Robocart

System Framework for Machine Vision Component of an Autonomous Vehicle

Submitted by

Gabriel Isko
Robotics Engineering

Advised by

Professor Alexander Wyglinski, *Advisor-of-Record*
Professor Taskin Padir, *Co-Advisor*

September 2014–December 2015
Project Code: MQP-AW1-PATH

Abstract

This project documents a framework for an extensible, flexible machine vision software implementation for the Robocart project. It uses a distributed mobile computing framework in order to best leverage the scalability of machine vision. This process aims to improve upon current machine vision implementations in commercial autonomous vehicles, as well as provide a basis for further development of Robocart's autonomous navigation systems. This framework is tested with the use case of road detection.

Table of Contents

List of Figures

List of Acronyms

ALV Autonomous Land Vehicle

API Application Program Interface

CAD Computer Aided Design

CAM Computer-Aided Manufacturing

CMU Carnegie Mellon University

CNC Computer Numerical Control

DARPA Defense Advanced Research Projects Agency

ECE Electrical and Computer Engineering

GPIO General Purpose Inputs and Outputs

GPS Global Positioning System

GUI Graphical User Interface

IMU Inertial Measurement Unit

IN-DASH Intuitive Dashboard

LIDAR Light Image Detection and Ranging

MQP Major Qualifying Project

PWM Pulse Width Modulation

RALPH Rapidly Adapting Lateral Position Handler

RAM Random Access Memory

RBE Robotics Engineering

RC Remote Controlled

ROS Robot Operating System

RXD Receive Data

SIFT Scale Invariant Feature Transform

SSH Secure Shell

TXD Transmit Data

URDF Universal Robotic Description Format

USB Universal Serial Bus

VIAC VisLab's Intercontinental Autonomous Challenge

VPN Virtual Private Network

API Application Programm Interface

WPI Worcester Polytechnic Institute

Acknowledgements

The Author would like to acknowledge the following individuals for providing support and assistance throughout the entirety of our project. Without all of you, this project would not have been possible.

To our Advisor, Professor Alexander Wyglinski: *Thank you for meeting with us weekly and guiding us in the right direction for the duration of this MQP.*

To Professor Ken Stafford: *Thank you for helping us determine logistics for storing our wireless server within the Rec Center Loading Dock.*

To Meredith Merchant: *Thank you for providing us with access and granting us space in the Rec Center Loading Dock to work on our MQP.*

To Tracey Coetzee: *Thank you for placing part orders for the Robocart MQP in a timely manner.*

To Prateek Sahay: *Thank you for creating the CAD model of the Robocart and working with us to develop mechanical systems for the first-generation autonomous golf cart.*

To Liz Miller: *Thank you for your work in testing and systems engineering work on the first-generation autonomous golf cart.*

Executive Summary

Motivation

The promise of saving 1.2 million lives a year [1] and solving traffic congestion problems has struck a chord with scientists, engineers, and programmers around the world. As result, autonomous vehicles are expected to contribute to roughly half of the total cars produced by the early 2030s [2]. This project focuses on the implementation of a software computer vision processing system that is capable of running on an autonomous vehicle.

Proposed Design

The scope of this MQP is to give an example of a design for a software system that can be used for an autonomous vehicle in a way that is extensible for further development, and is intended to be used in a robotic system. This is achieved by creating a system that shares video data for possessing among a network. This system is designed using tools that are designed for modular expansion of a single robot and flexible processing among multiple sources.

Implementation and Experimental Results

The implementation of this system is designed to provide the frame for a system that integrates vision processing from multiple sources moving forward, and has not currently been implemented on an autonomous vehicle. Provided in this MQP are recommendations for this

systems implementation in autonomous vehicles that enable use vision.

The purpose of this system is to achieve a framework for an autonomous vehicle that uses computer vision to navigate. This system is designed to integrate vision from any IP enabled source, and is not limited by dedicated hardware stationed on a single vehicle.

Conclusions and Recommendations

Continuing with development in this framework should be in the pursuit of a system that can be an experimental test bed for using machine vision in vehicle navigation. While going forward, these recommendations that are concluded from this project:

Recommendation #1: *An autonomous vehicle that uses machine vision for navigational purposes should be agnostic to the source of its image data.*

In order to take advantage of the scalability [3] strengths of computer vision, any vehicle that uses computer vision for autonomous navigation should not be locked into a single hardware setup. Since the system put forward in this project uses networking systems to control the flow of data throughout an autonomous vehicle, providing a system for agnostic data flow will make it possible to add more data sources or even dynamically integrate potential data sources. **Recommendation #2:** *The design of an API to extend the usability of this framework needs to be considered.*

Parallel to the system proposed in this MQP's flexibility when it comes to hardware design on a single autonomous vehicle, a software framework must also be put in place in order to make software development under this framework possible. This should be achievable with the open source frameworks this project uses for image processing and for data coordination. A design that will allow future developers to employ dedicated functions and systems that encapsulate various computer vision algorithms and abilities will make it possible for developers to completely leverage the advantages of this system. One of the major advantages that can be leveraged by providing a diverse development system around this framework is that it can

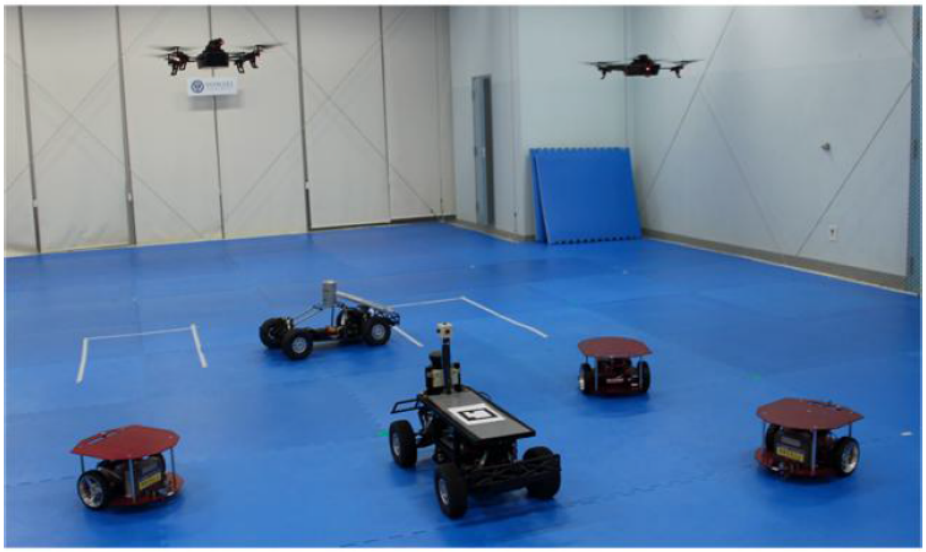

Figure 1: An example of a collaborative network consisting of ground and aerial vehicles. Data is relayed from/to the aerial vehicle to/from the ground vehicle to obtain more reliable information and greater accuracy about the vehicles' surroundings [4]. This network is very well suited to using agnostic video sources because it opens up for much more flexibility in its configuration - multiple combinations of these vehicles cooperating together can be achieved without defining new standards for each new vehicles video stream.

take advantage of the internet. Developers can add improvements to an autonomous vehicle through software updates delivered online, rather than having to install new hardware in order to improve an autonomous vehicle.

Recommendation #3: *More dedicated hardware should be used for critical real time performance and safety functions.*

While it is important for data to be shared across a network to take advantage of the higher level aspects that can be gained from computer vision, under this framework it is still possible to use data streams from local sources. Maintaining the flexibility of this setup is important for critical safety functions that could be optimized through dedicated hardware. While this dedicated hardware is not explored in this project to avoid pre-mature optimization, the approach of optimizing certain vision functions might be necessary for applications where real time performance, safety, and security is a risk.

0.1 Sample Results

Using an approach that relays data from a local camera over a network interface to a ROS node that performed a road detection test, this method was proven viable. A ROS node was able to get consistent different readings on the mean green color value when looking at a road or a lawn. This information can be used in an autonomous vehicle to gather information about the location of a road that it is on and be integrated into navigation decisions.

Limitations of this Research

The scope of this project's research is limited by the following factors:

1. While a current test bed for an autonomous vehicle is being developed through other MQPs at WPI, it is not currently ready yet - this research still needs to be tested on an autonomous vehicle.

