5-2015

Lateral Control of a Vehicle Platoon

Samuel A. Mitchell
Utah State University

LATERAL CONTROL OF A VEHICLE PLATOON

by

Samuel A. Mitchell

Thesis submitted in partial fulfillment of the requirements for the degree

of

HONORS IN UNIVERSITY STUDIES WITH DEPARTMENTAL HONORS

in

Electrical Engineering in the Department of Electrical and Computer Engineering

Thesis/Project Advisor
Ryan Gerdes

Thesis/Project Advisor
Donald Cripps

Departmental Honors Advisor
Dean Adams

Director of Honors Program
Kristine Miller

UTAH STATE UNIVERSITY
Logan, UT

Spring 2015

Abstract

Multiple lateral control systems are analyzed for use in a vehicle platoon system. In order to ensure the safety of the vehicle platoon, the system must operate under three constraints: (1) accurate path following, (2) string stability, and (3) functionality in the presence of noise. Computer simulations are employed to analyze candidates according to the safety requirements. Actual vehicle kinematics and nonlinearities – limits on turn radius, velocity, and acceleration – are included in this analysis.

Successful control system candidates are implemented in a platoon of five differential-steer vehicles. The sensing and communication requirements of the control system are discussed. Experimental results are compared to the computer simulations. This analysis results in an implementation of a control system which functions according to the previously listed constraints.

Acknowledgements

Many thanks to my advisors, Dr. Ryan Gerdes, Dr. Rajnikant Sharma, Dr. Don Cripps, and Jolynne Berrett.

The members of the SATS Group have been amazing, offering insight and many hours of vehicle assembly. Daniel Dunn, Ali Al-Hashimi, Soudeh Dadras, and Imran Sajjad.

This project was funded in part by an URCO grant by USU Office of Research and Graduate Studies and NSF Award #1410000.

Document Description

This thesis is composed as a conglomeration of documents. Each document covers one chapter of the thesis. Tables of Contents, Lists of Tables, Lists of Figures, and bibliographies are contained within each chapter (when necessary). Page numeration is also relative to each chapter.

Contents

Chapter 1

Proposal

Lateral Control of a Vehicle Platoon

Sam Mitchell
Electrical and Computer Engineering Department
Utah State University

December 11, 2014

Contents

List of Figures

List of Tables

1 Project Summary

A proposal introducing a senior design project that will analyze and implement vehicle-following control algorithms in platooning systems. Motivations, objectives, methods, costs, and a timeline are discussed. The first phase of this project analyzes and simulates various algorithms. The final phase is an implementation of these algorithms on a physical system. The result of the project is the implementation and demonstration of a functioning vehicle-following platoon on a fleet of ten robots.

The assistance requested is permission to proceed. The materials will be purchased by Dr. Ryan Gerdes. Wages are supplied by Dr. Ryan Gerdes and the Office of Research and Graduate Studies via an Undergraduate Research and Creative Opportunities Grant. The budget is found in Table 2.

2 Problem

In recent years, there has been a great deal of discussion in the media about self-driving vehicles. For instance, the Google Car, an autonomous vehicle, recently received a license to drive on public roads.

With these advances in technology, transporting cargo has the potential to become more efficient through a vehicle following system known as platooning through vehicle following. An example of this is shown in Figure 1.

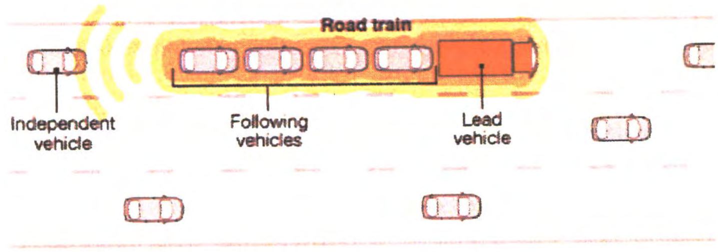


Figure 1: A vehicle platoon demonstrating following vehicles trailing behind a lead vehicle.

Application of vehicle following control technology would benefit military transportation in adverse environments. Transporting supplies in a caravan puts a minimum of one person in danger for each vehicle in the caravan. A remote controlled platooning system will allow a driver to remotely control the lead vehicle, and the other vehicles will follow autonomously. This distance from the vehicles will potentially save money and lives.

Another potential application of vehicle following is shipping companies. One benefit of driving vehicles in a close formation is reduced air drag. By driving close together, the vehicles would have reduced fuel consumption, effectively lowering fuel costs.

