CPU have the sufficient clock speeds and cores, it runs on the low power 12V bus, and has the USB3 and Ethernet interfaces utilized for the MSSPs various sensors and network connection. Power via a Li-Ion battery and Wi-Fi network access untethers this system. Figure 2.6 below identifies each item in the MSSP.

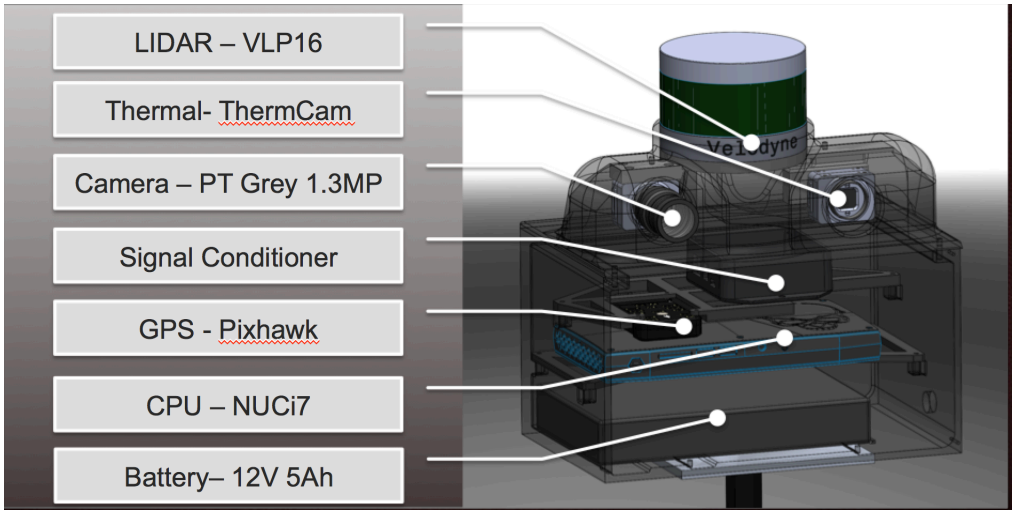


Figure 2.6: MSSP component visualization and labeled diagram.

While the scope of this research's implementation was limited to computer vision utilizing the visual camera feed, the MSSP was design for further development and fusion of sensors. The LIDAR utilized was a Velodyne VLP-16, and provides a highly accurate point cloud of the environment. LIDARs measure the euclidean distance to a target measuring the returning laser pulses fired at a high frequency. The VLP-16 is small and lightweight (less than 850g), operates within the MSSP's power requirements, and up to 100m range. The Velodyne LIDAR also consist of a proprietary signal condition which had a relatively large form factor. The specific mounting area for this can be seen in the figure above. The thermal camera is a ThermCap, and delivers 384 x 288 resolution image. These two sensor have advantages in poorly lit situations or night time. These sensor feeds can be fused with the visual camera to provide more robust localization of the vehicle and obstacles within the scene. The MSSP also consists of a GPS and IMU within the Pixhawk micro controller. Typically used as flight controller, this microcontroller has filters to integrate GPS and IMU data for more accurate localization, and can assist the MSSP in applications in need of a global reference frame. The scope of this research operated within the relative reference frames of one MSSP to another. Finally, the most utilized sensor in this research, the visual camera, is a Point Grey camera with a 1.3MP sensor named the Chameleon3. This camera can capture up to 30

frames per second, has a CCD sensor, and has a raw output resolution of 1288 x 964 pixels. This sensor feed was utilized for the computer vision process within each MSSP.

Beyond the components shown above, the MSSP needs to be mounted at an advantageous perspective. In most cases, if the lens permits this means a high placement above the ground plane, looking down at the area of operation. To achieve this, a 24ft tall tripod was used, and an aluminum plate that could bolt onto the underside of the MSSP was manufactured to act as the interface between the MSSP and tripod. This adapter plate had 4 hole for attaching to the MSSP securely, and a center threaded hole the interfacing with the tripod. This adapter's engineering drawing is shown in Appendix A.

2.3.2 Scaled Electric Vehicle

Because the IEA system utilized the infrastructure for all autonomy components, the only required aspects of the vehicle are carrying the components for wireless connectivity to the ROS network, and the drive by wire components. This means, that a full scale drive by wire vehicle use is synonymous with a scale electric vehicle. Due to this simplicity, the testing for this research was performed by a 1/12 scale electric vehicle. A view of this vehicle with the body removed is shown below in Figure 2.7.

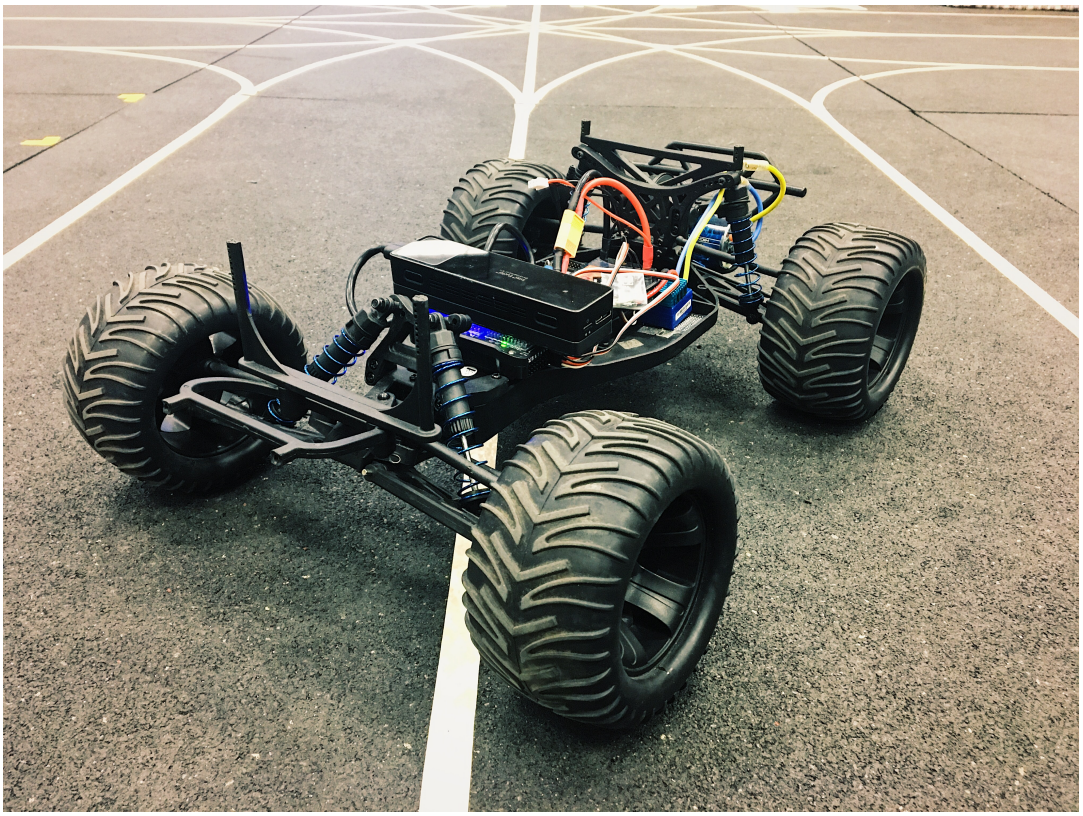


Figure 2.7: Isometric view of scaled electric vehicle without body.

The components that make up the electric vehicle's hardware can be seen below in Figure 2.8.

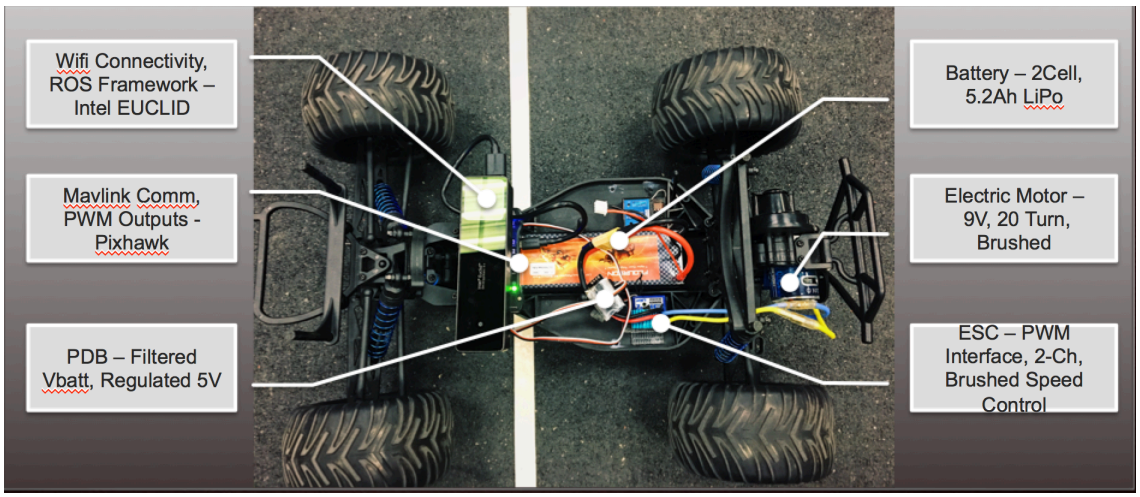


Figure 2.8: Top down view of the scaled electric vehicle, diagramming each system component.

An Intel Euclid is the small processor on board the vehicle. This processor runs an Ubuntu Linux operating system, similar to the MSSPs, and has the capabilities of connecting to the ROS network wirelessly. This device interprets the command velocity vector sent from either MSSP, and outputs actuator commands via Mavlink to the Pixhawk. The vehicle also contains a Pixhawk, not for the GPS and IMU like the MSSP, but to interpret the Mavlink control commands and output them as electrical signals. The drivers used on this vehicle use a PWM protocol. Pulse Width Modulation varies the duty cycle of a oscillating signal to vary the power level sent to the speed controllers. From here, the speed controllers (An ESC and Servo) interpret the PWM signal, and use the 7.4V power source to drive the motors to the desired position of the servo or rotational speed of the electric motor. These components receive a filtered power signal from a small power distribution board that outputs a 5V rail.

2.4 Software Development Description

The software development within each MSSP can be segmented into Computer Vision, Map Generation, Path Planning, Path Tracking, Odometry, Confidence, and the MSSP Selector. These section are consistent with the architecture diagram shown in Figure 2.1 above. This sections details each of these aspects of the MSSPs software implementation.

2.4.1 Computer Vision

The computer vision section begins with the initialization process of the IEA system, the detection, tracking and classification methods used, and the outputs sent to other various nodes throughout the system. As described in the MSSP Hardware section, the computer vision of this research was utilized visual camera's sensor feeds. OpenCV enabled the computational processes to be run for achieving the desired computer vision outputs.

Initialization

The initialization process begins with undistorting the raw sensor feeds. A predetermined calibration file is generated to determine the distortion due to the camera lens. A ROS pipeline, called Image Transport can interpret this file, and adjust the image, respectively. This calibration

process was completed through use of a the image transport's camera calibration, which can be completed with a given, open source script file. This process, and the MSSP initialization utilizes an 8x6 checkerboard. The size of the checkerboard is irrelevant, however the size of each square should be uniform, and input into the parameters of the IEA system. In this way, each MSSP can calculate a conversion rate for the ground sampling distance for each pixel.

After the lens distortion is removed, the image still remains distorted with respect to the ground plane. This is due to the angle of the MSSP with respect to the ground. For undistorted Cartesian mapping of pixels for vehicle control, this image needs to be transformed such that it shows a top-down view of the environment. To do this, the checkerboard must exist within the FOV of both MSSPs. This means the checkerboard must reside within the overlapping area between the MSSPs. Because the checkerboard is a rectangle, the detected vertices of the checkerboard can determine how distorted the checkerboard is within the MSSP's perspective. To accomplish this, OpenCV provides functions that can achieve this. The outputs are the pixel coordinates of the vertices. The next step to perform the top-down transformation, is to determine the homography between the cameras feed's checkerboard vertices and the known relation of a rectangle's corners. For this comparison, only the four corners of the checkerboard's vertices are needed. OpenCV also provides another method, getPerspectiveTransform() that has inputs of vertices like the problem suggests. At this point, the transformation matrix for a single MSSP to its relative top down view can be determined. A top-down view, for a single MSSP can be seen below in Figure 2.9, note the rectangular shape of the checkerboard.