2. Requirements of passengers and future developers of autonomous vehicle should be taken into account as this project is expanded upon.

Potential Uses of the Recommendations

This project establishes a beginning framework for the software design and approach for an autonomous vehicle. It is a place to start at, and provides examples of possible test cases of the research in action. Further efforts in designing an usable framework should be expanded upon. The development, implementation, and maintenance of a software package that can be used to establish this framework in order to create an autonomous vehicle is a good next step for a future MQP.

Chapter 1

Impetus for a New Autonomous Vehicle

1.1 Establishment

Autonomous vehicles are large and complex projects that are currently only accessible to efforts with large capital and engineering resources. It is a problem that current efforts depend on a proprietary solution. Mobile Eye is a company that provides solutions to autonomous car sensing systems, such as collision warning, pedestrian detection, and lane tracking. Mobile eye is not available to general consumers or researchers, and is instead targeted at car makers and corporate engineering efforts such as Tesla and Google. Proprietary solutions such as mobile eye are not accessible to researchers who are striving to work on new autonomous vehicle techniques. Instead, an autonomous vehicle must be built on open standards. Problem Identification: Image processing is a powerful tool in the development of an autonomous vehicle that is readily available to researchers. Image processing can be used to solve problems such as object recognition that is normally done by the driver in a non-autonomous vehicle. Open source libraries such as openCV make image processing accessible, and provide a scalable solution that does not depend on specialized hardware such as a LIDAR. However mobile computing, despite recent advancements, is still a long way from providing a commercially available solution for the amount of image processing needed for an autonomous vehicle. Current mobile

computers that exist at a research level are not powerful enough to leverage the full potential of computer vision, as its scalability has the potential to outstrip computers that could exist on a vehicle.

1.2 Scope

A solution to the lack of specially designed hardware that is commercially available is to split the computation between multiple mobile computers. Ethernet IP is a widely used open standard to share data between computers, and can provide a way for computers already on the market to share computed data and distribute the computation required for computer vision, or to deliver data to dedicated processors on a non - mobile network. The processing of video data should be agnostic to its source in order to provide the greatest flexibility in terms of processing it on a network.

1.3 Research Gap

Efforts to create open source solutions to address this issue are underway, but are still in their infancy. In order to create systems that are coordinated over an Ethernet network, systems like ROS exist to provide a way to organize computational clusters into separate nodes that can exist on different locations in an IP network. However, a solution for ROS to apply camera data to autonomous vehicle solutions does not currently exist.

1.4 Goals

The goal of this project is to provide a solution to employ a solution to use the rocon_rstp_camera_relay package to provide image data under a ROS framework in order to develop a solution to autonomous vehicle functions in ROS. This project will investigate the applications of such a

solution, and explain and provide examples of usage for an autonomous vehicle. The implementation of road detection will be used as a case study in how a distributed vision processing solution would look in practice.

1.5 Project Objectives

The objectives of this project is to propose and provide example implementations of a design for the software component of a vision processing system for Robocart. The goal of this system is to provide the basis for future development of Robocart. The objectives of this system are the following:

> **Objective #1:** *Make it easy to integrate multiple forms of hardware as development on Robocart continues.*
>
> **Objective #2:** *To use agnostic video sources in order to make the location of vision calculations flexible.*
>
> **Objective #3:** *To use tools and practices that are designed to represent the structure of Robocart in software.*

1.6 Report Structure

The rest of this document provides the documentation of the vision system for the Robocart, as well as an example of it's implementation. A description of the proposed framework is contained in chapter 2, which goes over its design. Chapter 3 contains the implementation of the framework. Chapter 5 presents the conclusions that can be drawn from this project.

Chapter 2

Literature Review

Autonomous vehicles have been in development for over 65 years [5]. In 1948, cars were introduced to the first cruise-control systems [5]. By 2030, cars are expected to be fully autonomous and driver less. The promise of saving 1.2 million lives a year [1] and solving traffic congestion problems has struck a chord with scientists, engineers, and programmers around the world. Thanks in part to advances made in computing technologies over the past 30 years, inexpensive sensors, reliable object recognition, and real-time, portable, large-scale data analysis has become a reality. Inspired by ongoing research today from around the world, this MQP aims to explore autonomous vehicle technology from the perspective of modern vision algorithms combined with affordable sensing technology.

Several other labs have demonstrated the viability of autonomous cars in general, such as those of Google and the Autonomous Systems Laboratory at the University of California, Santa Cruz, but these systems use expensive, bulky, and ungainly roof-mounted Light Image Detection and Ranging (LIDAR) detectors [6]. The mission of this MQP is to explore vision-based systems for autonomous vehicles that use sensors in a manner that is similar to the way a human perceives his/her environment. The VisLab of Parma University in Italy, as well as several others, have demonstrated the viability of this option [6].

Autonomous vehicle research has exploded in past decades, due to increased fascination

with driverless vehicles and the impact they can have on society. Cars today come with options for adaptive cruise control, lane detection, and automated parallel parking. Further advanced autonomous vehicles blend human-control with autonomous systems, and as result, have the ability to control brakes and alert drivers of dangers [7]. Tracing the origins and historical discoveries of autonomous vehicle technologies leads us to the basis for the Robocart MQP.

2.1 Major Milestones in Autonomous Vehicles

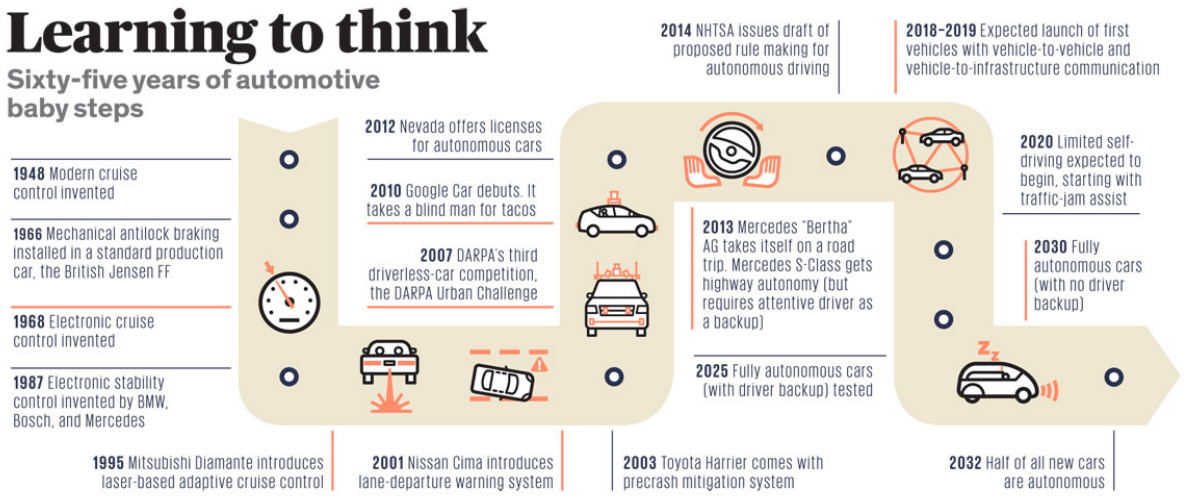

Figure 2.1: This timeline depicts the development of autonomous vehicles from 1948 to today. Starting with the invention of cruise control in 1948, vehicles are becoming more autonomous and less operator-dependent. It is expected that by 2032, half of all new cars will be fully autonomous [5].

Early on, fully-autonomous vehicles (ones that did not rely on devices embedded into roads) were few and far between. Before Martin Marietta, in conjunction with several of the research facilities and funded by Defense Advanced Research Project Agency (DARPA), introduced the Autonomous Land Vehicle (ALV) Project in 1985 [6], [7]. Martin Marietta's ALV used computer vision and laser scanning for sensing and six server racks for path correcting. It successfully traveled a half mile on an empty road in 1985, but was notoriously fickle and easily tricked by shadows and small variations in lighting [8]. During the same time period, Ernst Dickmanns

in Munich introduced saccadic vision and Kalman probabilistic filters for use in autonomous vehicles [6].