3 Objectives

The goal of this senior design project is to implement and demonstrate a vehicle following system that will allow a platoon of vehicles to follow a lead vehicle.

The purpose of this project is the implementation and demonstration of a vehicle-following control algorithm. Previous applications of vehicle following demonstrate the following using either a limited number of vehicles or at limited speeds, while the demonstration of this system will take place on a fleet of ten robots at speeds up to 15 meters per second (mps).

4 Solution

There are a variety of methods to control a vehicle following system. Rajamani provides an effective overview of the control problem [6]. CC Chien analyses what parameters are required to effectively track another vehicle [2]. Novel communication methods [1] and modeling methods [3][4][5][7] are presented.

The control of a vehicle platoon can be split into two controllers — lateral and longitudinal. The lateral controller steers the vehicle. The longitudinal controller will maintain a set distance between the current vehicle and the preceding vehicle.

A longitudinal controller for the vehicle platoon has already been developed, so this project will implement a lateral controller that will work in conjunction with the longitudinal controller.

5 Methods

The end goal of the senior design project is the implementation and demonstration of a vehicle-following controller for a platoon of vehicles. The demonstration will take place on a fleet of ten robots. Prior to delving into the details of the project, a broad overview of the platooning system and motivations are given.

The platoon is grouped into two types of vehicles: follower(s) and one leader. The leader will be controlled via radio transmission from the user with the option to follow a predetermined route. The leader will collect and transmit information about its state to the other vehicles in the platoon.

Each follower will collect data on its own state and receive data from the lead vehicle. Using these data points, the follower will use a lateral control algorithm in order to determine what trajectory it will follow. The follower will use this data to control their motion, following in the same track as the leader. In addition to the lateral control, the follower will use sonar to determine the distance between itself and the preceding vehicle so it will not cause a collision with other vehicle(s).

5.1 Algorithm research and selection

5.1.1 Preliminary selection

Before simulation, a high-level analysis of available vehicle-following algorithms will be performed. Three algorithms will be selected for simulation and testing. This phase of the project is currently underway with potential candidates found here [1][2][3][6][7].

5.1.2 Simulation

After a preliminary evaluation of potential control algorithms, a more thorough evaluation will take place. A vehicle platoon will be modeled in Matlab and each of the selected algorithms will be simulated under various non-ideal conditions. A few of the simulation conditions include sensor error and time delay types of events that could potentially cause instability in the system.

Matlab is the premier tool used by professional engineers to model control systems. Effective utilization of this tool will allow for rapid development and analysis of control algorithms.

These simulations will be a safe method for checking the viability of the selected control algorithms, as a poorly implemented controller could result in a spectacular collision.

5.2 Design

Specifications will be developed to ensure the system will be able to run the control algorithms. These specifications will be used to evaluate the vehicles currently in the lab. If necessary, system requirements will be created that will give the vehicles capability to fulfill the specifications.

5.3 Development

5.3.1 Obtain components

Once the system requirements are completed, I will purchase and install the parts outlined. The selection of these components will not be completed until after the algorithm verification is complete. While exact parts aren't known yet, an anticipated budget is found in Table 2. All required equipment will be supplied by Dr. Ryan Gerdes.

5.3.2 Code development

The controller will be implemented in the C programming language on the Tiva C development board, which uses a TM4C123GH6PM processor. The code will be written to interact with the heritage code from Daniel Dunn's longitudinal vehicle control project.

5.3.3 Testing and verification

An important part of product development is the testing and verification that the product meets requirements. Effective testing criteria will fully capture the strengths and limitations of a system.

Testing will be performed as code development reaches completion. The specification document lists testing procedures that will verify that the objectives of the project were met.

5.4 Produce documentation

Final documentation will be compiled as the testing is completed. This documentation will include a final report.

5.4.1 Final review and presentation

Upon completion of the project, the project will be reviewed by Dr. Don Cripps and Ms. Jolynne Berrett. Results from the project will be presented at Senior Design Night.

6 Resources

The vehicle system used is a Battlekit robot system (see Figure 2). These vehicles can sustain speeds of up to 15mps and are constructed to remain usable after collisions. More information on peripherals utilized in the system is found in Section 5.3.1.

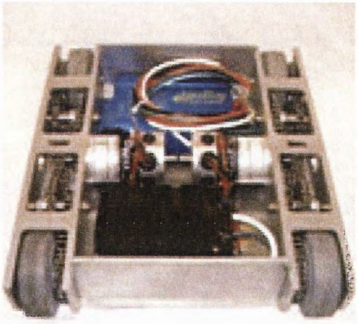


Figure 2: One of the vehicles to be used in demonstration of the control algorithms.