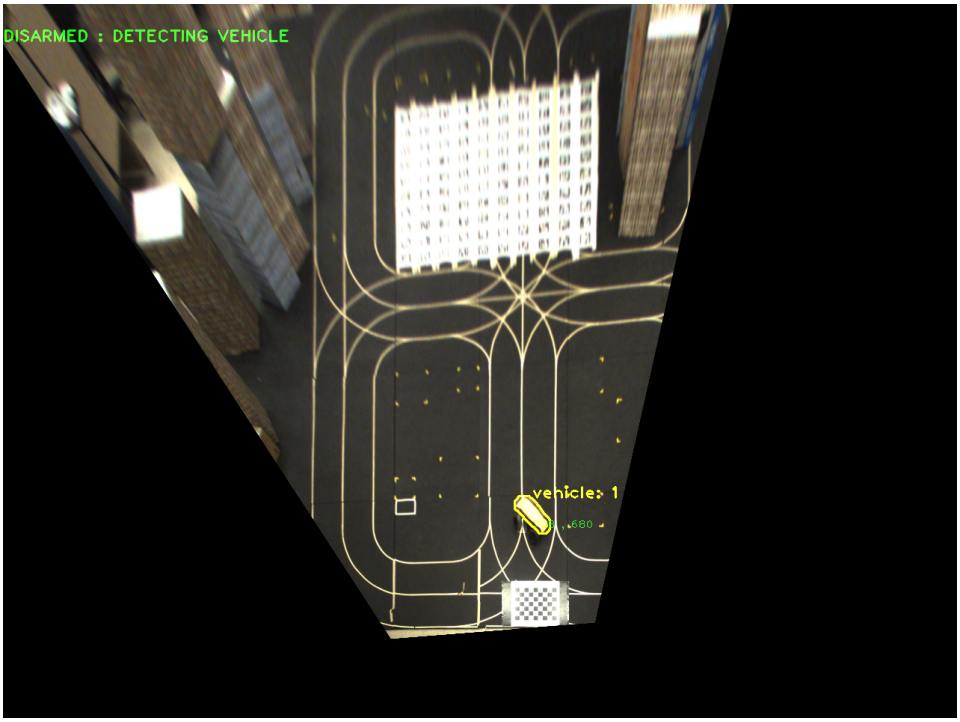


Figure 2.9: Transformed perspective to a undistorted ground plane.

The next step is to use this matrix to transform the four coordinates such that the four corners of the transformed image are determined. At this point, the euclidean distance between each corner can be calculated. Utilizing the known size of the checkerboard and these knew pixel distances of the undistorted, top-down image, the conversion factor for a single MSSP can be calculated. Each MSSP performs these initialization processes individually.

Now that each MSSP's image feed is transformed to the desired perspective, The detection and tracking computer vision methods need to be initialized. For this implementation, each MSSP stores an image of an empty environment as the background, and each following image is compared to this background, and objects are derived from differences between the background and current frame. This method is named background subtraction, and is best suited for static use cases. This is an example of implementations that would not be possible on the typical stand alone autonomous vehicle, because it's surrounding environment is constantly changing. At this point in the process, the initialization is complete, and these methods will only run again, when instigated by the remote

user over the GUI. The final preprocessing calibration that is involved with deploying an MSSP is the vehicle color segmentation optimization. This process will be introduced next, in the Detection section.

Detection

The purpose of detection of the MSSP unit is to detect both the vehicle, and other obstacles within the environment. Objects within the image are determined through the background subtraction method. However, it still remains to refine this image, and to extract which object, if any, is the vehicle of interest. To accomplish this task, the MSSP classifies objects by color. Utilizing the HSV color space, which segments an image into the partitions of hue, saturation, and value, the extraction of specific colors of interest is achievable. In this implementation, a yellow body was used for the vehicle. This body can be seen in Figure 2.10 shown below.

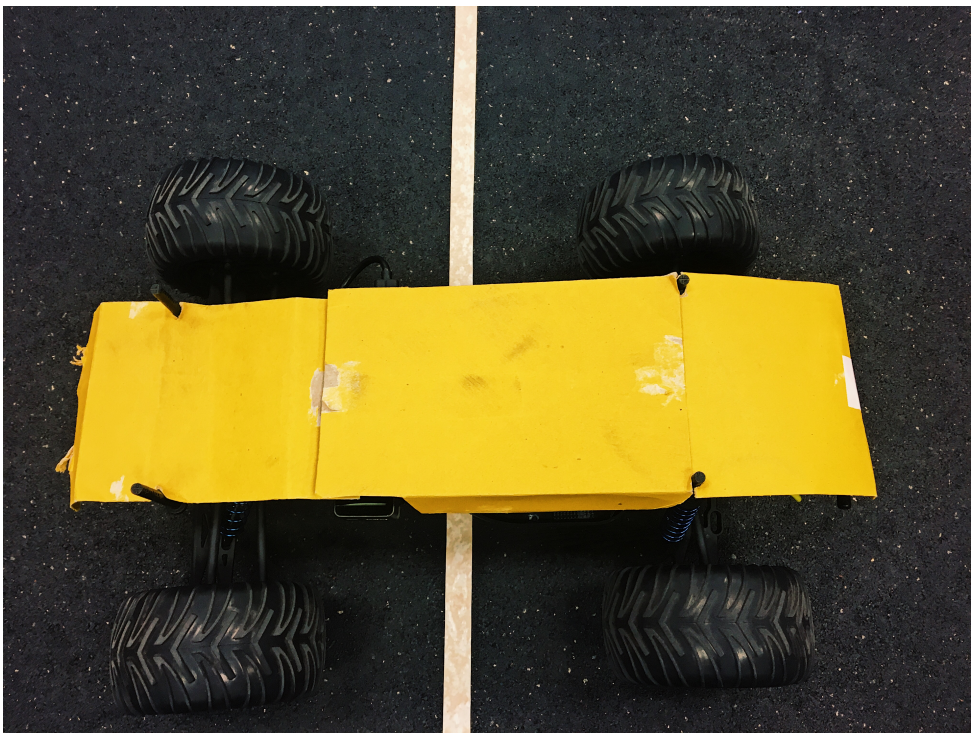


Figure 2.10: Scaled electric vehicle displaying body for detection and classification methods.

To define the range of HSV values for robust detection and classification of the vehicle, the MSSP had to be calibrated. This process is done with an optional GUI constructed to adjust the HSV segmentation values, as well as other morphological operations discussed later in this section. To complete this calibration, the vehicle was placed in frame (along with a variety of other color board for robust verification). An image capture of this process is shown below in Figure 2.11

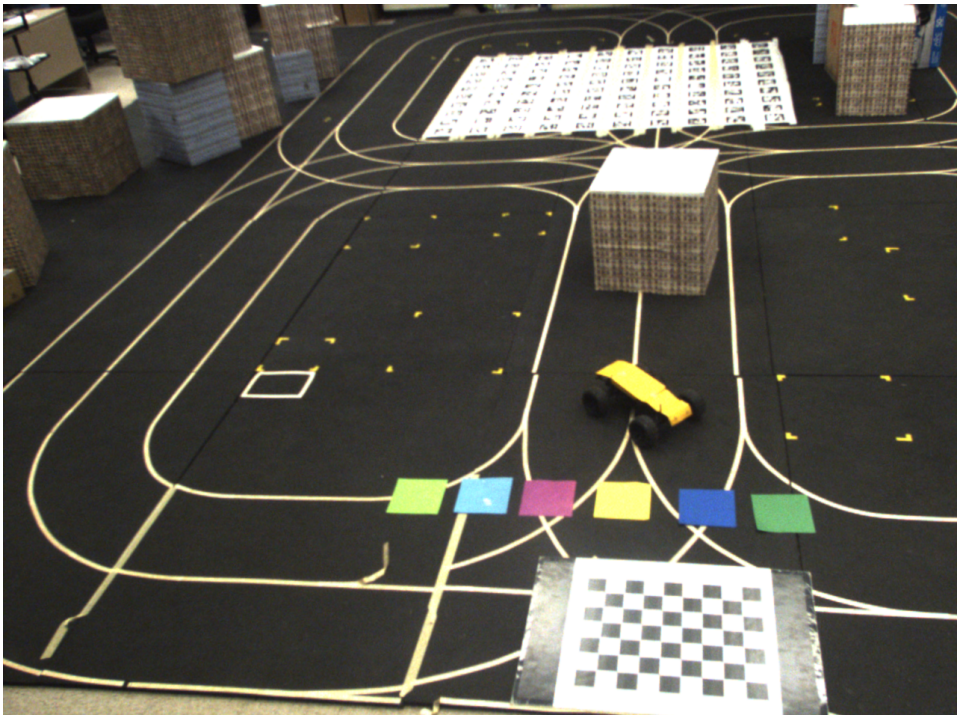


Figure 2.11: Original frame of interest for color segmentation calibration.

Notice the similar colors to the vehicle, such as the lighter yellow, and lime green. Also, take note of the small yellow tape markers that have the potential to throw off the vehicle detection. Noise reduction must address this, and will be achieve through morphological operations. The OpenCV HSV thresholding method is capable generating a segmented mask of the thresheld input values given. Therefor, the maximum and minimum values need to narrowed such that only the vehicle is detected in the scene. Figure 2.12 below shows the output of almost the complete range of HSV values. The output image is a binary set of objects which can then be further analyzed.

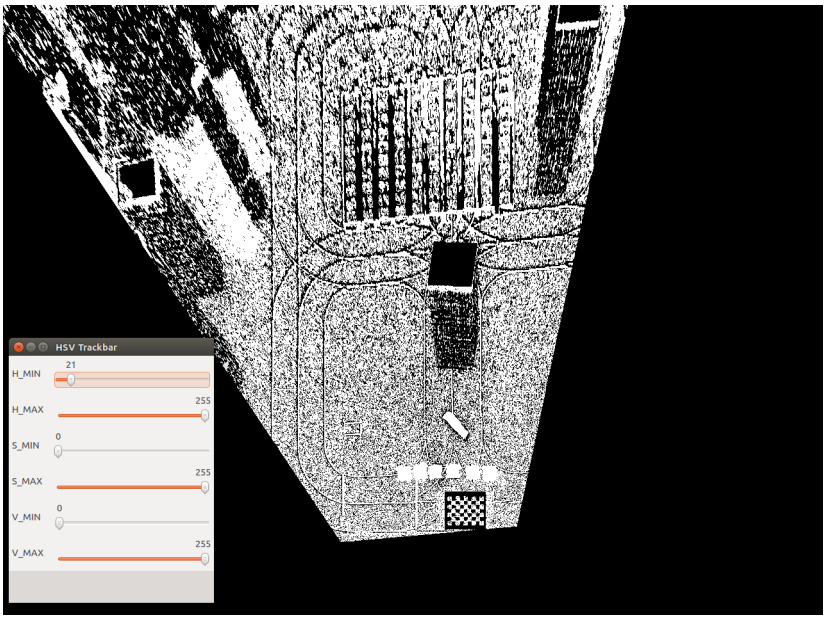


Figure 2.12: Original thresheld image with maximum HSV boundaries.

There is noticeably more segmentation to be performed, as the goal remains to only segment the vehicle for detection. At this point, the HSV sliders are adjusted to best isolate the vehicle's color. The following three images in figures 2.13, 2.14, and 2.15, illustrate the process of adjusting the hue, saturation, and value, in each respective image.

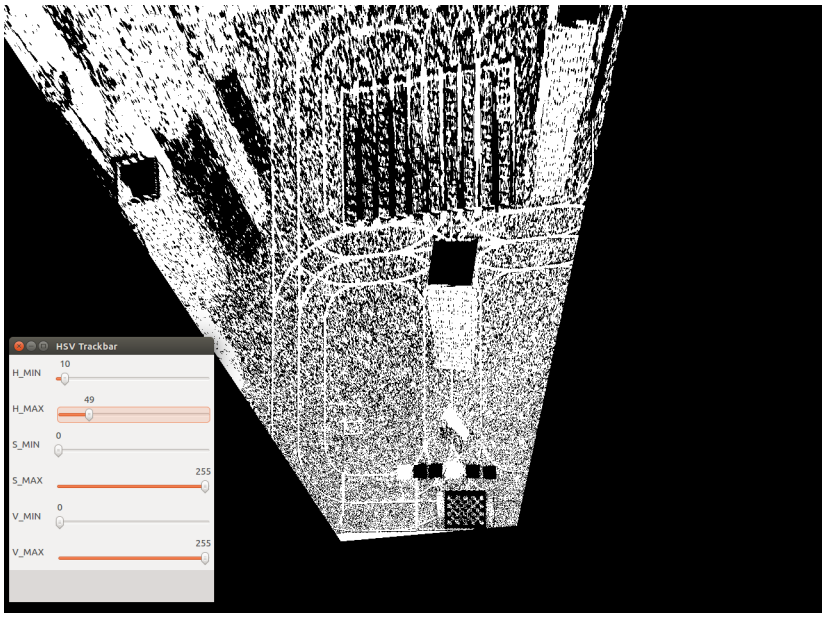


Figure 2.13: Color detection after setting limits for hue.