A decade later, in 1995, Carnegie Mellon developed the Rapidly Adapting Lateral Position Handler (RALPH) which used computer vision to determine the location of the road ahead to autonomously steer a car as two researchers controlled the throttle and brakes [6], [9]. Dean Pomerleau was able to "teach" an artificial neural network to drive the car (it learned to use the grass as boundaries) and was able to successfully drive on a highway at 55 mph [10]. Researchers from Carnegie Mellon were able to use this software, in a project called computer vision and No Hands Across America, to drive an autonomous car from Pittsburgh, PA to San Diego, CA for over 98% of the journey [6], [9]. By this point, autonomous path planning was in existence, but there were still many issues to be resolved before a car could actually drive itself.

DARPA also hosts a Grand Challenge prize competition for autonomous vehicles. Competing universities and organizations build autonomous vehicles to race through the competition field. This competition is over ten years old and has helped set the foundation for autonomous vehicles used in research, today. For example, the DARPA Grand Challenge in 2005 challenged universities to have driverless cars traverse a 132 mile-long off-road driving course in the Mojave Desert [11]. The competition was actually the second of its kind—the first DARPA Grand Challenge in 2004 was unsuccessful [10]. The competitors again took a wide number of approaches, utilizing combinations of Global Positioning Systems (GPS), radar, LIDAR, computer vision, sonar, and machine learning to navigate a trafficless desert course at speeds up to 25 mph [6]. The winning autonomous car, Stanley of Stanford University, used machine learning to distinguish errant sensor readings. Stanley was equipped with sensors that could detect bumpiness, differing light conditions, and accounted for them using probability distributions. Essentially, Stanley could calculate the accuracy of its readings and make fewer errors—only about 1 error in 50,000 readings [10].

Four years later, in 2010, the VisLab from the University of Parma in Italy constructed a

fully-electric autonomous vehicle that embarked upon and completed a VisLab Intercontinental Autonomous Challenge (VIAC): an 8,000 mile road trip from Parma to Shanghai [1]. Refer to the prototype vehicles in Figure 2.2. Throughout the journey, the vehicle encountered a variety of traffic, road, and weather conditions [6]. Unlike cars from the DARPA Grand challenge, the VisLab vehicle largely relied on image processing for local mapping. Other sensors on board included laser-scanning and GPS, but the lasers were mainly used for detecting terrain [1]. VisLab proved the reliability and viability of vision algorithms rather than the use of complex sensors—more akin to the way humans navigate.

Beginning in 2011, Google started a self-driving car project that leveraged their mapping technology in order to navigate roads (see Section 2.3.1). This prompted the Nevada Department of Motor Vehicles to issue the first Driver's License for an autonomous vehicle. Along the way, Google discovered more challenges with autonomous driving, including the need to program behavior for moving through a four-way intersection and for city driving.

In 2014, Volkswagen implemented the AdaptIVe Project with the objective of creating autonomous vehicles that function in various levels of traffic and different driving scenarios. Specific goals include navigating a traffic jam, parking in a parking garage, and creating a robot taxi.

Autonomous vehicles of today, and even within the next 20 years [2], have high potential to provide intuitive and innovative solutions to everyday transportation needs. Autonomous vehicle research today can be grouped into three distinct sectors—government regulations, motivation of automobile manufacturers, and motivation within research institutions.

Deeva, the new VisLab autonomous vehicle

Vehicle for VIAC, Vislab Intercontinental Autonomous Challenge

BRAiVE, VisLab's first x-by-wire vehicle

Vehicle equipped by VisLab for pedestrian detection in urban areas

Vehicle for driving tests under the influence of drugs, alcool, medicines

Figure 2.2: VisLab has many prototypes for testing autonomous navigation and path planning. As shown above, majority of the autonomous vehicles are sedan styles, whereas a the Vehicle for the Vislab Intercontinental Autonomous Challenge is more similar to a van. What separates VisLab from some of the other organizations, is their focus on algorithms and cross-vehicle compatibility. In the future, I expect to see enhanced vehicle autonomy influenced by VisLab's algorithms.

2.2 Government Regulations and Issues with Autonomous Vehicles

The lack of consistent laws regarding autonomous, driverless vehicles in the United States is the main barrier hindering widespread implementation. Due to the fact fact that automated driving is a new technology, it requires legislative and regulatory action from federal and local governments. Figure 2.3 shows which states have passed, rejected, or considered implementing autonomous vehicle laws.

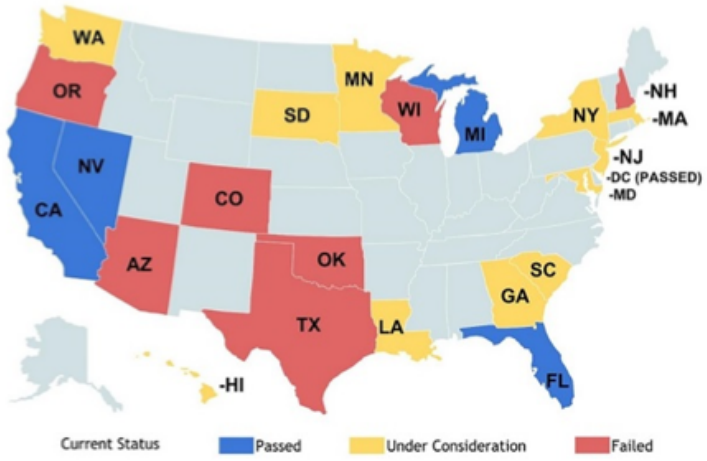

Figure 2.3: Majority of the states in the U.S. have not considered passing laws for autonomous vehicles. Of the states that have filed bills, only four states have passed: California, Nevada [12], Florida, and Michigan. In order to promote the development of autonomous vehicles, legislatures will need to consider passing autonomous vehicle laws and incentives for use on public roads and interstates.

Autonomous, driverless vehicles provide a multitude of benefits such as:

- Faster reaction time for decision-making
- Increased safety by eliminating driver error and thus reducing the number of traffic fatalities and crashes
- Reduced costs from accidents and damages

- Reduced congestion at peak-hours
- Greater fuel economy
- Greater mobility for handicapped drivers

Downsides of autonomous vehicles include policies that need to be developed around their operation. As our vehicles get smarter, complex questions such as *"Who is at fault in the event of a crash?*" or *"If the vehicle is lost or stolen, is there enough encryption to protect the information on board?*" are asked by policymakers in order to form ideas about how to mandate autonomous vehicle use. Even though there seems to be more benefits than barriers of autonomous vehicles, only four states, (California, Nevada, Michigan, and Florida), have passed bills allowing the use of autonomous vehicles on roadways [13]. As time progresses, and automakers' innovations become more reliable, governments will have to modify their policies in order to keep up with the modern technology. Autonomous vehicle development has been driven by both technology companies (such as Google), the automotive industry, and academic research. The next sections delve into the motivation behind such development.

2.3 Computer Vision Systems in Autonomous Vehicles

The Autonomous vehicles of today are starting to integrate computer vision in order to perform various tasks. The following Sections 2.3.1-2.3.2, provide insight about the current solutions for computer visions employed in road ready vehicles, as well as other autonomous mobile robotic systems.

2.3.1 Mobileye Vision system

The current solution for vision in commercially produced vehicles comes from the company Mobileye [14]. Mobileye makes a suite of vision tool that integrates vision data with other data input streams such as radar in order to perform multiple driving and safety functions,

including pedestrian collision detection and lane switch detection. Mobileye's system also provides various functionality beyond computer vision. Radar fusion is a big feature of Mobileye's system, which is the ability to combine radar data with camera data to provide up to date information about surrounding obstacles. The feature was first used in the Volvo S80 in 2007[14]. This solution was used to generate collision warnings in the S80, using radar for range detection and computer vision for angle detection. This marked the first deployment of EyeQ, Mobileye's custom hardware chip that is designed to integrate sensor data to provide various roadside driving functions. Figure 2.4 is a hardware layout of the chip:

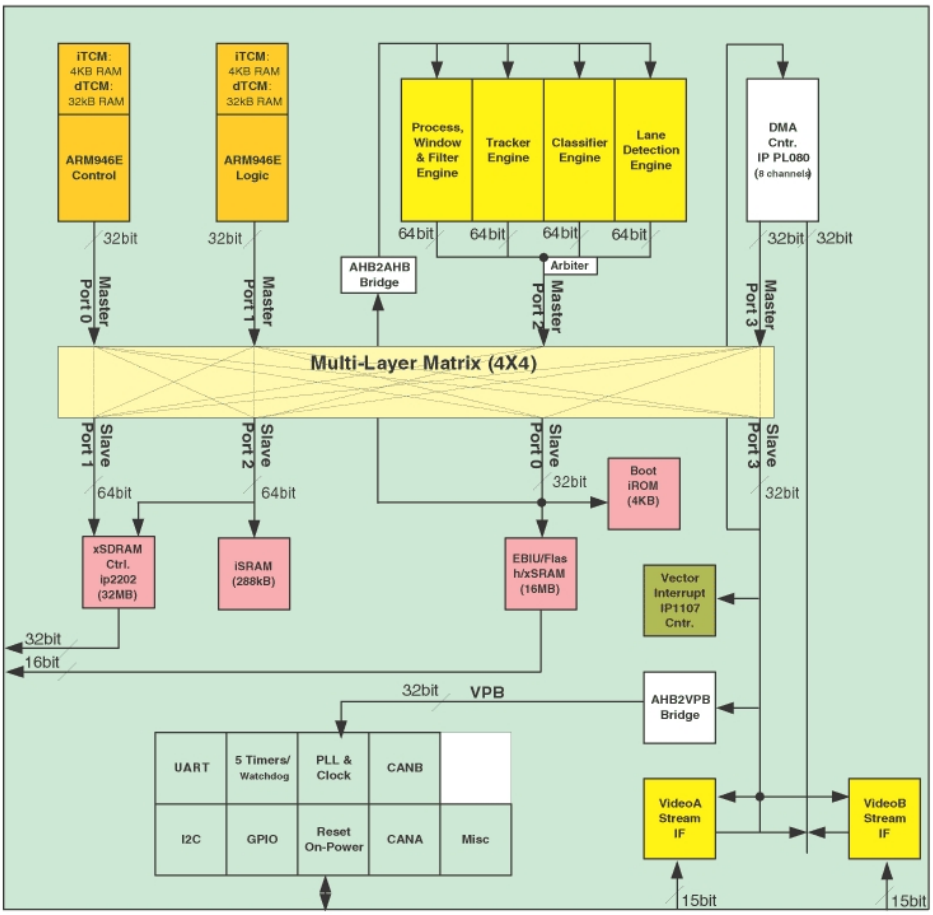

Figure 2.4: The hardware layout of the EyeQ system allows it to directly integrate sensor data in order to generate collision warnings. Sensor inputs and integration is tied to dedicated hardware components, making it impossible to extend the system to have more camera inputs. Instead, Mobileye adds more functionally by releasing future revisions of EyeQ

Mobileye's vision system is strictly a monovision, or one camera, system. The EyeQ system

is programmable, but is also designed to contain an entire application directly on the chip. It is not designed to connect to or provide data to any external processors.This design is built around taking advantage of native processing on the EyeQ system to provide for all of the Vision calculations that could be used for the autonomous vehicle. Native functions included on the chip include a classifier engine, a tracker engine, a Lane detection engine, and a window preprocessing engine. All vision functions on the chip have to use one of these features.

2.3.2 Advances in Vision integration in Robotic Systems

Since the introduction of machine vision into autonomous vehicle's on the market, advances have been made in the area of using integrated Machine Vision in Robotic Systems. Large strides have been made in using multiple streams of vision data for a single application. Researcher Enric Cervera in particular has put forward an implementation for a video source agnostic system for multiple networked robots equipped with video cameras called Agnevis [3]. The Design purpose of Agnevis is to Agent based Networked Vision. Agnevis uses the Ethernet IP protocol to share data from different video sources across multiple robots. As displayed in figure 1.5.

The design of Agnevis leverages 3 things to make full use of the scalability of computer vision. the Real time protocol for Ehternet transmission, which makes the real time transmission transmission of video data to different parts of a network. the real time protocol is a comprise between traditional Ethernet transmission protocols such as TCP and UDP, which respectively are too slow, and too unreliable for video transmission. The real time protocol is what allows for agnostic data sharing between different actors in it's network. This avoids the problems encountered by current solutions on the market since vision data can be pulled from multiple sources. This is what provides extensibility to Agnevis - it can work with multiple hardware setup

The second thing it leverages is the native power of C/C++ vision libraries. It does this to gain fast vision processing. Although not optimizing vision calculations in hardware, C/C++ vision

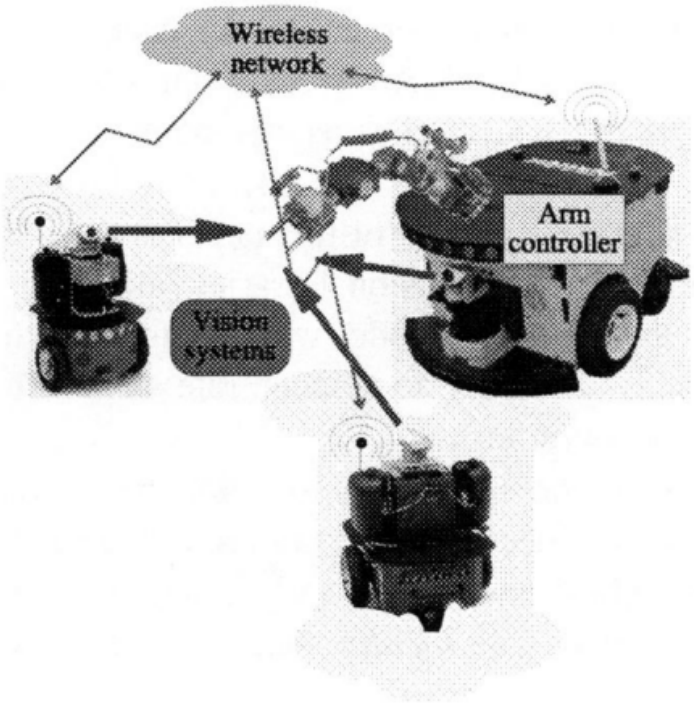

Figure 2.5: Agnevis is designed to network multiple robotic agent's video sources in order to perform complex tasks using computer vision [3].

calculations were chosen by Cervera due to them "providing an excellent, robust toolset for the development of more complex algorithms." [3]. These libaries are a good compromise for both fast calculation as well as the robust application of computer vision algorithms that are currently lacking in commercial applications.

The last tool that Cervera employs in the development of Agnevis is the Java media framework, which is what the controller software for Agnevis were written in. The Java media framework, allows for controller classes to both integrate video data and run on multiple types of hardware. In order to make this implementation work, software written in the Java programming language had to be developed specifically for this application. Cervera's research delves into his implementation of this software.

2.4 ROS Overview

ROS is an open source software system designed to coordinate multiple parts of a single robot [15]. Each part of the robot is represented in ROS as a "Node". This node structure is ideal for imposing modular design, since each node is self contained. By using ROS, all the principles that were put forward in Agnevis [3] can be attained.

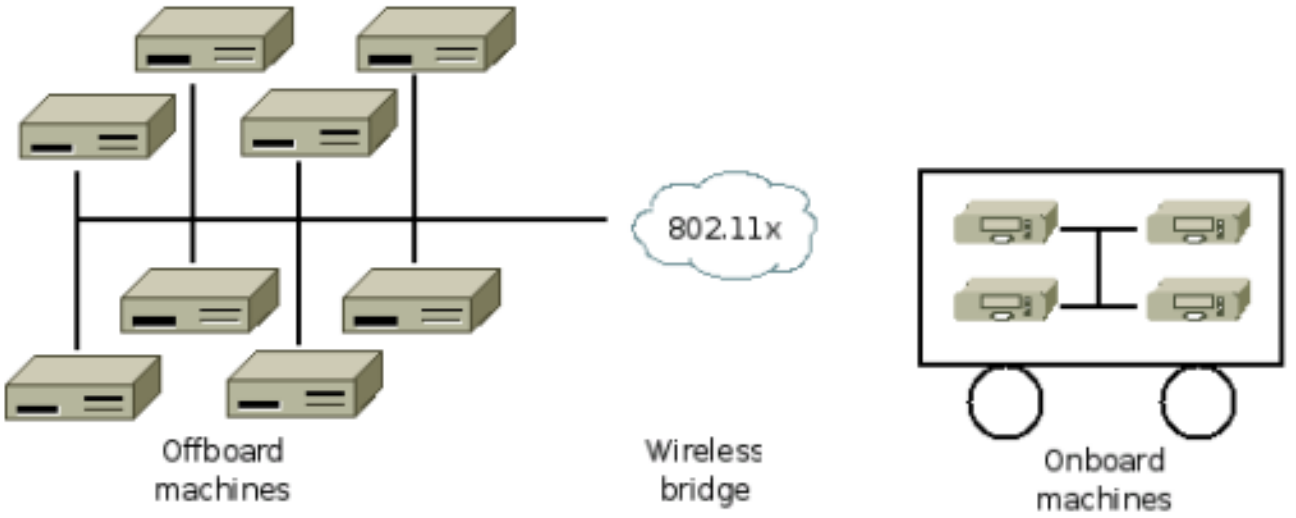

Figure 2.6: This picture shows an example of hardware implementations of ROS nodes. Nodes can either communicate with each other locally or over a network.