7 Schedule

The timeline for this project is broken into four phases, using Section 5.2 - 5.4 as phases of development. I will report to Dr. Ryan Gerdes about the completion of each milestone on the schedule.

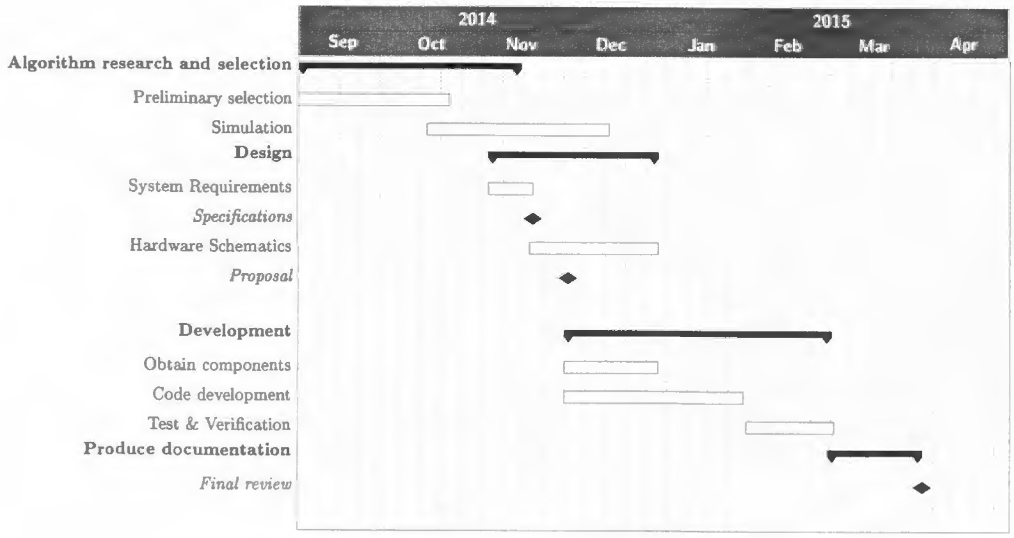


Table 1: Proposed schedule for the senior design project.

8 Qualifications

I am a Senior in the Electrical and Computer Engineering Department, majoring in Electrical Engineering. For more information, see the attached resume.

9 Costs

The budget is shown below.

Table 2: Anticipated Budget

Bill of Materials			
Item	Cost	Quantity	Total Cost
Vehicle	$1,000	10	$10,000
Xbee wireless transceiver	$25	10	$250
Ultrasonic range finder	$25	10	$250
Tiva C Development Board	$13	10	$130
Wage Sources			
SATS Program	$10	75 hours	$750
URCO Grant	$10	125 hours	$1,250
Total			**$12,630**

10 Conclusion

This project is pertinent to fields of study including embedded systems and mechanical control. The control of vehicles is becoming progressively more automated, and an accurate vehicle following system is one method to optimize travel and transport costs in multiple applications. This project will analyze and implement various algorithms, resulting in a discussion of the algorithms.

An investigation to potential algorithms is under way, and results will be produced in the following weeks.

The rest of the project still needs to be completed. These steps are: algorithm research and selection, design, development, and document production.

11 Bibliography

[1] Mohammad Y Abualhoul, Mohamed Marouf, Oyunchimeg Shagdar, and Fawzi Nashashibi. Platooning control using visible light communications: A feasibility study. In *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on*, pages 1535–1540. IEEE, 2013.

[2] CC Chien and P Ioannou. Automatic vehicle-following. In *American Control Conference, 1992*, pages 1748–1752. IEEE, 1992.

[3] Takehiko Fujioka and Manabu Omae. Vehicle following control in lateral direction for platooning. *Vehicle System Dynamics*, 29(S1):422–437, 1998.

[4] Teck Chew Ng, Javier Ibanez Guzman, and Martin David Adams. Autonomous vehicle-following systems: A virtual trailer link model. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3057–3062. IEEE, 2005.

[5] P Petrov. Nonlinear adaptive control of a two-vehicle convoy. *Open Cybernetics & Systemics Journal*, 3:70–78, 2009.

[6] Rajesh Rajamani. *Vehicle dynamics and control.* Springer, 2011.

[7] Soo-Yeong Yi and Kil-To Chong. Impedance control for a vehicle platoon system. *Mechatronics*, 15(5):627–638, 2005.