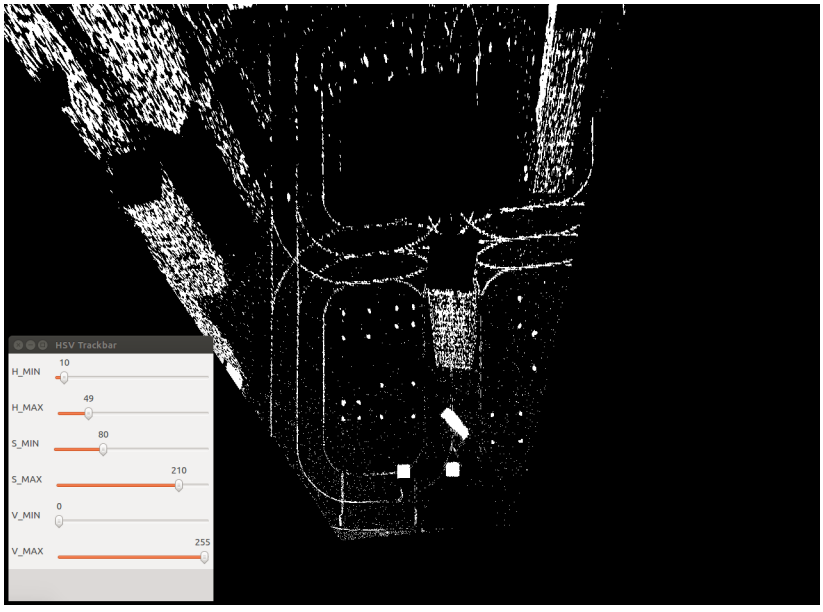


Figure 2.14: Color detection after setting limits for hue and saturation.

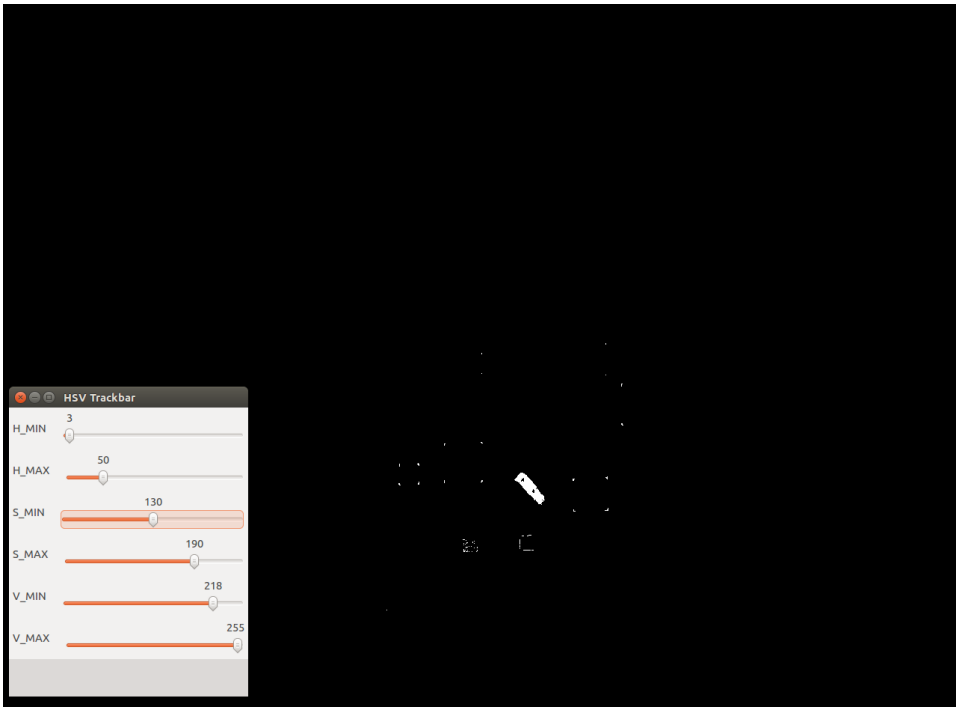


Figure 2.15: Final HSV segmentation before morphological operations are applied.

After adjusting the minimum and maximum values of the hue, saturation, and value thresholding limits, there remains a decent detection of the vehicle, along with some surrounding noise. From here, the goal is to create one continuous enclosed contour for the vehicle detection so that an accurate centroid can be calculated, which will later be utilized for localization. Furthermore, in order to eliminate noise, and further differentiate the vehicle from surrounding noise, morphological operations can be performed on the output binary image from the HSV thresholding. The operations performed include, a gaussian blur, erosion process, and dilation process. The gaussian blur smooths the image, which can aid in full object detection due to glare, imperfections of the object of interest, or hard shadows. The next step, erosion, eliminates surrounding pixels of each object. The erosion method eliminates the small particle noise seen in the image. Finally, now that the small noisy objects are removed, the remaining objects are dilated. The purpose behind this is to enclose each contour in hopes of calculating the most accurate centroid of the contour. At this point, there is hopefully a single remaining object. The output of these morphological operations

can be seen below in Figure 2.16.

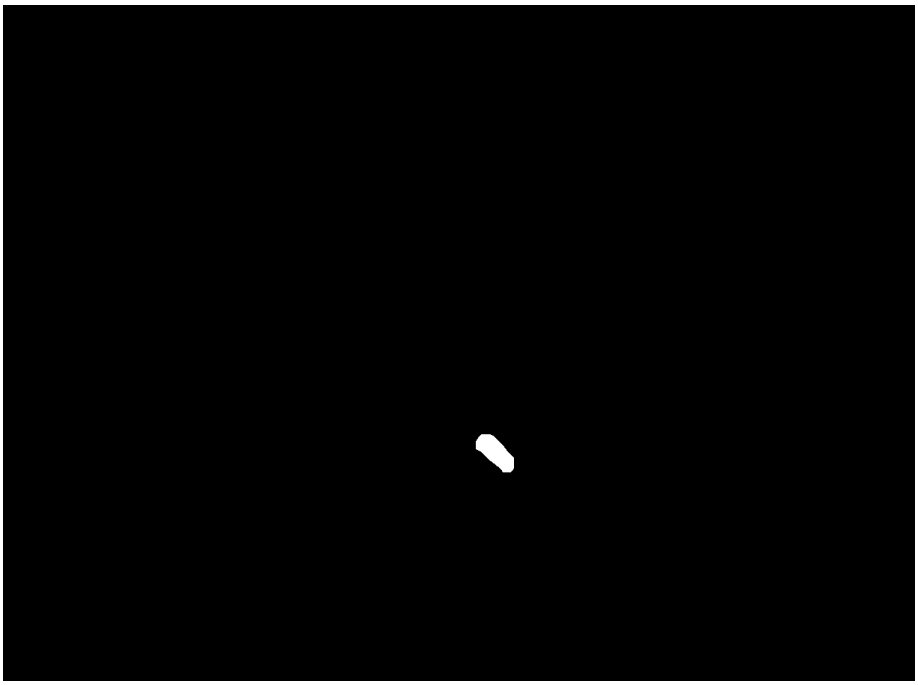


Figure 2.16: Final vehicle detection mask displaying a single object in frame, post morphological operations.

Upon successful HSV and morphological segmentation of the vehicle, these parameters are stored and utilized for each frame captured by the MSSP. It should be noted that each MSSP's camera sensor may have different sensitivities, and therefore, each MSSP needs to perform this calibration process. If there remains multiple objects, further analysis takes place. Multiple metrics on the proficiency of detection were derived and utilized to verify which object in the scene is the vehicle of interest. These metrics include, the detection size in pixels, converted detection area size, and resolution of detection in pixels/unit area. For the best suited object, the centroid is calculated in pixel coordinates, a vehicle class is instantiated, and the pose is stored as the vehicle.

With the vehicle segmented from the image, one can generate an object mask that contains each object except the vehicle. At this point, there will still be noise, and small negligible objects, and therefore the same morphological operations are performed on this mask as well. The output

will make of the obstacles in the environment, and act as the local occupancy grid for the MSSP.

Tracking

There are only two small differences in the tracking method compared to the detection method for the vehicle. Because under tracking, there exists the assumption that the vehicle will be detected over a number of consecutive frames. With this in mind, the past vehicle positions can be utilized to more robustly determine the vehicle's position. The two values added during the tracking method are the consecutive frame tracks, which aid in the confidence development, and the proximity to the past vehicle position, which can eliminate remaining contours far away from the area of interest in which the vehicle exists.

Beyond just the vehicle detection, the tracking method also determines the vehicles global orientation. This is important for controlling the vehicle, because the difference between the heading and the desired heading, is proportional to the steering input. To accomplish this, two methods were utilized, a polynomial projection, and a principal component analysis. The polynomial projection utilized three pass coordinates to generate a third order polynomial. This polynomial is extrapolated one time step, and the vector from the current position to the extrapolated position acts as the heading of the vehicle. This is a very robust solution under the assumption that the vehicle detection is sufficient. The downfall to this method is when the vehicle is not moving. In this instance, the past position coordinates could either be identical or very similar, which would not be able to produce a meaningful polynomial. To combat this downfall, a second heading method was implemented for when the vehicle is stagnant. The principal component analysis generates the eigenvectors of the contour, which would point along the longitudinal and horizontal axis of the object. This provides a fall back method for determining the heading of the vehicle. When the vehicle has been moving, the past direction of travel is known, and therefor the front is simply detected. However, the downfall of the PCA method, is if the vehicle begins at rest, the method has no way of determining the front of the vehicle from the rear. This is why a combination of the two methods was implemented, and performs the best.

Finally the tracking method publishes the vehicle's position and orientation to the ROS network

to be used for planning and path tracking.

Computer Vision Outputs

The computer vision node handles the environment initialization, vehicle detection and tracking, and generating the occupancy grids for the planner. The outputs for vehicle position consist of pixel coordinates in the MSSP's local reference frame, as well as the current heading. The obstacles residing in the second image mask have the vehicle removed. This binary matrix is used as the occupancy grid after resized. For each MSSP, and for each frame, a low resolution and hight resolution occupancy grid are generated. Because MSSPs share their sensed environments with each other, each MSSP publishes an occupancy grid wirelessly over the ROS Network. This publishing remains part of the time dependent path planning, and therefore it is important to minimize the traffic. The high resolution occupancy grid remains local within the respective MSSP, in hopes to provide a higher fidelity path, when the MSSP is selected as the controller. The final image collected from the computer vision node, is RGB image with drawn outputs such as obstacles, contours, position, and heading. This image is transferred to the remote GUI at a lower rate (2 or 4 Hz), as to not over load the network. Finally, each topic is published to the ROS network for use of other nodes. Figure 2.17 displays a single MSSP's corresponding high resolution occupancy grid, and detailed image sent to the remote interface.

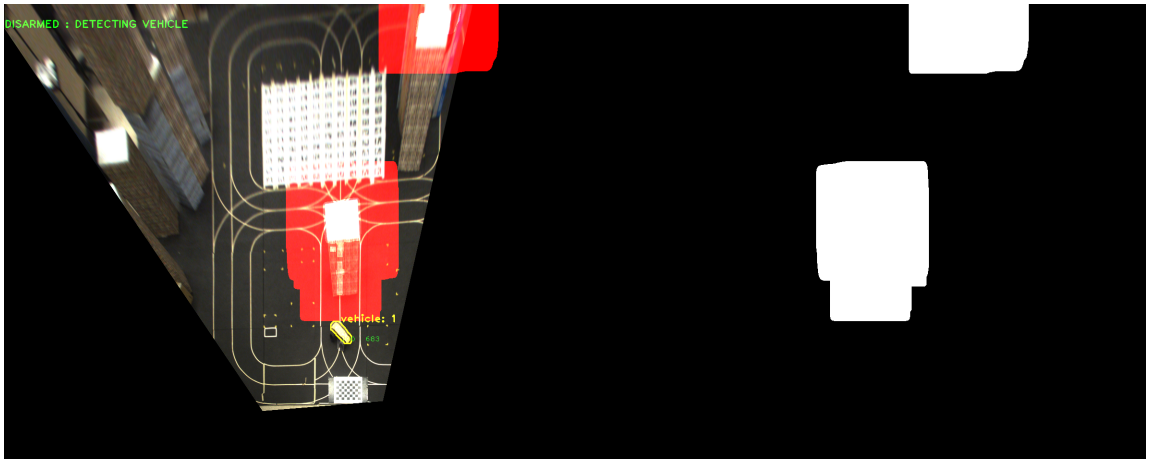


Figure 2.17: Computer vision node image outputs for a single timestamp.