Information is passed between ROS nodes through topics. Topics act as information broadcasts that can be subscribed to through other nodes. For this framework, two nodes have to be defined - one that relays camera data over a ROS topic, and a node that subscribes to the ROS topic in order to decode and compute the information. Because topics can be subscribed to from any Node, it acts as an Agnostic data stream, fulfilling another goal from Agnevis.

2.5 Computer Vision and Image Processing

Computer vision is the study and implementation of computers to analyze imagery from data gathered from the real world and process it into a numerical representation. Similar and related fields include signal processing and artificial intelligence. Vision analysis can come in multiple forms: 2D image analysis, extracting 3D information from 2D images, object recognition, or transforming images through the use of mathematical filters. It is a technology used in applications such as controlling manufacturing processes and inspection [16], navigation in autonomous vehicles [17], and in human computer interaction. Image processing relies on the quick application of algorithms to image data. In ROS, support for these algorithms is provided through OpenCV. OpenCV acts as an open source interface for implementations of various vision algorithms.

The ultimate goal of the computer software when processing data from the cameras will be to perform calculations relevant to autonomous vehicles. For instance, by using triangulation [18] we can find the position of a point in space based on the two pictures. This method involves projecting rays from each camera to a point that is included in both pictures. Once the rays are computed, an intersection can be found between them, which corresponds to the position of the point. Refer to Figure 2.7 for a visual of triangulation.

A method for computing this intersection is given in [18] that involves computing camera matrices, and then using them to construct the rays to two similar points in the camera image based on their 2D position. Each line is represented by a parametric equation and the distance between the points on each line corresponding to each camera is minimized. The absolute minima of this will be the point in which the rays intersect, or come closest to intersecting.

In order to employ this algorithm, similar features must be found in the images from the two cameras in order to indicate that they indicate a single point in 3D space. This can be done using common feature matching algorithms. One such algorithm is the Scale Invariant Feature Transform (SIFT) algorithm [19]. SIFT is able to match features of objects by estimating the gradient position and orientation of locations on different pictures and matching key locations

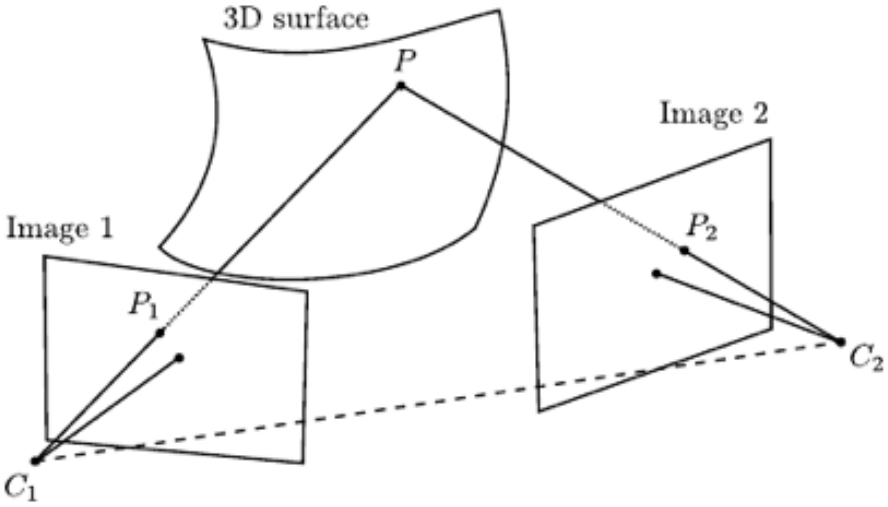

Figure 2.7: Stereo reconstruction consists of emulating human vision by using the differences between two images focused on the same scene. Each image is captured from a different point of view; triangulation and geometry then can be used to infer 3D information about the surroundings.

where they are similar (See Figure 2.8). By doing this, it is able to match locations on 3D objects with 2D images. OpenCV includes an implementation for using the SIFT algorithm [20].

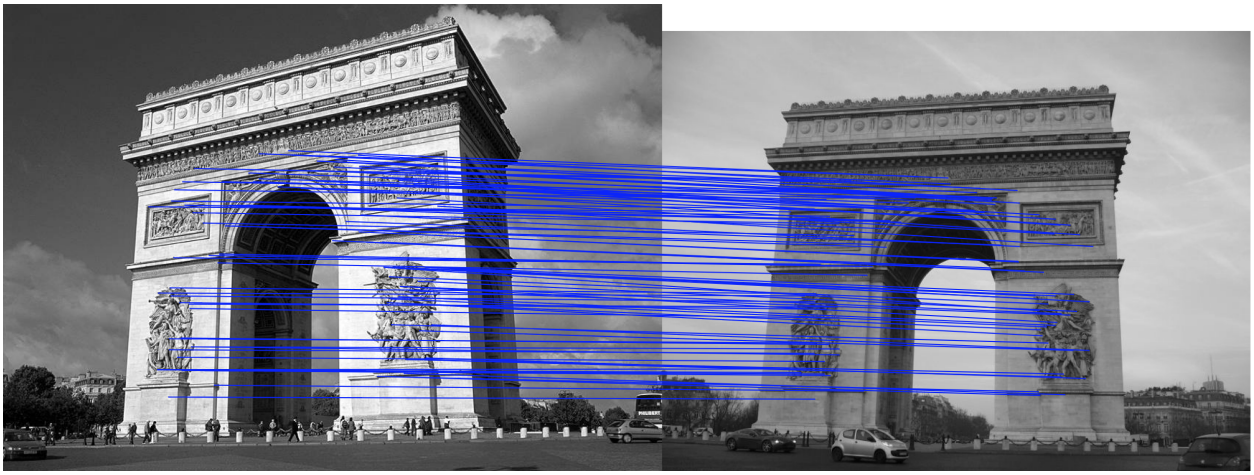

Figure 2.8: Two separate photographs of the Arc De Triomphe were taken as shown above. The blue connecting lines show that SIFT algorithm can find and recognize the same features of each photograph shown using the blue connecting lines.

Chapter 3

Implementation

3.1 System Overview

The vision design proposed by this MQP is as follows: A central server receives image data over the RTP protocol from multiple image camera sources. A ROS node running on that server relays the RTP video streams to a ROS topic, which both local ROS nodes running on the server and ROS nodes running on other locations of a network that the server is on can subscribe to that ROS topic for video data. Calculations based on that video data can be made from each ROS node. A visual representation of this framework is depicted in Figure 3.1.