Chapter 2

Specification

REQUIREMENTS SPECIFICATION

for

Vehicle-Following Platoon

Release 2.0
16 January 2015

Samuel Mitchell, s.mitchell.us@ieee.org

Contents

Signature Page

Reviewed by:

Name, Title, email	Date
Name, Title, email	Date
Name, Title, email	Date
Name, Title, email	Date
Name, Title, email	Date

Revision History

Revision	Description	Author	Date	Approval
1	Draft	Samuel Mitchell	11/06/2014	Failed
2	Draft	Samuel Mitchell	1/16/2015	Pending
3				
4				
5				
6				
7				
8				
9				
10				

1 Scope

The scope of this document is to establish the design, performance, development, and test requirements of the vehicle-following platoon system (VFPS).

This document does not detail the following: implementation methods, potential controller algorithms, or motivations for the project.

2 Applicable Documents

The following documents shall form part of this specification, as a more complete description of the purpose and motivations of the VFPS.

- Technical Memo: Lateral Control of a Vehicle Platoon. S. Mitchell to D. Cripps. 16 Oct. 2014.
- FCC Part 15.247. Operation within the bands 902-928 MHz, 2400-2483.5 MHz, and 5725-5850 MHz.
- IEEE Standard 802.15.4.

2.1 Product Perspective

A brief description of the product is given in Technical Memo (S. Mitchell, 2014).

> The completed VFPS will result in the implementation and demonstration of a vehicle-following controller for a platoon of vehicles. The demonstration will take place on a fleet of ten robots. Prior to delving into the details of the project, a broad overview of the platooning system and motivations are given.
>
> The platoon is grouped into two types of vehicles: follower(s) and one leader. The leader will be controlled via radio transmission from the user with the option to follow a predetermined route. The leader will collect and transmit information about its state to the other vehicles in the platoon.
>
> Each follower will collect data on its own state and receive data from the lead vehicle. Using these data points, the follower will use a lateral control algorithm in order to determine what trajectory it will follow. The follower will use this data to control their motion, following in the same track as the leader. In addition to the lateral control, the follower will use sensors to determine the distance between itself and the preceding vehicle so it will not cause a collision with other vehicle(s).

3 Requirements

3.1 Control Algorithm Performance

3.1.1 Accuracy

3.1.1.1

The controller shall cause the follower vehicle to track the lead vehicle at turn angles up to 40° — the vehicle travels straight, changes heading by 40° and continues to travel straight.

3.1.1.2

The controller shall cause the follower vehicle to track the lead vehicle's exact path within 30 cm.

3.1.2 Following distance

The controller shall allow the follower vehicle to follow the preceding vehicle at short distances. When the lead vehicle is traveling at 15 m/s, the following vehicle shall be capable of following at a following distance of 1 m.

3.1.3 Reaction time

The following vehicle shall react to the lead vehicle at full speed (15 m/s) while following at a distance of 1 m. Reaction times to variations in the lead vehicle's path shall be a maximum of 67 ms.

3.2 Physical Characteristics

3.2.1 Lead Vehicle

3.2.1.1 Mechanical

3.2.1.1.1
The vehicle shall drive at speeds up to 15 m/s.

3.2.1.1.2
The vehicle shall accelerate at rates up to 5 m/s^2.

3.2.1.2 Electrical

The vehicle shall be able to operate at maximum power usage for a minimum of 10 minutes.

3.2.1.3 Communication

3.2.1.3.1
The vehicle shall be able to communicate with the operator at distances of at least 100 m.

3.2.1.3.2

The vehicle shall be able to communicate with at least 2 other vehicles.

3.2.1.4 Computation

The vehicle shall be able to store and follow a predefined path.

3.2.2 Follower Vehicle(s)

3.2.2.1 Mechanical

3.2.2.1.1

The vehicle shall drive at speeds up to 15 m/s.

3.2.2.1.2

The vehicle shall accelerate at rates up to 5 m/s^2.

3.2.2.2 Electrical

The vehicle shall be able to operate at maximum power usage for a minimum 10 minutes.

3.2.2.3 Communication

3.2.2.3.1

The vehicle shall be able to communicate with the operator at distances of at least 100 m.

3.2.2.3.2

The vehicle shall be able to communicate with at least 2 other vehicles.

3.2.2.4 Computation

3.2.2.4.1 Data storage The vehicle shall be able to store 100 32-bit samples of x, $\dot{x}$, y, $\dot{y}$ position data.