The obstacle overlay can be seen in red, and cross reference with the occupancy grid next to it. Furthermore, the vehicle contour and position data can be seen on the frame as well. This image was taken during the detection mode, and does not contain heading or past positions. The following figure displays an example of a low resolution occupancy grid in Figure 2.18.

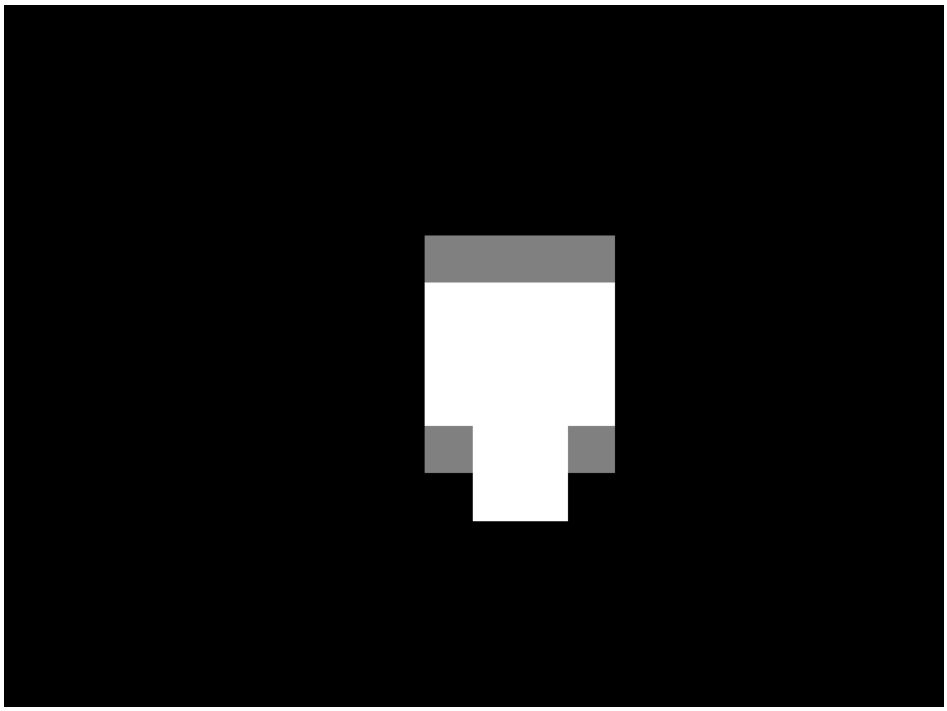


Figure 2.18: Low resolution occupancy grid.

The size of each pixel is noticeably larger scale due to the downsizing of the occupancy grid. Furthermore, the gray areas are due to a merging of the previous grid with the current one. In this way there is some level of probabilistic C-space implementation.

2.4.2 Map Generation

The transformation within the computer vision node to a top-down view is just the first step in the process to fuse each MSSP system. It was described in the initialization section, that the distorted checkerboard corners, and the assumption from a standard checkerboard's geometry, were

utilized to produce the perspective transform for the top-down view. This transformation was also applied to the corner's coordinates such that the transformed checkerboard coordinates were known. At this point, the checkerboard in each image should be rectangular in shape. The remaining adjustments consist of translation, scaling, and rotation. Because this set of transformations do not need a perspective transform, this operation will be an affine transform. An affine transformation is a linear mapping method. The affine transformation consists of the three operations necessary for fusing the two MSSP perceptions. is a linear mapping method that preserves points, straight lines, and planes. Sets of parallel lines remain parallel after an affine transformation. The affine transformation technique is typically used to correct for geometric distortions or deformations that occur with non-ideal camera angles. Similar to the determination of the homogeneous perspective transform, the transformation matrix for the affine transform utilizes the two sets or transformed corners. Applying this transform to the second MSSP, effectively aligns it with the first. Figure 2.19 displays the interface with two MSSPs transformed to the same coordinate system.

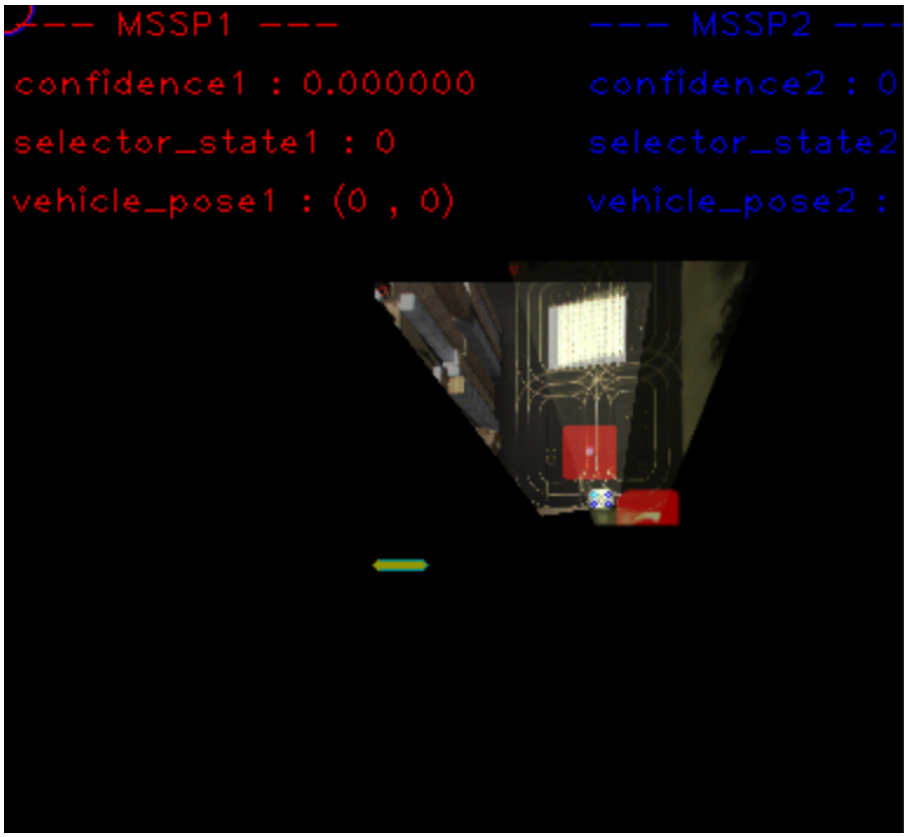


Figure 2.19: Visualization of fused MSSP perspectives, and their stream's overlay.

Each image is reduced to half its value, such that when overlaid, the image is not washed out. This also enables visibility of the distinction between each MSSP visualization. The figure above shows the overlapping checkerboard, and the accuracy at this point can be seen visibly. Certain imperfections can even be seen in the above image, away from the checkerboard, and at high contrast intersections. However, the results section will describe this accuracy is sufficient to control the vehicle, and therefore the next section will present the path planning method implemented.

2.4.3 Planning

Collected from the remote GUI, and inputted by the operator, the vehicle's goal is sent to the planner. Along with receiving a local MSSP occupancy grid, and a remote one, the planner, also subscribes to the transformation matrices, and the vehicles position from its respective MSSP. This enables the goal and vehicle position to be placed in the fused reference frame, and use the map

generated from the previous section. At his point, the exists a goal, initial position, and obstacles all in the same frame. This is what the path planning algorithm is applied to.

The A* path planning algorithm is a search algorithm that optimizes a path from point A to B. This algorithm is superior to other path planners, because it not only uses the distance to the goal, but also the distance from the start, and in this way, can output the optimal path. A* begins with a set of open and closed nodes, in this case, the occupancy grid. For each instance along the path, the algorithms looks at each neighboring node, and computes a cost based on the distances mentioned above. For each path found to the goal, the cost is minimized in an effort to find the most efficient path to take. This path is advantageous in an IEA implementation, because it is considered a global planner, meaning it plans the entire route to the goal. A stand alone autonomous car may have its perception obstructed, and in turn can only plan its next few steps. The output of this algorithm is a down sampled series of waypoints. These waypoints will be utilized in the path tracker in the next section. Figure 2.20 shows the output of the path presented on the occupancy grid.

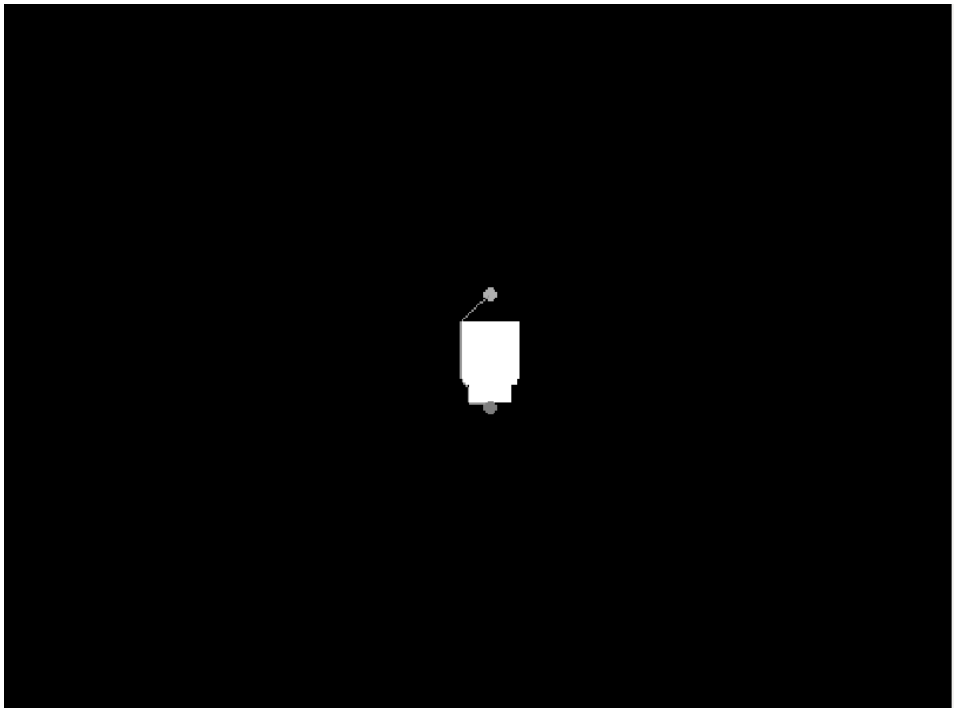


Figure 2.20: Visualization of A* planned path around the obstacle from start to goal.

The circles denote the start and goal of the planned path, while the white is an obstacle. The gray line weaves around the obstacle providing a complete path, even when the vehicle would be out of sight of the goal. The following figure, Figure 2.21 displays the path on the user interface for remote visualization.

Figure 2.21: Early version of remote GUI interface for a single-unit system visualizing the A* path, the obstacle, and the transformed camera feed.

2.4.4 Path Tracking

Utilizing the output vector of waypoints from the planner, the path tracker ensures the vehicle can follow along these waypoints with minimal error. To accomplish this, an algorithm called pure pursuit was implemented. Pursuit utilized a look ahead radius, determined by the turning radius of the vehicle, to find the radius' intersection with the path. The outcome is the vehicle operating as if it were chasing a carrot along the path. More complicated algorithm could take place of this, but

the results section will demonstrate this method produces stable control. Because the output of the planner were discrete points, these points needed to be parameterized such that there is always be an intersection with the radius. The geometric setup of this algorithm can be seen below in Figure 2.22.

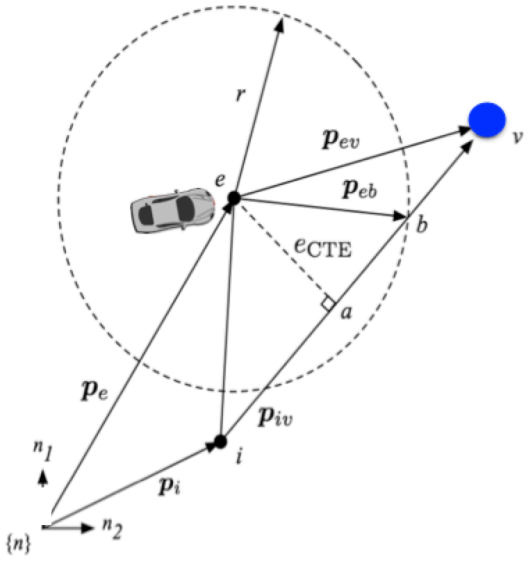


Figure 2.22: Diagram of pure pursuit, geometric approach to path tracking.