3.2 System Configuration for Video streaming

For video streaming, the node $rocon\_rtsp\_camera\_relay$ is used. This node is a prebuilt node for ROS that is part of a series of nodes designed for robots that work together. This node is able to take IP enabled camera sources and relay them into a ROS image topic. The Bash commands that are used to launch this can be found in Appendix A section A.1

The $roslaunch$ command for $rocon\_rtsp\_camera\_relay$ first takes the argument of an IP camera's URL, and is used to start the ROS node that will relay camera data to a ROS topic named $RTSP$. It is important to take into consideration that this node converts data from the

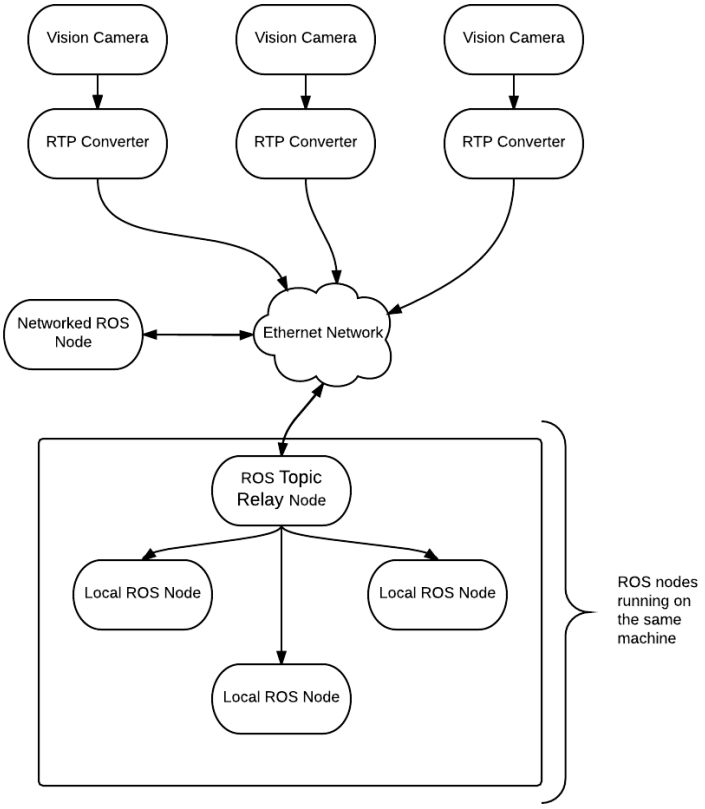

Figure 3.1: In order to take advantage of sharing video data across a network, the approach should be extensible, optimized for a central machine, but also be able to support distributed computing. This chart shows an approach that is capable of those requirements.

RTP format into a ROS image format that it then transports over a network. To get the best performance and to preserve the advantages of using RTP data transfer, this node should be run locally to the nodes that are performing the bulk of the vision calculations. However, this setup allows for image data to be received over an ethernet network in order for additional calculations to be done on remote ROS nodes.

3.3 Image Calculation

Once there is a ROS topic with relevant image data, a node can be created to subscribe to that data and perform calculations on it. The code to create this subscriber can be found in Appendix A section A.2

This code block is an example of a node that is able to turn a received message into an OpenCV image. As an open CV image, multiple calculations can be performed on it, including a sift algorithm for feature recognition, or simple color calculations.

Chapter 4

Results

4.1 Network Framework in Practice

In order to test the feasibility of this network framework, it was run on a laptop using an RTP IP video stream from the laptop's web cam. The laptop running ROS and the designed ROS nodes, was put to the test of detecting whether or not it was viewing a street, as opposed to a lawn. This use case is important for autonomous vehicles since they will have to adhere to street guidelines, navigate turns, and make sure that are able to stay on a road if they are a road vehicle.

4.2 Testing Method

A use case was implemented by aiming the camera both at a road and lawn. Video streams from these are pushed through the framework for vision defined by this MQP and then a test is performed to see if the camera is looking at a street or at a lawn through color detection. Two sample images of a road and lawn that were taken by the camera are included in this section. Figure 4.1 is a picture taken of a road, while Figure 4.2 is taken of the lawn. Both were taken from above, pointing down as if the camera were mounted to an autonomous vehicle.

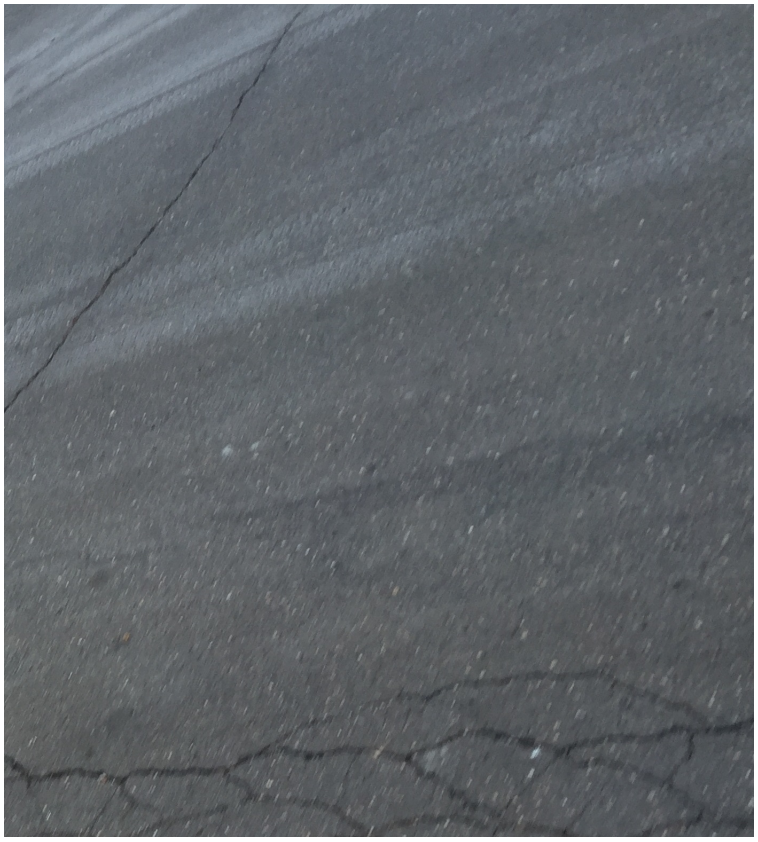

Figure 4.1: A picture of a road. This image includes variances in the road, such as cracks and wear. These are expected to be encountered on drivable roads, and a road detection test should account for them.

4.3 Results

Each image was able to be successfully pushed through the stack and delivered to a ROS node running a color detection test that isolates the green colors in an image and then computes the mean RGB values of the image. For the image in Figure 4.2, the mean green value after isolation came out to 50.240625, while for Figure 4.1 was at 2.006485. Throughout testing, images of lawn stayed consistently over a meant green color value of 38.2, while images of roads stayed below 7.6. For the code used to perform this test, please refer to Appendix A section A.3

Figure 4.2: An image of a lawn. After being pushed through the vision stack, a script should be able to detect that this is a picture of a lawn, and an autonomous vehicle should avoid it.

4.4 Discussion

These results show that is is possible to use this framework in order to relay images within an autonomous vehicle. One of the constraints of these tests is that it only tested the framework as an implementation on a local machine. However, although it does not completely test the flexibility of this approach, it is an ideal running condition for it. This use case also acts as a real time test, and it does prove that this method is useful for a real time application. This is important for when this framework is implemented in an autonomous vehicle as it will have to deliver camera data and make calculations in real time. The road detection test can be replaced by other methods of detection including pattern matching for pedestrians or road signs. however as the complexity of these tests grow, computational time can become an issue. This can be combated by performing multiple calculations in parallel by doing them on nodes that are running on separate machines on the network.

4.5 Chapter Summary

This framework was tested with the use case of trying to detect whether a road or a lawn was in a picture from the camera in real time. Images from the camera were relayed to a ROS node that calculated the mean green color value in the image. Based on this mean value, a threshold test can be performed in order to decide whether there is a lawn or not in the picture.

Chapter 5

Conclusions and Recommendations

The scope of this MQP is to give an example of a design for a software system that can be used for an autonomous vehicle in a way that is extensible for further development, and is intended to be used in a robotic system. This is achieved by creating a system that shares video data for possessing among a network. This system is designed using tools that are designed for modular expansion of a single robot and flexible processing among multiple sources.

Implementation and Experimental Results

The implementation of this system is designed to provide the frame for a system that integrates vision processing from multiple sources moving forward, and has not currently been implemented on an autonomous vehicle. Provided in this MQP are recommendations for this systems implementation in autonomous vehicles that enable use vision.

The purpose of this system is to achieve a framework for an autonomous vehicle that uses computer vision to navigate. This system is designed to integrate vision from any IP enabled source, and is not limited by dedicated hardware stationed on a single vehicle.