3.2.2.4.2 Algorithm The vehicle shall be able to execute the controller defined in Section 3.1.

4 Requirements Verification

4.1 Control Algorithm Performance

The controller shall be tested by simulation. The simulation must incorporate communication and measurement errors so as to accurately represent reality. Further verification shall be performed in Sections , , and .

4.1.1 Accuracy

Requirements in Sections and . This requirement shall be verified by simulation. The lead vehicle's path shall follow a track with two (2) 20° turns and one (1) 40° turn. The follower vehicle shall follow the lead vehicle, and the follower and lead vehicles' paths shall be compared and found to be within 30 cm of each other.

4.1.2 Following distance

Requirements in Section . Perform the test in at 15 m/s and a following distance of 1 m. When the lead vehicle is traveling at 15 m/s, the following vehicle shall be capable of following at a distance of 1 m.

4.1.3 Reaction time

Requirements in Section . Verified by .

4.2 Physical Characteristics

4.2.1 Lead Vehicle

4.2.1.1 Mechanical

4.2.1.1.1
Requirements in Section . Verified by test. The test is passed if the vehicle propels itself to go 15 m/s.

4.2.1.1.2
Requirements in Section . Verified by test. The test is passed if the vehicle propels itself from a standstill to 5 m/s in 1 second.

4.2.1.2 Electrical

Requirements in Section . Verified by analysis. The power consumption is measured during acceleration. This data is compared to the battery capacity to verify the operation time.

4.2.1.3 Communication

4.2.1.3.1
Requirements in Section . Verified by test. The test is passed if the vehicle can communicate with the controller radio at a distance of 100 m.

4.2.1.3.2

Requirements in Section . Verified by test. The test is passed if the vehicle can connect to 2 vehicles at the same time.

4.2.1.4 Computation

Requirements in Section . Verified by test. The vehicle must be able to follow a path 100 m long.

4.2.2 Follower Vehicle(s)

4.2.2.1 Mechanical

4.2.2.1.1

Requirements in Section . Verified by test. The test is passed if the vehicle propels itself to go 15 m/s.

4.2.2.1.2

Requirements in Section . Verified by test. The test is passed if the vehicle propels itself from a standstill to 5 m/s in 1 second.

4.2.2.2 Electrical

Requirements in Section . The vehicle shall be able to operate at maximum power usage for at least 10 minutes.

4.2.2.3 Communication

4.2.2.3.1

Requirements in Section . Verified by test. The test is passed if the vehicle can communicate with the user radio at a distance of 100 m.

4.2.2.3.2

Requirements in Section . Verified by test. The test is passed if the vehicle can connect to 2 vehicles at the same time.

4.2.2.4 Computation

4.2.2.4.1 Data storage Requirements in Section . Verified by design. The design must contain data space for 100 32-bit samples of x, $\dot{x}$, y, $\dot{y}$ position data.

4.2.2.4.2 Algorithm Requirements in Section . Verified by test. The vehicle must execute the controller in real-time.

Chapter 3

Final Report

Lateral Control of a Vehicle Platoon

Sam Mitchell, Kyle Hymas
Electrical, Mechanical Engineering
Utah State University

April 30, 2015

Contents

List of Figures

List of Tables

Executive Summary

Multiple lateral control systems are analyzed for use in a vehicle platoon system. In order to ensure the safety of the vehicle platoon, the system must operate under three constraints: (1) accurate path following, (2) string stability, and (3) functionality in the presence of noise. Computer simulations are employed to analyze candidates according to the safety requirements. Actual vehicle kinematics and nonlinearities – limits on turn radius, velocity, and acceleration – are included in this analysis.

Successful control system candidates are implemented in a platoon of five differential-steer vehicles. The sensing and communication requirements of the control system are discussed. Experimental results are compared to the computer simulations. This analysis results in an implementation of a control system which functions according to the previously listed constraints.

Scope

The purpose of this document is to provide a brief overview of the project, "Lateral Control of a Vehicle Platoon." This project was performed in adherence with the design portion of the Utah State University Electrical Engineering undergraduate degree requirements.

This document is not a comprehensive description of the system design. A more thorough description of the system is found in the SATS Vehicle Platoon Handbook and the Vehicle Platoon API.

The document does outline the selection and design of the vehicle platoon lateral control system. It also includes an explanation of the software framework.