This diagram uses P for position notation, e to notate the current position, i to notate the initial position, and v to notate the goal. Notice e_CTE stands for the current cross track error of the vehicle. Cross track error is the orthogonal distance to the planned path from the vehicle's current location, and this is a useful metric to store when analyzing the effectiveness of stable control and path tracking. Before real time implementation, the algorithm was developed and tested in Matlab. Figure 2.23 below plots various cases along a straight path, the look ahead radious' intersection, and the desired heading.

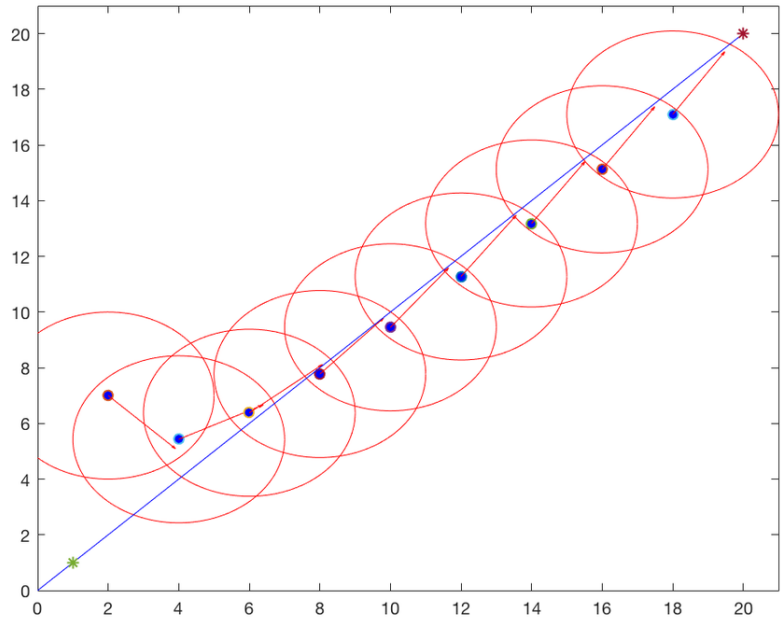


Figure 2.23: Plot of pure pursuit developed in Matlab prior to implementation into a real time system with C++.

The path is shown in the straight blue line, the look ahead radius in the red circles, and the desired heading is shown as the red arrow. Each arrow should be pointing to the intersection of the look ahead radius and the straight path. This process verified the algorithm, which was then implemented into the IEA system with C++.

2.4.5 Odometry

Odometry uses the data from the MSSPs, in particular, the pose pixel data to determine the real world position and the corresponding velocities. The odometry node, similar to the map generation, needs to transform each vehicle position coordinates into the fused reference frame, which consists of subscribing to each MSSP's transformation matrix, and then extract the homography between the two transformed corner sets from the checkerboard. As this point, each vehicle's position, from any MSSP is in the same frame, and can be used to control the vehicle. Furthermore, the purpose of the odometry node is to convert from pixels to real world coordinates. The conversion factor,

calculated and published in the computer vision node, is subscribed to and applied at this point. The final task for the odometry node is to output the quaternion, which combines all the pose data into a single message unit. The output odometry is published to the vehicle as pose data for current speeds and position changes.

While odometry utilizes the fused reference frame, and real world metrics, it is prone to error. Because the velocities rely on time integration and a high accuracy time stamp, and because ROS is only pseudo real time, error propagation can be induced. Furthermore, pixel error can be accentuated when converting to real world coordinates, as well as small MSSP movements due to wind and sway. The effectiveness of this position and velocity data will be quantified in the results section.

2.4.6 Confidence and MSSP Selection

In the instance of the vehicle residing within the overlap, both MSSPs should be capable of tracking the vehicle. In this instance, instead of sending two commands, or fusing the vehicle position data, a series of tracking metrics are used to determine which MSSP should take over the control of the vehicle. The metrics, similar to detection metrics discussed earlier, are the tracked area resolution, ratio of tracked area to the area initially detected, and the number of consecutive frames the vehicle has been tracked. The tracked area resolution converts the pixel size to a real world area, as well as the number of pixels found within the area. The ratio of tracked area to the area initially detected compares the detection sufficiency to when the vehicle was initialized. This value can exceed 1.0 in cases where the tracking exceeds the initialization detection. Finally, the number of consecutive frames the vehicle has been tracked ensures that the MSSP only switches when necessary. If both MSSP units track the vehicle successfully, the selector will continue to choose the MSSP that in currently in charge of operation. Otherwise, the MSSP selector will select the higher of the two MSSP's confidence values. The three confidence metrics are fused by complementary filter, and normalized to the initially detected confidence. For the complementary filter, alpha and beta terms are equal to 0.3 and 0.36 respectively. Figure 3.5 below shows an example of confidence levels, and the MSSP selector state choosing MSSP 1 or 2 for control.

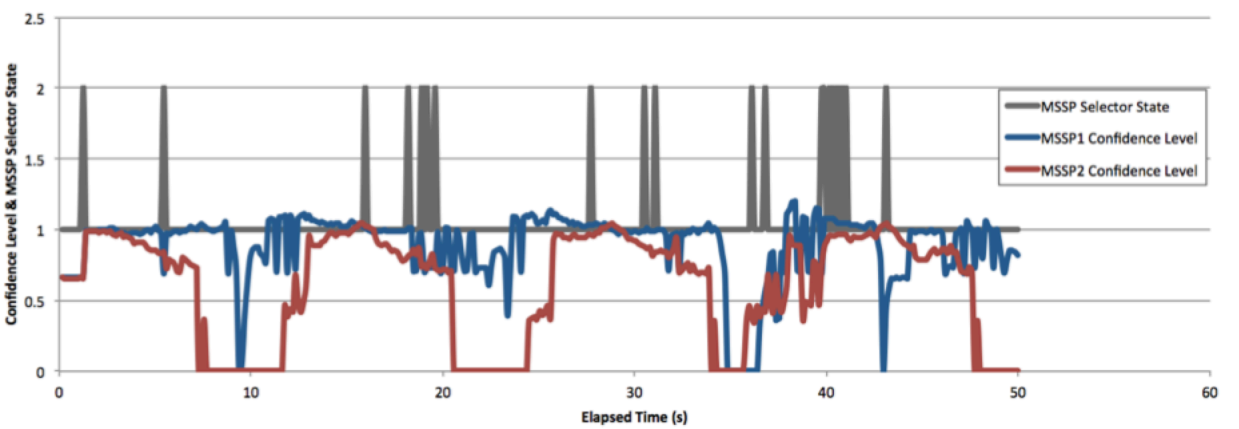


Figure 2.24: Two MSSPs confidence levels, and the MSSP Selector state.

3. RESULTS AND DISCUSSION

This chapter will demonstrate the implementation of the IEA system, discuss the testing matrix developed to produce outcomes to draw conclusions on the operation of the IEA system, and present results and post processed data from the trials. The sections will be presented as Implementation, Accuracy and Error analysis, and close with the overall Research Outcomes from this work.

3.1 Implementation

A binary result of the operation was the proof of concept through implementation. If the vehicle could not be controlled by the IEA network, there are no grounds to do accuracy analysis, and dive deeper into the advantages and shortcomings. Due to this, the first stage was to demonstrate the operation of the IEA system. Once operational, tests could be put in place to analyze particular aspects of this research, such as pixel accuracy, time delay, and cross track error, as will be discussed later in this section.

3.1.1 Single-Unit

The first milestone in development was the operation of a single MSSP unit. The single-unit focused on the implementation of many of the algorithms residing within the MSSP. These algorithms spanned from computer vision techniques, path planning, path tracking. Testing these in a single unit system gave a platform in which debugging was faster, and variables could be isolated more simply, while still early on in the development stage. This image below shows a screen capture of the interface for the single-unit system in Figure 3.1.

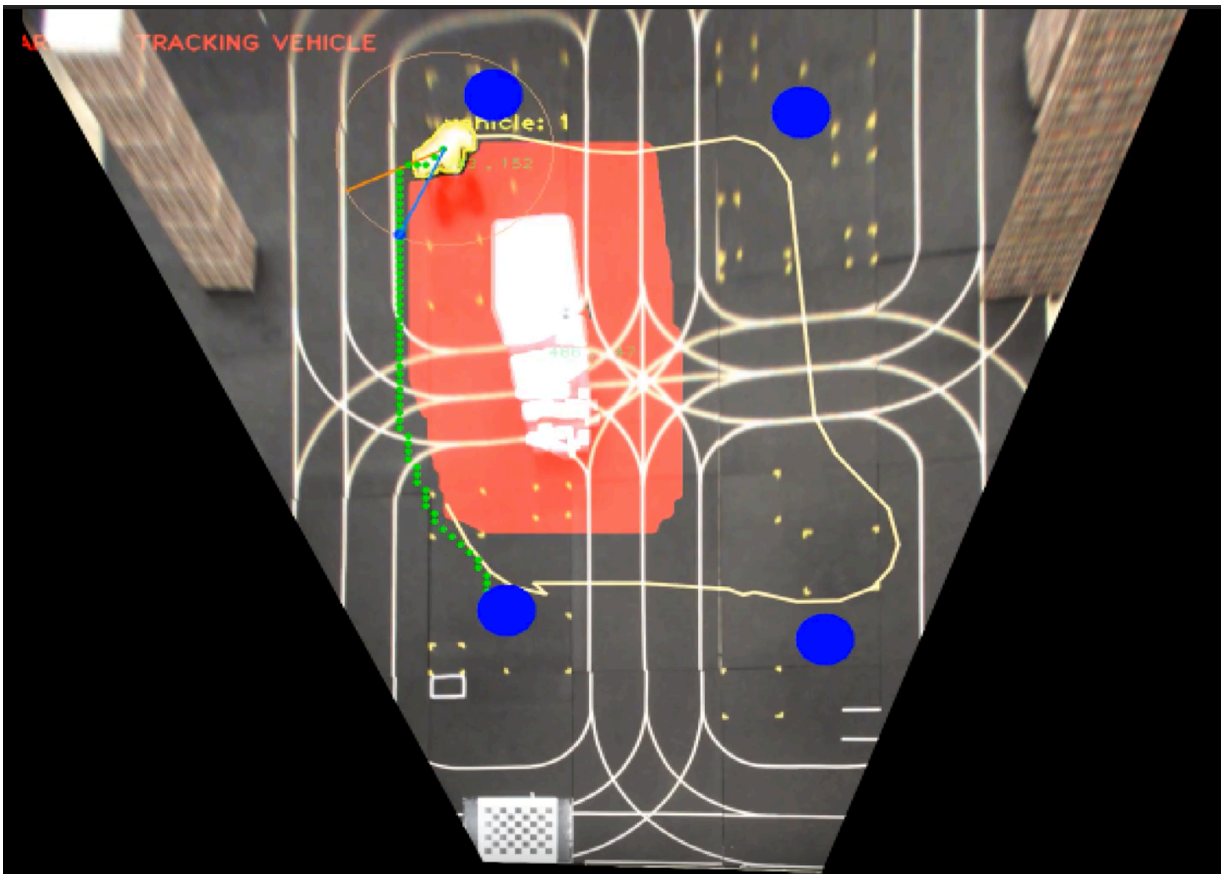


Figure 3.1: Remote GUI for a single-unit system showing the vehicle maneuvering around an object while tracking its path.

In this video for a single-unit implementation, four subgoals were selected from a remote user, and are shown as large blue dots. The vehicle is tasked with cycling continuously through these subgoals, while avoiding the obstacles in the environment. A center obstacle was placed for testing, and is shown by the white object within the red boundary. This object is dilated to the vehicles turning radius to ensure collision free navigation. To plan globally, A* path planning implementation is utilized, and series of waypoints are visualized as the green dots, showing the path to the next subgoal. The path tracking look ahead radius is displayed as the orange outlined circle, the orange line is the heading vector, and the blue line is the target heading angle. The waypoint closest to the intersection of the look ahead circle is colored blue. The vehicle, is detected with a yellow body, and the vehicles position data is traced in a light orange. It can be seen the vehicle

is capable of safely cycling through goals, while avoiding obstacles. Below shows a series of captures demonstrating the stright path tracking and control capabilities of the IEA system in Figure 3.2.