Conclusions and Recommendations

Continuing with development in this framework should be in the pursuit of a system that can be an experimental test bed for using machine vision in vehicle navigation. While going forward, these recommendations that are concluded from this project:

Recommendation #1: *An autonomous vehicle that uses machine vision for navigational purposes should be agnostic to the source of its image data.*

In order to take advantage of the scalability [3] strengths of computer vision, any vehicle that uses computer vision for autonomous navigation should not be locked into a single hardware setup. Since the system put forward in this project uses networking systems to control the flow of data throughout an autonomous vehicle, providing a system for agnostic data flow will make it possible to add more data sources or even dynamically integrate potential data sources. **Recommendation #2:** *The design of an API to extend the usability of this framework needs to be considered.*

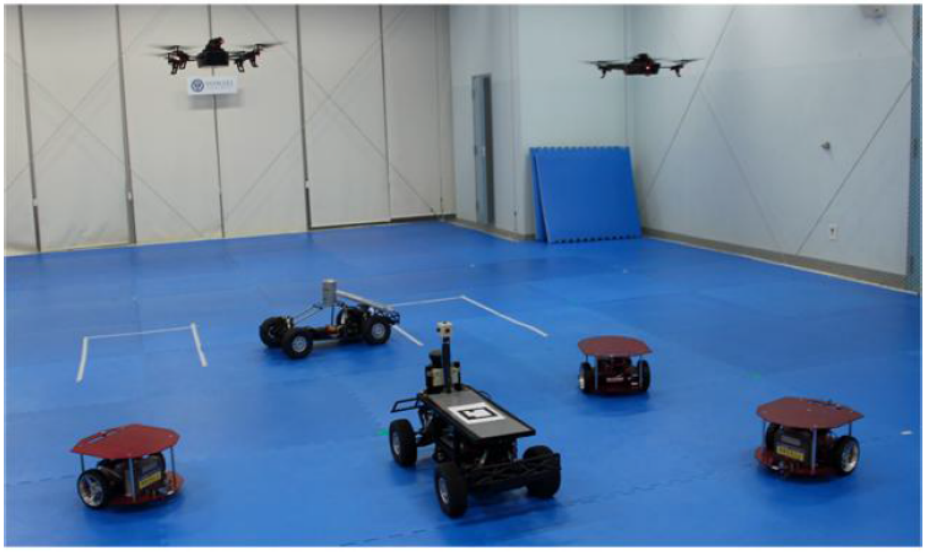

Figure 5.1: An example of a collaborative network consisting of ground and aerial vehicles. Data is relayed from/to the aerial vehicle to/from the ground vehicle to obtain more reliable information and greater accuracy about the vehicles' surroundings [4]. This network is very well suited to using agnostic video sources because it opens up for much more flexibility in its configuration - multiple combinations of these vehicles cooperating together can be achieved without defining new standards for each new vehicles video stream.

Parallel to the system proposed in this MQP's flexibility when it comes to hardware design on a single autonomous vehicle, a software framework must also be put in place in order to make software development under this framework possible. This should be achievable with the open source frameworks this project uses for image processing and for data coordination. A design that will allow future developers to employ dedicated functions and systems that encapsulate various computer vision algorithms and abilities will make it possible for developers to completely leverage the advantages of this system. One of the major advantages that can be leveraged by providing a diverse development system around this framework is that it can take advantage of the internet. Developers can add improvements to an autonomous vehicle through software updates delivered online, rather than having to install new hardware in order to improve an autonomous vehicle.

Recommendation #3: *More dedicated hardware should be used for critical real time performance and safety functions.*

While it is important for data to be shared across a network to take advantage of the higher level aspects that can be gained from computer vision, under this framework it is still possible to use data streams from local sources. Maintaining the flexibility of this setup is important for critical safety functions that could be optimized through dedicated hardware. While this dedicated hardware is not explored in this project to avoid premature optimization, the approach of optimizing certain vision functions might be necessary for applications where real time performance, safety, and security is a risk.

Limitations of this Research

The scope of this project's research is limited by the following factors:

1. While a current test bed for an autonomous vehicle is being developed through other MQPs at WPI, it is not currently ready yet - this research still needs to be tested on an

autonomous vehicle.

2. Requirements of passengers and future developers of autonomous vehicle should be taken into account as this project is expanded upon.

Potential Uses of the Recommendations

This project establishes a beginning framework for the software design and approach for an autonomous vehicle. It is a place to start at, and provides examples of possible test cases of the research in action. Further efforts in designing an usable framework should be expanded upon. The development, implementation, and maintenance of a software package that can be used to establish this framework in order to create an autonomous vehicle is a good next step for a future MQP.

References

[1] J. L. Kent, “Driverless van crosses from Europe to Asia,” 2010. [Online]. Available: http://edition.cnn.com/2010/TECH/innovation/10/27/driverless.car/index.html?iref=allsearch

[2] J. Fingas, “Self-driving vehicles and robotic clerks could take your job in 20 years,” 03 2015. [Online]. Available: http://www.engadget.com/2015/03/08/robots-may-take-more-jobs/

[3] E. Cervera, “Integrating computer vision libraries in networked robotic systems,” in *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*, June 2005, pp. 267–272.

[4] J. H. Kim, “Multi-uav-based stereo vision system without gps for ground obstacle mapping to assist path planning of ugv read more at: http://phys.org/news/2014-09-air-ground-based-robot-vehicles.html#jcp,” *Electronics Letters*, vol. 50, no. 20, pp. 1431–1432, 09 2014. [Online]. Available: http://phys.org/news/2014-09-air-ground-based-robot-vehicles.html

[5] P. E. Ross, “Driverless Cars: Optional by 2024, Mandatory by 2044 - IEEE Spectrum,” 2014. [Online]. Available: http://spectrum.ieee.org/transportation/advanced-cars/driverless-cars-optional-by-2024-mandatory-by-2044

[6] J. Schmiduber, “Professor Schmidhuber’s Highlights of Robot Car History,” 2011. [Online]. Available: http://people.idsia.ch/ juergen/robotcars.html

[7] L. Martin, “Driving Forces: Lockheed Martin‘s Autonomous Land Vehicles,” 2012. [Online]. Available: http://www.lockheedmartin.com/us/100years/stories/alv.html

[8] M. Novak, “DARPA Tried to Build Skynet in the 1980s,” 2013. [Online]. Available: http://paleofuture.gizmodo.com/darpa-tried-to-build-skynet-in-the-1980s-1451000652

[9] D. Pomerleau, “RALPH: Rapidly Adapting Lateral Position Handler,” *IEEE Symposium*, no. September 25-26, 1995, 1995. [Online]. Available: http://www.cs.cmu.edu/ tjochem/nhaa/ralph.html

[10] J. Davis, “Say Hello to Stanley,” 2006. [Online]. Available: http://archive.wired.com/wired/archive/14.01/stanley.html?pg=1&topic=stanley&topic_set=

[11] DARPA, “DARPA Grand Challenge 2005,” 2005. [Online]. Available: http://archive.darpa.mil/grandchallenge05/gcorg/

[12] C. Dobby, “Nevada state law paves the way for driverless cars,” 2011. [Online]. Available: http://business.financialpost.com/2011/06/24/nevada-state-law-paves-the-way-for-driverless-cars/?__lsa=e443-35b3

[13] U. H. of Representatives, “How Autonomous Vehicles Will Shape the Future of Surface Transportation,” p. 1, 2013. [Online]. Available: http://transport.house.gov/calendar/eventsingle.aspx?EventID=357149

[14] Mobileye. (2014, 09) Artificial vision technology. Mobileye. [Online]. Available: http://www.mobileye.com/technology/

[15] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009, p. 5.

[16] D. Vernon, “An Optical Device for Computation of Binocular Stereo Dispairity with a Single Static Camera,” in *Opto-Ireland 2002: Optical Metrology, Imaging, and Machine Vision*, vol. 38, 2003.

[17] D. L. Baggio, *Mastering OpenCV with Practical Computer Vision Projects*, 2012.

[18] R. I. Hartley and P. Sturm, “Triangulation,” *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146–157, 1997. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1077314297905476

[19] D. Lowe, “Object recognition from local scale-invariant features,” *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, 1999.

[20] “Feature Matching — OpenCV 3.0.0-dev documentation.” [Online]. Available: http://docs.opencv.org/trunk/doc/py_tutorials/py_feature2d/py_matcher/py_matcher.html

[21] M. Bertozzi, A. Broggi, M. Cellario, A. Fascioli, P. Lombardi, and M. Porta, “Artificial vision in road vehicles,” *Proceedings of the IEEE*, vol. 90, no. 7, 2002.