Applicable documents

The following documents are pertinent to the design and implementation of this vehicle platoon:

- Technical Memo: Lateral Control of a Vehicle Platoon. S. Mitchell to D. Cripps. 16 Oct. 2014.
- Requirements Specification for Vehicle-Following Platoon. S. Mitchell. 16 Jan. 2015.
- SATS Vehicle Platoon Handbook. Secure Automated Transportation Systems Group. 30 Apr. 2015.
- Vehicle Platoon API. Secure Automated Transportation Systems Group. 30 Apr. 2015.

1 Introduction

Nationally, freeway congestion costs the economy over $1 trillion annually. Automated vehicles are part of a solution to preventing future costly and time-wasting traffic jams. Currently there is a great deal of research on autonomous vehicle control [1]. Most of the research is focused on longitudinal control (linear velocity) of vehicles in a freeway environment, while the lateral control (steering) of the vehicles is neglected. When lanes aren't well marked due to construction, redirection of traffic, or upkeep negligence, the standard lanekeeping control algorithms aren't effective. Vehicle following is a solution to this problem.

Various authors [2, 3, 4, 5, 6] have proposed lateral controllers which control the lateral and longitudinal behavior simultaneously. While combining the lateral and longitudinal controllers is an effective method of reducing computation time, a flaw in either the lateral or longitudinal controller renders the entire system useless.

One application of improved lateral control is military transportation in adverse environments. In hostile environments, supply caravans often form a formation and traverse an area. This puts a minimum of one person in danger for each vehicle in the caravan. A remote controlled platooning system will allow a driver to remotely control the lead vehicle, and the other vehicles will follow autonomously. This distance from the vehicles will potentially save money and lives. Another benefit of driving vehicles in a close formation is reduced air drag. By driving close together, the vehicles would have reduced fuel consumption.

The purpose of this project is to implement and demonstrate a vehicle-following control algorithm to be used by the Secure Automated Transportation Systems (SATS) Group. Previous applications of vehicle following demonstrate the following using either a limited number of vehicles or at limited speeds, while the demonstration of this system will take place on a fleet of five robots at speeds up to 8 meters per second (mps).

1.1 Structure of paper

The organization of this paper is as follows: In Section 2, controller selection and simulation methods are discussed. Section 3 contains the simulation results. Section 4 describes the vehicle assembly and software implementation, which is followed by results and analysis in Section 6. Conclusions and an outline of future work work are in Section 7.

2 Controller selection and simulation

The focus of this project is to select and implement a suitable lateral controller for use in a vehicle platooning system. The controller must meet the following criteria:

- The controller must not compromise the longitudinal stability of the system. See Yanakiev's [7] on string stability.
- The vehicle must follow the path of the lead vehicle accurately.

The platoon is grouped into two types of vehicles: follower(s) and one leader. The leader will be controlled via radio transmission from the user with the option to follow a predetermined route. The leader will collect and transmit information about its state to the other vehicles in the platoon.

Each follower will collect data on its own state and receive data from the lead vehicle. Using these data points, the follower will use a lateral control algorithm in order to determine what trajectory it will follow. The follower will use this data to control their motion, following in the same track as the leader. In addition to the lateral control, the follower will use sonar to determine the distance between itself and the preceding vehicle so it will not cause a collision with other vehicle(s).

After a preliminary evaluation of potential control algorithms. A vehicle platoon was modeled in Matlab and each of the selected algorithms was simulated under various non-ideal conditions.

Electronic simulation is a safe method for checking the viability of the selected control algorithms, as a poorly implemented controller could result in a spectacular collision.

Matlab is the premier tool used by professional engineers to model control systems. Effective utilization of this tool will allow for rapid development and analysis of control algorithms.

2.1 Controller selection

A range of papers on lateral control of a vehicle platoon were analyzed on how the controller fulfilled the above requirements. Three controllers were selected and simulated. These are:

- Nonlinear adaptive control, P. Petrov.
- Impedance control of a vehicle platoon, S.-Y. Yi.
- Separable control of a vehicle platoon, S. Mitchell.

2.2 Controller design

Lateral controllers found in the literature combined the lateral and longitudinal control into one nonlinear controller. This vehicle platooning project is being developed for rapid testing of longitudinal controllers, which is simplified by separating the lateral and longitudinal controllers.

A separable control law is proposed that accepts lateral and longitudinal control laws as inputs. The controller design drew heavily from Cripps [8] and Ferrin [9] for design ideas. The proposed vehicle model and controller controller is developed in full in Appendix A, B.