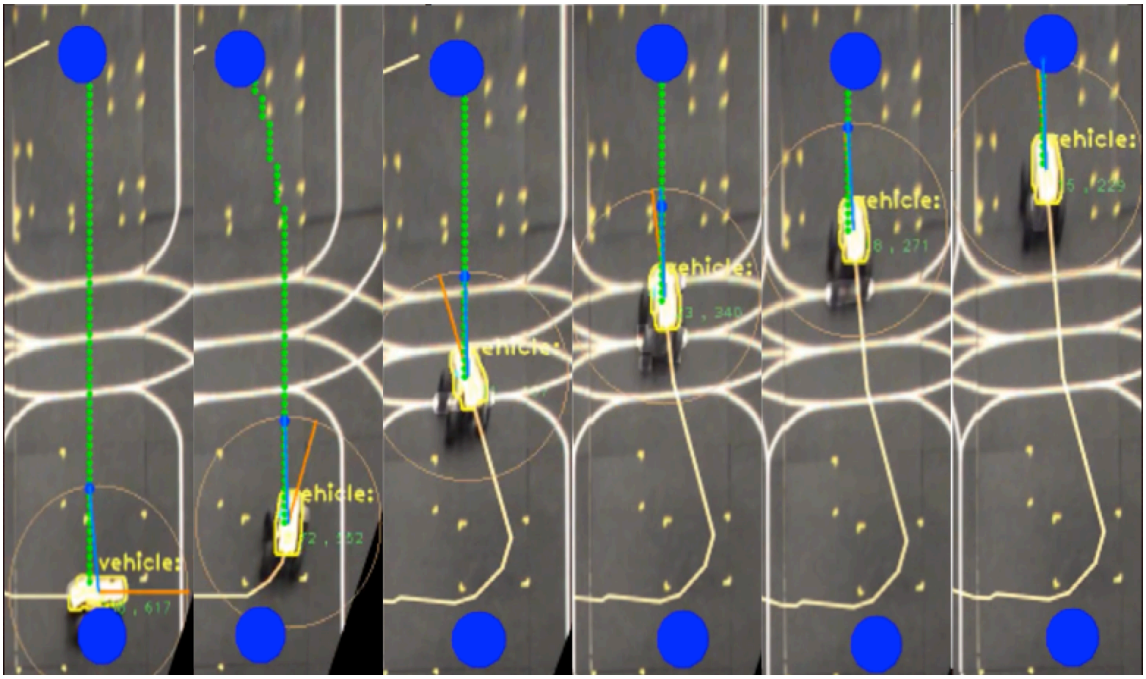


Figure 3.2: Vehicle control visualization of path planned and taken.

At this point, in single unit applications, the vehicle could operation autonomously through the IEA Network, both in straight lines, and continuous paths with obstacle avoidance. The next milestone is to implement a second MSSP to work cohesively with the first.

3.1.2 Multi-Unit

In order to present a comprehensive solution, it is necessary to development a multi-unit system. This allows further research into the scaled application of an IEA system, and how to best perform hand offs from one MSSP to another. The two main areas of focus for the development of the multi-unit system are the architecture for data transfer, and a reliable and fast hand off protocol. To test the development of a multi-unit system, two trials were performed. The first was a

straight path along the axis of overlapping between the two MSSPs, and the second was a horizontal path between the same MSSP configuration. The following GUI display shows a frame during the horizontal travel trial. Note the MSSP to path configuration in 3.3.

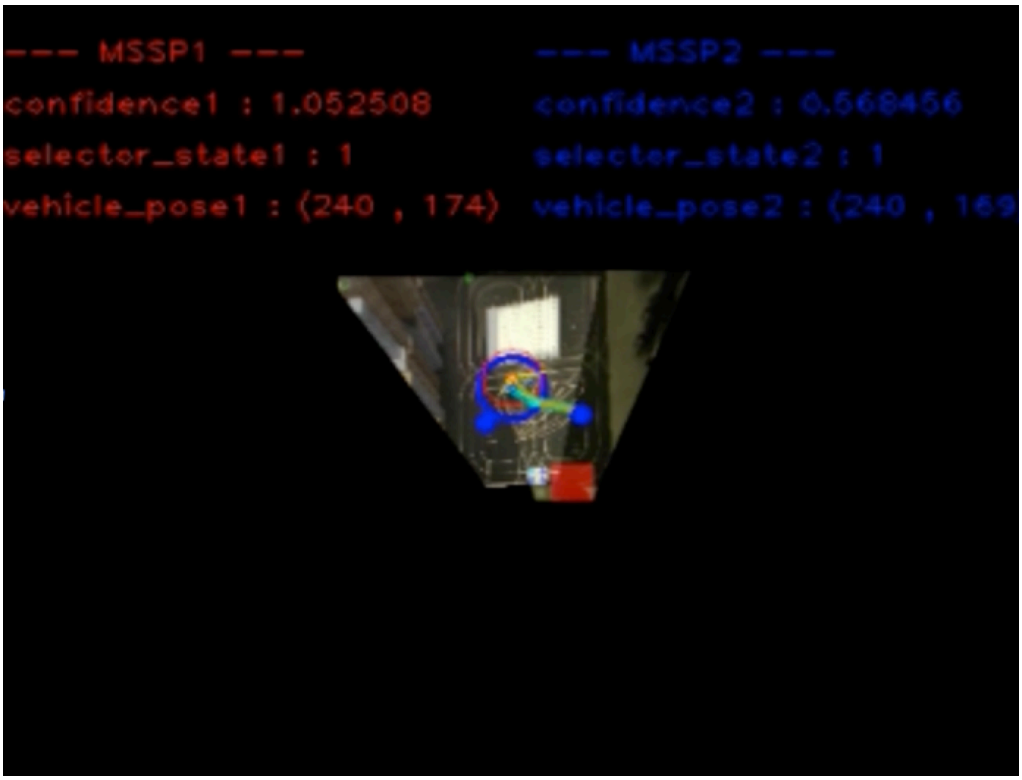


Figure 3.3: Remote GUI for a multi-unit system showing the vehicle on a set of goals horizontal to the MSSP's overlap configuration.

The goal of the horizontal travel trial was to test the loop time of vehicle commands, and its variance. The implications of this trial set the reliability of the shared control over a wireless network. To set the baseline, a static test was perform where the vehicle was tracked but not armed. This means there was little to no switching between the MSSPs, and there was good detection from frame to frame.

The second trial was vertical travel, this testing was put in place to quantify the level of cross track error along a straight path. This setup focused more heavily on the pixel error propagation

away from the calibration checkerboard, and the shared vehicle control between the two MSSPs. The following, Figure 3.4, displays the configuration of path and MSSPs.

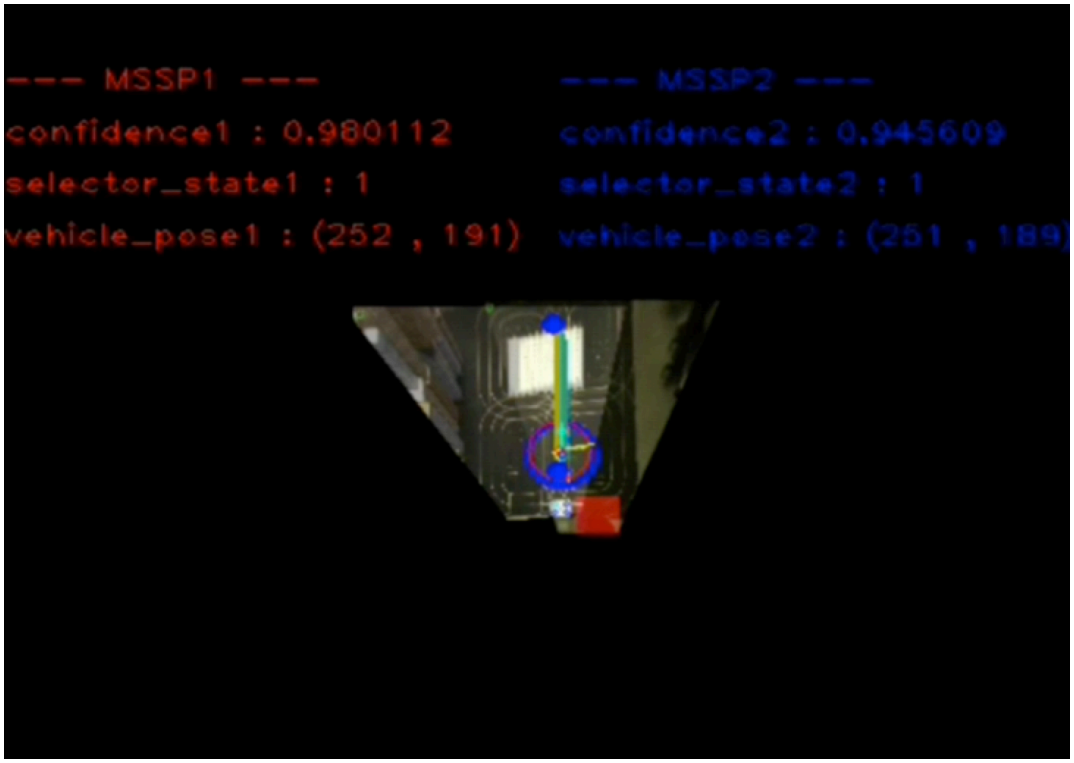


Figure 3.4: Remote GUI for a multi-unit system showing the vehicle on a set of goals vertical to the MSSP's overlap configuration.

For much of the vertical trial, the vehicle was operating in the overlapping area. This means, each MSSP should be tracking the vehicle, and therefore, publishing a confidence greater than zero. In the case where each MSSP have a decent level of confidence, the MSSP selector is relied upon to select the best MSSP for control. Utilizing the parameters discussed in the methods section, the MSSP selects the higher confidence except for the case when both confidence levels are above 1.0, in which case the selector keeps the current MSSP selected. Figure 3.5 shows the confidence levels and selection for this trial.

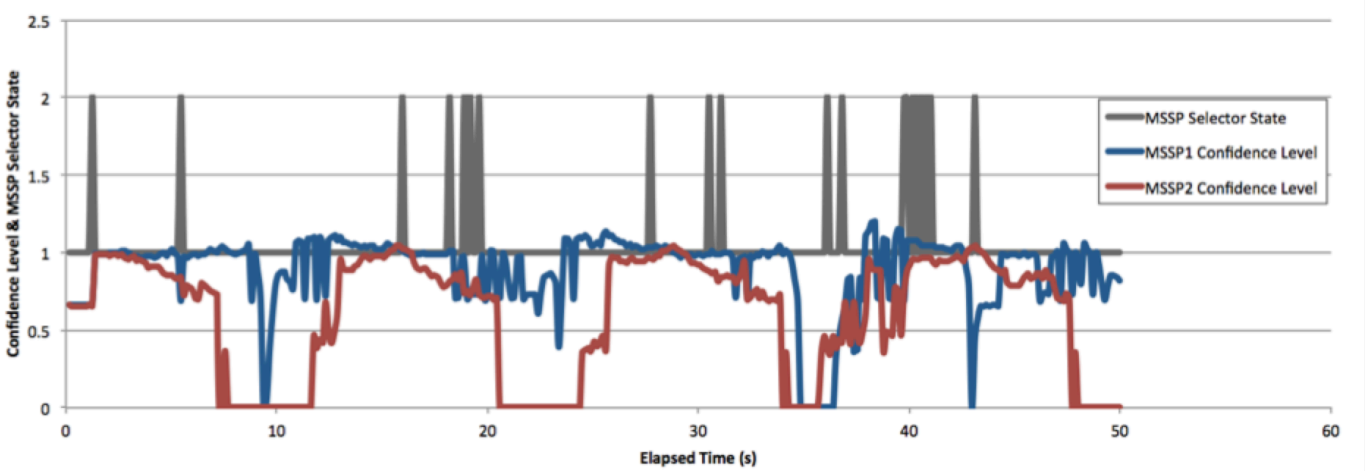


Figure 3.5: Remote GUI for a multi-unit system showing the vehicle on a set of goals horizontal to the MSSP's overlap configuration.

This figure shows the confidence levels of tracking for each MSSP unit over a single trial in red and blue. A value of zero is when there is no detection, in most cases, meaning the vehicle is outside of the respective MSSP's FOV. The grey data indicates which MSSP is selected for control. The only two values for this item is 1 or 2. As the figure shows, for most of the trial, MSSP 1 is in control. As stated above, this is due to the high levels of confidence from both MSSPs throughout the trial, and therefore, the MSSP selector minimizes switching.

3.2 Accuracy and Error Analysis

Once the multi-unit system was tested to be operational, the accuracy metrics were declared such that the success of the IEA system could be quantified. Pixel accuracy, variance in loop time, and cross track error, were determined to encompass the analysis portion of the IEA system. The trials utilized for these analyses were the horizontal and vertical tests described above.