[22] H. Fountain, “Yes, Driverless Cars Know the Way to San Jose,” 2012. [Online]. Available: http://www.nytimes.com/2012/10/28/automobiles/yes-driverless-cars-know-the-way-to-san-jose.html?pagewanted=1&_r=0

[23] L. Gannes, “Here’s What It’s Like to Go for a Ride in Google’s Robot Car,” 2014. [Online]. Available: http://recode.net/2014/05/13/googles-self-driving-car-a-smooth-test-ride-but-a-long-road-ahead/

[24] A. Goncalves and S. Joao, “Low Cost Sensing for Autonomous Car Driving in Highways.” [Online]. Available: http://welcome.isr.ist.utl.pt/img/pdfs/1663_hans_icinco07.pdf

[25] E. Guizzo, “How Google’s Self-Driving Car Works,” 2011. [Online]. Available: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works

[26] L. Hardesty, “Think Fast, Robot,” 2014. [Online]. Available: http://newsoffice.mit.edu/2014/think-fast-robot-0530

[27] A. Heyden and M. Pollefeys, “Multiple view geometry,” in *Emerging Topics in Computer Vision*, 2005, pp. 45–107.

[28] A. Iliafar, “LIDAR, Lasers, and Logic: Anatomy of an Autonomous Vehicle,” 2013. [Online]. Available: http://www.digitaltrends.com/cars/lidar-lasers-and-beefed-up-computers-the-intricate-anatomy-of-an-autonomous-vehicle/

[29] B. Kehoe, A. Matsukawa, S. Candido, J. Kuffner, and K. Goldberg, “Cloud-based robot grasping with the Google object recognition engine,” *2013 IEEE International Conference on Robotics and Automation*, pp. 4263–4270, May 2013. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6631180

[30] T. Lassa, “The Beginning of the End of Driving,” 2013. [Online]. Available: http://www.motortrend.com/features/auto_news/2012/1301_the_beginning_of_the_end_of_driving/

[31] D.-J. Lee, J. Archibald, X. Xu, and P. Zhan, “Using distance transform to solve real-time machine vision inspection problems,” *Machine Vision and Applications*, vol. 18, no. 2, pp. 85–93, Nov. 2006. [Online]. Available: http://link.springer.com/10.1007/s00138-006-0050-2

[32] M. de Paula, “Autonomous driving,” *Popular Science*, May 2014. [Online]. Available: http://www.popsci.com/blognetwork/tags/autonomous-driving

[33] S. J. D. Prince, “Computer vision : models , learning and inference,” 2012.

[34] P. Stenquist, “On the Road to Autonomous, a Pause at Extrasensory,” 2013. [Online]. Available: http://www.nytimes.com/2013/10/27/automobiles/on-the-road-to-autonomous-a-pause-at-extrasensory.html?pagewanted=all

[35] “Automated Driving Applications and Technologies for Intelligent Vehicles - AdaptIVe FP7 project- Automated Driving Applications and Technologies for Intelligent Vehicles.” [Online]. Available: http://www.adaptive-ip.eu/

[36] “Volvo Car Group‘s first self-driving Autopilot cars test on public roads around Gothenburg - Volvo Car Group Global Media Newsroom.” [Online]. Available: https://www.media.volvocars.com/global/en-gb/media/pressreleases/145619/volvo-car-groups-first-self-driving-autopilot-cars-test-on-public-roads-around-gothenburg

[37] A. Davies and A. Gallery, “Self-driving cars will make us want fewer cars,” 03 2015. [Online]. Available: http://www.wired.com/2015/03/the-economic-impact-of-autonomous-vehicles/

[38] A. Stoklosa, “Google shows off how its autonomous vehicles aren‘t killing cyclists or hitting parked cars,” *Car and Driver*, 04 2014. [Online]. Available: http://blog.caranddriver.com/google-shows-off-how-its-autonomous-vehicles-arent-killing-cyclists-or-hitting-parked-cars/

[39] ——, “California attempts to wade into the uncharted waters of autonomous-car regulation,” *Car and Driver*, 03 2014. [Online]. Available: http://blog.caranddriver.com/california-attempts-to-wade-into-the-uncharted-waters-of-autonomous-car-regulation/

[40] J. Holloway, “Rinspeed releases details of micromax swarm car concept,” *Gizmag*, 02 2013. [Online]. Available: http://www.gizmag.com/rinspeed-micromax/26392/

[41] C. Weiss, “Rinspeed shows what the self-driving car will be like to ride in,” *Gizmag*, 12 2013. [Online]. Available: http://www.gizmag.com/rinspeed-self-driving-concept/30104/

[42] VisLab. (2015) Automotive. [Online]. Available: http://vislab.it/automotive/

[43] C. Atiyeh, "European manufacturers leading r&d for autonomous cars we may actually want to drive," *Car and Driver*, 02 2014. [Online]. Available: http://blog.caranddriver.com/european-manufacturers-leading-rd-for-autonomous-cars-we-may-actually-want-to-drive/

[44] "Characteristics of Fifth-Wheel (Wagon Steer) Steering Page," in *Engineering Design Handbook - Automotive Series - Automotive Suspensions: (AMCP 706-356)*. U.S. Army Materiel Command, Nov. 2012. [Online]. Available: http://app.knovel.com/web/view/swf/show.v/rcid:kpEDHASAS1/cid:kt00AC8JW7/viewerType:pdf/r design-handbook-18?cid=kt00AC8JW7&page=4&b-q=ackermann steering&sort_on=default&b-subscription=TRUE&b-group-by=true&b-sort-on=default&q=ackermann steering

[45] "Systems Engineering Management," in *Systems Engineering Fundamentals*. U.S. Department of Defense, Jun. 2001. [Online]. Available: http://app.knovel.com/web/view/swf/show.v/rcid:kpSEF00001/cid:kt00TYSBI1/viewerType:pdf/roc engineering-fundamentals?cid=kt00TYSBI1&page=2&b-q=system engineering&sort_on=default&b-subscription=TRUE&b-group-by=true&b-search-type=tech-reference&b-sort-on=default&b-toc-cid=kpSEF00001&b-toc-root-slug=systems-engineering-fundamentals&b-toc-url-slug=purpose&b-toc-title=Systems Engineering Fundamentals

[46] U. D. of Transportation. (2013, 08) Model systems engineering documents for adaptive signal control technology (asct) systems. [Online]. Available: http://ops.fhwa.dot.gov/publications/fhwahop11027/sec_b.htm

[47] F. Brunet. (2011, 07) First definitions and concepts. [Online]. Available: http://www.brnt.eu/phd/node16.html

[48] J. V. Does. (2014, 08) Openmvg photo reconstruction. [Online]. Available: http://blog.htmlfusion.com/openmvg/

[49] Minidoodle. Gear ratio. [Online]. Available: http://jleibovitch.tripod.com/id111.htm

[50] D. Bray. (2014, 09) Getting a raspberry pi on worcester polytechnic institute (wpi) wifi (wpa-eap). esologic.com. [Online]. Available: http://www.esologic.com/?p=1088

Appendix A

Vision Code Repository

This Appendix serves as a repository for all code used throughout this MQP

A.1 ROS RTP Relay Initialize Bash Commands

The following is the bash command that initializes and launches the ROS node that performs a relay of RTP video stream data to a ROS topic.

```
> export ROCON_RTSP_CAMERA_RELAY_URL=rtsp://YOUR IPCAM URL
> roslaunch rocon_rtsp_camera_relay rtsp_camera_relay.launch --screen
```

A.2 ROS Image Subscriber Node

The following is the code used to create a ROS node that subscribes to the RTP image topic.

```
from SimpleCV import *
import rospy
from std_msgs.msg import String

#analyzes image data from camera, tries to verify if there is a road.
```

```
def callback(data):
        Roadimg = Image(imgmsg_to_cv2(data,
                                      encoding="passthrough"), cv2image=True)
    #Roadimg Stores an image that can have vision calculations done on it.

def listener():
    rospy.init_node('node_name')
    rospy.Subscriber("RTSP", imgmsg, callback)
    # spin() simply keeps python from exiting until this node is stopped
    rospy.spin()
```

A.3 Image Road Detection Color Test

The following is the code that was used to conduct the road detection test for an image taken and routed through this framework

```
from SimpleCV import *
import rospy

def Road_test(testImage): #This is the function that tests Roadimg.
        greenExtract = testImage.colorDistance(Color.GREEN)
    Lawn = testImage - greenExtract
    rospy.loginfo(Lawn.meanColor())
    #this will log through loss the mean RGB color values of a green extracted
    #image.
    #The green color value can be useful if trying to detect a lawn vs.
    #a road.
```