3 Results

3.1 Simulation results

The three controllers in Section 2.1 were evaluated according to the design criteria found in Section 2. The simulation was that the leader of each platoon would travel the path in Figure 1 at velocities shown in Table 1. The results of the system are recorded in Table 2.

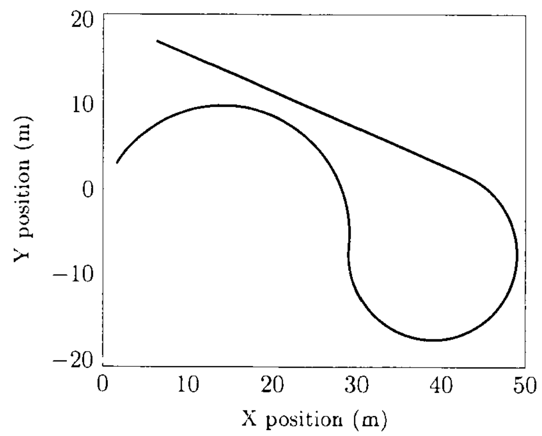


Figure 1: The path taken by the leader in simulation.

Table 1: Lead vehicle simulation maneuvers.

	Turn radius (m)	Duration (s)	Velocity (m/s)
Left turn	15	10	4
Right turn	10	22	2
Straight	∞	8	5

Table 2: Simulation results.

	Max path error (cm)	String stable
Impedance [5]	852	✓
Nonlinear adaptive [6]	12.4	
Separable	4.5	✓

3.1.1 Impedance control

The impedance control produced effective longitudinal control — the system was string stable. The path error was unacceptable.

3.1.2 Nonlinear adaptive control

The nonlinear adaptive control reduced the path error to minimal amounts. The longitudinal control amplified the error between vehicles, which is string instable.

3.1.3 Separable control

The separable controller utilized a constant time headway longitudinal controller. The path following algorithm worked well when the gains were tuned properly.

4 System design

4.1 Hardware design

The system was designed by Sam Mitchell and Daniel Dunn. Design reasoning and justification are found in "SATS Vehicle Platoon Handbook." This document only contains a brief overview of selected parts. Assembly and operation instructions are also found in the "SATS Vehicle Platoon Handbook."

4.1.1 Components

- XBee s6b Wi-Fi module
- LIDAR-Lite
- 2000 pulse quadrature encoders
- ADXL345 accelerometer

4.2 Software design

The control system has multiple sensors with various sample rates. A super-loop system with a fixed period should be sufficient, but the entire system would break if any sensor produces unexpected delay. In order to avoid system instability, the software is interrupt driven.

The control loop retains the highest priority to ensure continued execution. The sensors and communication have lower priority.

If the communication module receives a kill command (the character 'c' followed by a newline '\n'), the system performs the following:

1. Relays the kill command to its follower vehicle.
2. Disables the controller by disabling interrupts.
3. Sets the vehicle velocity to 0.
4. Repeats the kill command indefinitely to ensure the follower vehicle received the command.

4.2.1 Scheduling

The TM4C123GH6PM microcontroller has multiple on-board hardware timers available. Separate timers drive the control loop, sensors, and communication transmission. The task timing and priority is shown in Table 3.

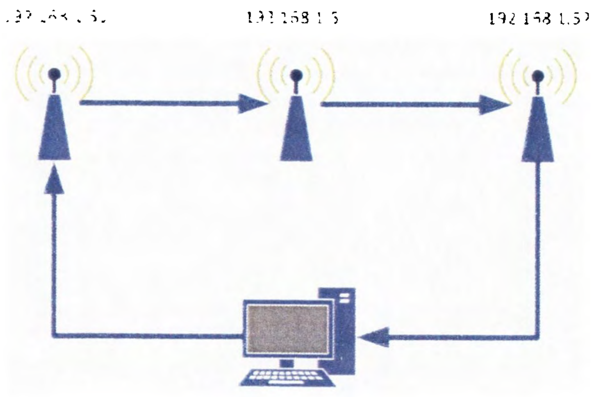


Figure 2: Vehicle platoon communication scheme.

Table 3: Task timing and priority schedule.

Application	Interrupt from peripheral	Frequency (Hz)	Interrupt level
Controller	Timer0	50	0
UART Transmit	Timer1	10	4
UART Receive	UART1	Event driven	5
Lidar, Accelerometer	Timer2	10	5

4.2.2 Communication

Communication is supported using the XBee modules and an external access point. Each XBee module requests a static IP address. The user's computer also connects, and each device transmits to the next one in line, as shown in Figure 2. An example program to interact with the vehicle platoon is found in Appendix ??.