3.2.1 Pixel Error Quantification & Propagation

For the MSSPs to accurately operate the vehicle under shared control, the perspective transform between the two MSSPs must be accurate. While the GUI provides a visualization of this map generation, the level error of cannot necessarily be visibly seen. To test and quantify the error,

multiple line markings at known distances from the MSSP unit crossed through the edges of the overlapping area. At this intersection, and because the line markings are white on a black ground plane, a simple thresholding produced a binary representation of these levels. At this point, each edge of the overlapping area could be compared. Discontinuous jumps in the line markings were noted as pixel error. The error for both the horizontal and vertical trials are shown below in Figure 3.6.

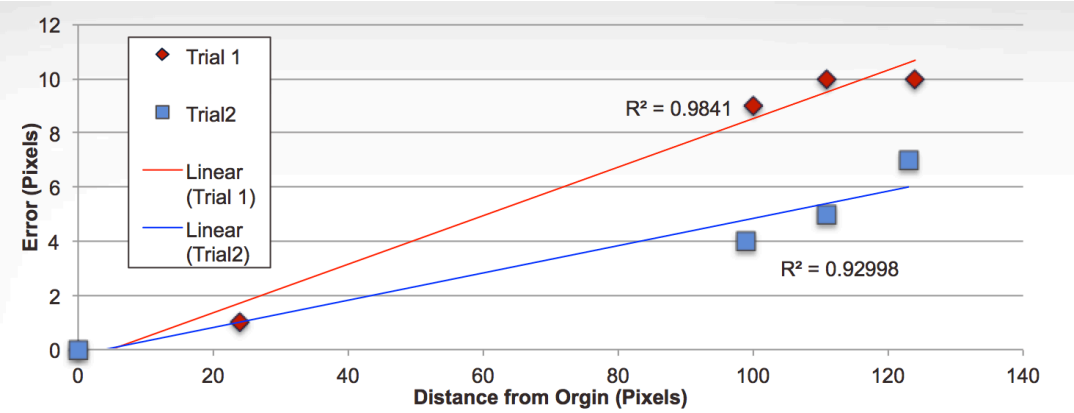


Figure 3.6: Two trials of pixel error, at a set of varying distances from the origin, along with their respective linear regression lines.

The plotted values represent each level measured from the origin pixel. Each data set also shows a linear regression line with both R-values greater than 92 percent. The max pixel error from both trials was 10 pixel and the minimum was 0. The positive trend points towards a propagation of error. For this configuration, the positive propagation may be because the calibration checkerboard in near the origin, and therefore the propagation is due to the distance from the transformation coordinates. The data from the two trials is compiled in Table 3.1 below.

Trial 1			
x-distance (pixels)	x-travel (px)	error (px)	error (m)
425	0	0	0
401	24	1	0.0106
325	100	9	0.0953
314	111	10	0.106
301	124	10	0.106
Trial 2			
x-distance (pixels)	x-travel (px)	error (px)	error (m)
429	0	0	0
330	99	4	0.0423
318	111	5	0.0529
306	123	7	0.0741

Table 3.1: Summary table of pixel error, and their corresponding error distance at each intersecting point of the MSSP overlapping area.

The maximum pixel error corresponds to a 10 centimeter error for this configuration. As stated above, the positive error trends along with distance from the origin. The propagation of error tends to point the source of error being the checkerboard detection methods. Because different trials had different initializations, the variance in propagation was different. With a more accurate marker detection method, the homogeneous transformation matrix between the two MSSPs could be more precisely determined, and hopefully, result in less error propagation.

3.2.2 Time Delay and MSSP Selector Analysis

The next component of analysis for the IEA system, was defined as the time delay, and MSSP selector reliability. More specifically than the time delay, it is the variance in loop time which can effect the vehicle's real time control capability. To capture this, each loop cycle time stamped the data transfer to the vehicle. The difference in these time stamps become the delta-t metric. From here, standard statistical analysis can give more insight about the uniformity of the system's loop time. To develop a baseline, a trial was ran without moving vehicle. This ensured there was little to no switching between the MSSPs, and there was good detection from frame to frame. With these variables eliminated, the trial looked more closely at the effectiveness of the network. The dynamic trial was ran to determine if actively switching MSSP units slowed the loop time, or effected it negatively. Figure 3.7 below shows the loop time values for the two trials described

above.

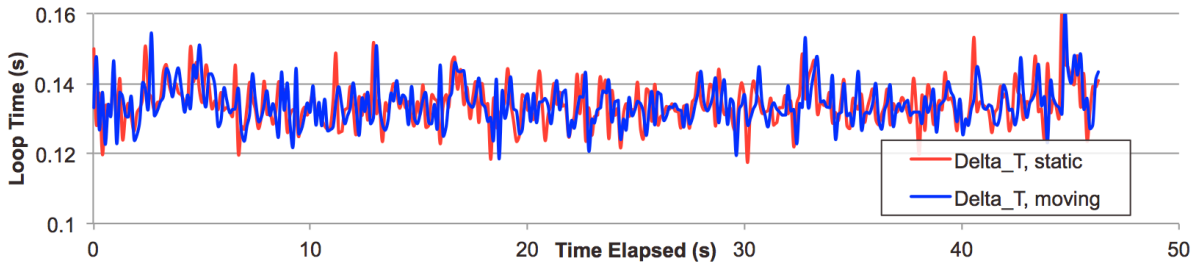


Figure 3.7: Looptimes for static and moving trials during the horizontal goal set.

The dynamic trial showed identical average loop times over the two trials (0.134 seconds). However, to inspect further, the variance between samples, and a moving average of these value overtime were calculated. The variance in loop time has the ability to quantify a lag in the IEA system. A large jump in variance could be a single lag, and an increase in the filtered data could be a delay due to buffering commands. Figure 3.8 below plots both of these outputs over the two trials.

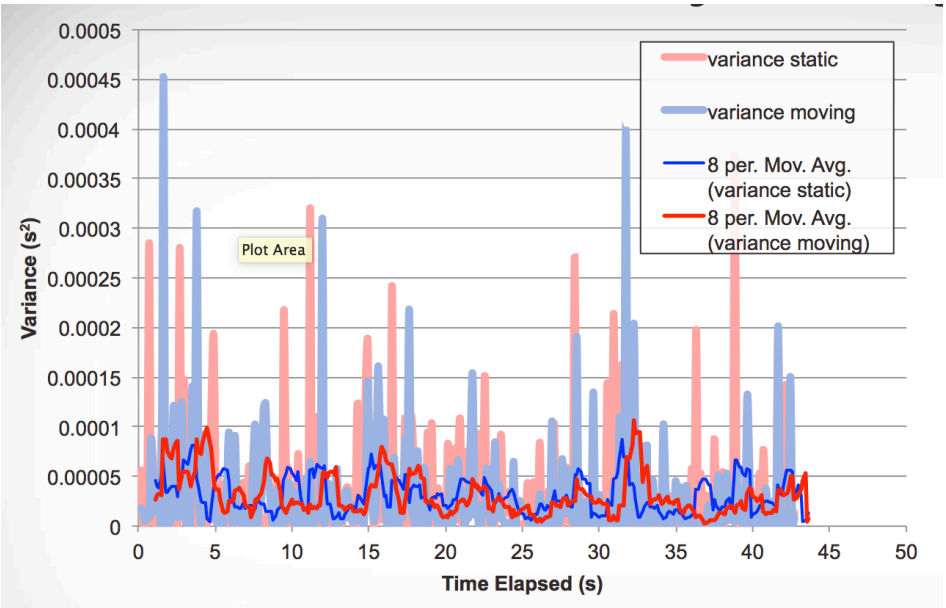


Figure 3.8: Looptime variance for static and moving trials during the horizontal goal set, and their respective 8-period moving averages.

Both trials have respective minimums and maximums, but do not seem to concatenate one after another. Therefor, the IEA is supporting the required bandwidth to operate a vehicle in the environment. The staticstical outputs are shown in Table 3.2 below. A t-test was tested at a 95 percent confidence interval.

	Mean Looptime (s)	Avg. Looptime Variance (s^2)	Looptime StdDev (s)	StdDev % Difference from Mean	T-test
Static trial	0.134	3.44E-05	5.88E-03	4.39%	Alpha = 0.05
Moving trial	0.134	3.85E-05	1.01E-02	7.54%	P-value = 0.094

Table 3.2: Summary table of statistical outputs for the static and moving trials.

The table above outlines the similarities of the static and moving mean loop times, and their variances. The standard deviation of loop time was almost double for the dynamic test. The t-test produced however, and p-value of 0.94 show that the two trials are not significantly different under a 95 percent confidence level.

3.2.3 Cross-track Error Analysis

Cross track error at each sample is defined as the orthogonal distance to the desired path (straight or otherwise) from the vehicle's current location. Using a straight path, cross track error can quantify the sufficiency of both vehicle controller and MSSP fusion. For this trial, the vertical goal setting was used to focus on the vehicle straight path control proficiency. For this trial, the IEA system continuously cycled through two goals. This required the vehicle to U-turn after reaching each goal. This trial aims to focus on the settling and control accuracy, which excludes the saturated zones during full turn arounds. Therefore, the raw data was segmented into each pass, eliminating the first over shoot for each turn around. Figure 3.9 below displays the raw, and segmented data sets.

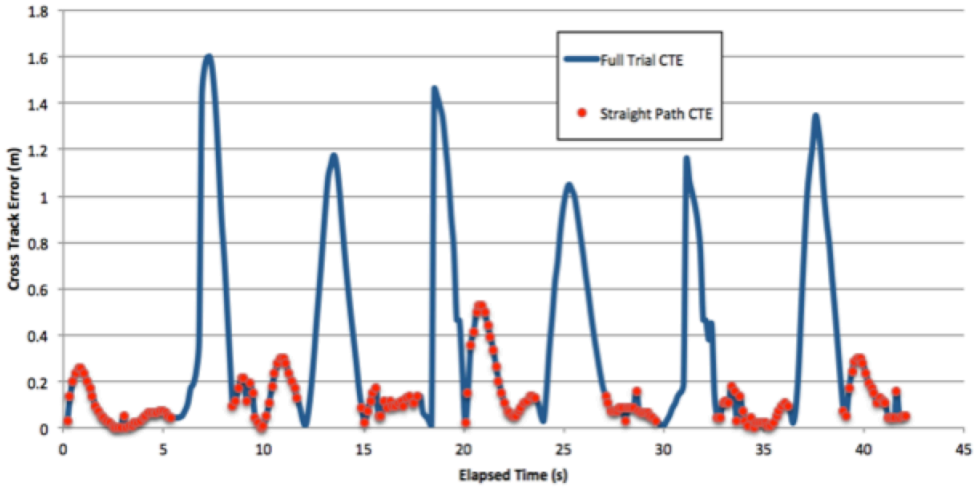


Figure 3.9: Raw and segmented cross track error of vertical goal set trial.

As stated above, the large overshoots are due to the U-turns, however, this induction of error

acts as and impulse response, and allows the controller to settle over time. Within the segmented datasets, small discontinuous jumps are the result of MSSP switching and localization error. This error could be induced during the MSSP vehicle detection, or the MSSP perspective and affine transformation processes. The average cross track error for the segmented dataset was less than 12 centimeters for the straight path. While this is satisfactory, the cross track error could be further reduced without the U-turn setup.

3.3 Research Outcomes

During the progression towards reaching the research requirements, simply stated as an operational IEA system with non-detrimental levels of error, certain specifics within the project stood out as notable achievements of operation. First, the implementation of a simple, drive-by-wire vehicle being controlled wirelessly through a distributed network of statically mounted sensors and computation proved to operate with consistent and reliable commands, with a loop time standard deviation less than 8 percent. Also, a single, simple identifier, in this case a checkerboard, enabled fused perceptions that accurately tracked, and in turn, controlled an autonomous vehicle with a small cross track error (+/- 0.12m CTE). The efficiency of the implemented A* algorithm enabled use of realtime processing for control applications with obstacle avoidance and path planning. These implementations operated on both MSSPs, demonstrating the distributed computer aspect of the IEA structure. Finally, confidence algorithms were successfully developed for effective detection hierarchy, and used in a series of selector logic for shared control over an autonomous vehicle. These tasks, collectively, enabled the success presented in this section, and the conclusions in the following section aim to tie in the overarching idea and beneficial advancements of implementing IEA systems into real world use cases.

4. SUMMARY AND CONCLUSION

The main objective of this research was to develop an infrastructure based platform, which enabled autonomous cars to safely navigate an environment. Through development of two MSSP systems and a scaled vehicle, results displaying the accuracy and reliability allowed the quantification of testing the IEA system. This section discusses the quantifiable and analytical outcomes of this research. The discussion is extended to detail drawn conclusions and the implications of using an IEA system for scaled applications. Finally, future work, final words, and back matter will complete this report.

4.1 Conclusions

The successful implementation and testing of the horizontal and vertical configurations discussed in the Results and Discussion section produces the outputs to look deeper at the outcome of this IEA system's ability to control a vehicle within its environment. The analysis performed was able to determine the pixel error, when fusing perceptions, does increase further from the point of transformation (checkerboard coordinates), however it was determined the error remained less than 11 pixels, or at this configuration's ground sampling distance, 11 centimeters. The horizontal trial accentuated the effect of handing off MSSP control, and therefor more susceptible to network interruption. The test outcome illustrated that a switching MSSP state of the IEA system showed similar loop time, but a greatly increased standard deviation. However, after performing a t-test with a 95 percent confidence interval, the static and switching MSSP state trials were not significantly different. The final testing quantified the cross track error. For a straight path, the vehicle being controlled by the IEA system had a cross track error less than 12 centimeters, meaning between the controller, network lag, and pixel error, the system was robust enough to generate stable control of the vehicle with minimal error.

This research process sheds light on several outcomes from the experiments. On a high level, the IEA is a system that, in highly congested areas of autonomous vehicles, uses less hardware,

bandwidth, energy, and money to maintain a controlled environment for a vehicle to operate in. Through the development of background detection algorithms, this research has shown the advantage of static MSSPs analyzing the same environment over time, and carrying an increased reliability from less unknowns about the area of interest. It was determined through testing that wireless commands can sufficiently operate a vehicle in a limited agent environment, and do not bottleneck the system. For this system, the computer vision process proved to be the bottleneck, limiting the pose update of the vehicle to as low as 8 Hz. This had the most detriment to the control of the vehicle.

Scaling this research to a real world implementation may be achievable, but would also generate some challenges. In all trials, the detection and tracking through use of a single or multiple MSSPs is sufficient to command stable control. Also, due to the real world calibration and initialization methods, the MSSPs would be capable of controlling full scale, drive by wire vehicles. Furthermore, the computer vision processes would hardly compound in a larger environments with more vehicles and obstacles, in terms of computational load. Independent of the quantity of objects detected, the same computer vision processes run each loop. The reliability of control in terms of cross track error, and network lag demonstrates the usability for two MSSPs and a single vehicle. However, with a crowded environment of vehicles, the path planning for each agent would become computationally expensive. While multi threading implementations help overcome these challenges, there is a point in which the cores of the CPU would be completely loaded, and updating path planning may lag. This is a concern since obstacle avoidance is dependent on continually updating the planned paths. Furthermore, there may be a limit to the number of vehicles within an environments, since real time control is dependent on consistent wireless communication between the vehicles and the infrastructure.

After processing these conclusions, and expanding the IEA idea in scale and scope, this research hints that certain applications have advantages when utilizing an IEA approach. GPS denied areas or tunnels could be outfitted with an IEA system to operate agents within a confined environment. The advantage lies in lack of reliance of a global reference frame by the IEA system for

operation. As long as there remains a limited amount of pixel error when fusing MSSP perspectives, the IEA system could prove superior to a stand alone implementation. specialized lanes are advantageous to the safety and complexity of autonomous transportation. This approach validated the idea that a simple drive by wire vehicle could be controlled in the absence of expensive and power hungry computing and sensing. The successful operation of the IEA system to this point begs the discussion that a IEA approach more efficiently implements a large number of autonomous agents, and their required infrastructure components en mass in terms of power consumption and overall cost. This platform operates with the potential to integrate multiple autonomous agents in a particular area into an Intelligent Transportation System. When agents operate on the same network, there is a consistent dissemination from MSSPs across all vehicles through the IEA system. Intelligent Transportation Systems aim to implement algorithms to increase predictability, which can lead to a safer transportation platform.

4.2 Further Study

While this research defined a scope to develop a proof of concept and implementation for the IEA system, continuing the development has its advantages. Due to the loop time dependencies, adding a local planner could offload computation time from the A* algorithm by only being called when the path is obstructed. Also, a local planner would better address the non-holonomic configuration of an automobile. Planning for particular turning radius' could increase the path tracking for curved paths, and increase reaction time for obstacle avoidance. Also to increase reaction time, deep learning object detection has the ability to classify a greater set of objects, at higher efficiencies, which in turn will improve control. Because the tracking limited the speed of this implementation, a method that operates over 15 Hz would be suggested. This would require the MSSP to house a GPU along with the CPU. This would require another iteration on the physical design to fit the size of the GPU, and meet the new power requirements. Next, moving out of the current testing environment, full scale, outdoor testing would give further insights on the deployability of system. Finally, further sensor fusion increases detection, tracking, and classification redundancy and reliability. The MSSP hardware and firmware enabled raw feeds from a thermal

camera, LIDAR, GPS, and IMU that could be utilized for a higher localization accuracy in both MSSP perception fusion, and vehicle tracking.

4.3 Closing Remarks

This research developed an end to end proof of concept of a Infrastructure Enabled Autonomy system. The testing performed during this research showed how this system can operate an autonomous vehicle safely, with non-detrimental error. This research showed that continuing to develop this system could bring full scale advantages when implemented into real world use cases, and how this could greater impact the a quickly accelrating industry. In order to maintain high standards of safty in the autonomitive industry, it is imperative the engineer continue to pursue noval, new approaches. Embodying this can be done through continued research, development, testing, and a general pursuit of learning. This research has enabled me to find a field where learning come everyday with the wide multi disciplinary requirements that come with robotics and unmanned systems. The remaining items of this document consist of the sources utilized in the bibliography, and the referenced appendix.

REFERENCES

[1] M. G. Plessen, D. Bernardini, H. Esen, and A. Bemporad, "Multi-automated vehicle coordination using decoupled prioritized path planning for multi-lane one- and bi-directional traffic flow control," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 1582–1588, Dec 2016.

[2] K. Tawara and N. Mukai, "Traffic signal control by using traffic congestion prediction based on pheromone model," in *2010 22nd IEEE International Conference on Tools with Artificial Intelligence*, vol. 1, pp. 27–30, Oct 2010.

[3] M. Hager, L. Wernecke, C. Schneider, and J. Seitz, "Vehicular ad hoc networks: Multi-hop information dissemination in an urban scenario," in *2015 38th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 65–70, July 2015.

[4] T. Wietholt and J. Harding, "Influence of dynamic traffic control systems and autonomous driving on motorway traffic flow," *Transportation Research Procedia*, vol. 15, pp. 176 – 186, 2016. International Symposium on Enhancing Highway Performance (ISEHP), June 14-16, 2016, Berlin.

[5] C. Park and J. Lee, "Intelligent traffic control based on ieee 802.11 dcf/pcf mechanisms at intersections," in *2011 IEEE Vehicular Technology Conference (VTC Fall)*, pp. 1–4, Sept 2011.

[6] M. Gerla, E. K. Lee, G. Pau, and U. Lee, "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds," in *2014 IEEE World Forum on Internet of Things (WF-IoT)*, pp. 241–246, March 2014.

[7] P. Gora and I. Rüb, "Traffic models for self-driving connected cars," *Transportation Research Procedia*, vol. 14, pp. 2207 – 2216, 2016. Transport Research Arena TRA2016.

[8] X. Zhao, K. Mu, F. Hui, and C. Prehofer, "A cooperative vehicle-infrastructure based urban driving environment perception method using a d-s theory-based credibility map," *Optik - International Journal for Light and Electron Optics*, vol. 138, pp. 407 – 415, 2017.

[9] F. Hagenauer, C. Sommer, T. Higuchi, O. Altintas, and F. Dressler, "Parked cars as virtual network infrastructure: Enabling stable v2i access for long-lasting data flows," in *Proceedings of the 2Nd ACM International Workshop on Smart, Autonomous, and Connected Vehicular Systems and Services*, CarSys '17, (New York, NY, USA), pp. 57–64, ACM, 2017.

[10] E. Ndashimye, S. K. Ray, N. I. Sarkar, and J. A. Gutiérrez, "Vehicle-to-infrastructure communication over multi-tier heterogeneous networks: A survey," *Computer Networks*, vol. 112, pp. 144 – 166, 2017.

[11] S. Gopalswamy and S. Rathinam, "Infrastructure Enabled Autonomy: A Distributed Intelligence Architecture for Autonomous Vehicles," *ArXiv e-prints*, Feb. 2018.

[12] A. Nayak, K. Chour, T. Marr, D. Ravipati, S. Dey, A. Gautam, S. Gopalswamy, and S. Rathinam, "A Distributed Hybrid Hardware-In-the-Loop Simulation framework for Infrastructure Enabled Autonomy," *ArXiv e-prints*, Feb. 2018.

APPENDIX A

MSSP TRIPOD ADAPTER PLATE.

The drawing below in Figure A.1 enables the attachment of the MSSP unit to the tripod via standardized 1/4"-20 threaded hole in the center.

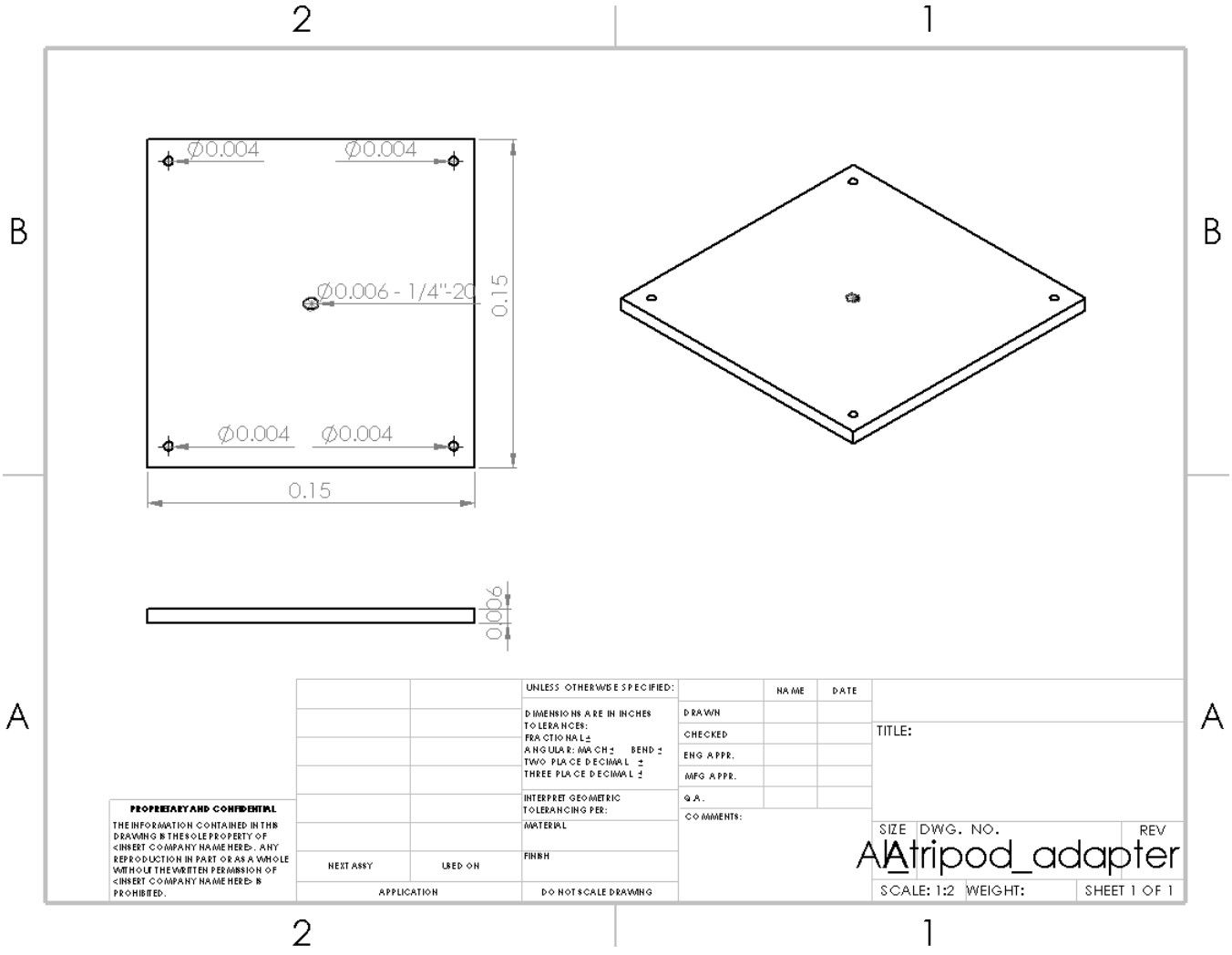


Figure A.1: Engineering drawing for the tripod adapter user for the MSSP units.