The vehicles communicate with each other using a communication packet, which is shown in Table 4. The header and footer bytes are ASCII characters, but the data is transmitted in IEEE 754 half-precision floating point format.

Table 4: The inter-vehicle communication packet definition.

Byte	Data	Data type
1	Packet flag 'z'	ASCII
2 3	Running time (s)	Half precision float
4 5	Velocity	Half precision float
6 7	X position (m)	Half precision float
8 9	Y position (m)	Half precision float
10 11	Heading (rad)	Half precision float
12	End transmission flag '\n'	ASCII

5 Experimental results

The vehicle platooning system was implemented as described in Section 4. The design originally included the implementation of vehicle tracking using local sensors, but these peripherals were not implemented. Due to the lack of sensors, the system is driven entirely by dead reckoning and wireless communication. This results in a system that behaves as expected roughly %50 of the time.

Despite the failed reliability of the system, the platoon adheres to many of the requirements detailed in the Requirements Specification document; see Table 5.

6 Discussion

The vehicle platoon does not behave as desired. This is attributed to the position information coming entirely from communication.

While these results are not what was originally intended, the errors are quite informative. The SATS Group is currently performing research on false data injection in cooperative adaptive cruise control environments. It is clear from this platoon's communication troubles that relying solely on communication could produce catastrophic results.

Current and future research is focused on obtaining the necessary control information through various means. Some of these sensing systems are:

- Stereoscopic camera
  - Inter-vehicle distance
  - Angle to preceding vehicle
- Infrared camera
  - Angle to preceding vehicle
- Microphone
  - Angle to preceding vehicle

Table 5: Requirement verification matrix.

Requirement number	Title	Description	Pass / Fail	Comments
1	**Control Algorithm Performance**	Simulation incorporates errors.	Pass	
1.1	Accuracy	Follow path. Two 20° turns and one 40° turn. Less than 30cm path error	Pass	
1.2	Following distance	Follow at 1 m spacing at 15 m/s.	Fail	The vehicle followed at 1 m spacing at 3 m/s.
1.3	Reaction time	Controller delay within 67ms	Fail	The vehicle has a 0.5 s time constant.
2	**Physical Characteristics**			
2.1	**Lead Vehicle**			
2.1.1	Mechanical			
2.1.1.1		Maximum velocity of 15 m/s.	Fail	The maximum velocity is 9 m/s.
2.1.1.2		Acceleration of 5 m/s^2	Pass	
2.1.2	Electrical	Minimum of 10 minutes of operation.	Pass	Verified by test instead of analysis.
2.1.3	Communication			
2.1.3.1		Communication range up to 100 m.	Pass	Verified by test at Test Track facility 4/23/15.
2.1.3.2		Communicate with 2 vehicles.	Pass	The vehicle can listen to preceding vehicle and send to follower.
2.1.4	Computation	Path storage.	Fail	The final design did not utilize this system.
2.2	**Follower Vehicle(s)**			
2.2.1	Mechanical			
2.2.1.1		Maximum velocity of 15 m/s.	Fail	The maximum velocity is 9 m/s.
2.2.1.2		Acceleration of 5 m/s2	Pass	
2.2.2	Electrical	Minimum of 10 minutes of operation.	Pass	Verified by test instead of analysis.
2.2.3	Communication			
2.2.3.1		Communication range up to 100 m.	Pass	Verified by test at Test Track facility 4/23/15.
2.2.3.2		Communicate with 2 vehicles.	Pass	The vehicle can listen to preceding vehicle and send to follower.
2.2.4	Computation			
2.2.4.1	Data storage	Path storage.	Fail	The final design did not utilize this system.
2.2.4.2	Algorithm	Execute controller in real-time.	Pass	The controller is executed every 20 ms (50 Hz).

7 Conclusion

A control system for the lateral control of a vehicle platoon was proposed, selected, and implemented. Simulations of the selected controller resulted in a path error of 4.5 cm or less.

The platoon system software design is outlined. Various sensing methods were discussed, and the system currently relies on communication to obtain information about the preceding vehicle.

This platooning research resulted in an effective separable controller that can be used in further simulations and development on the SATS Group vehicle platoon.

Appendices

A Mathematical model of the vehicle

A.1 Velocity

Given a desired velocity, obtain the input voltage of the system.

- V_s Source voltage
- i_m Source current
- L_m Motor inductance
- R_m Motor resistance
- ω_m Angular velocity of the motor
- J Moment of inertia of the rotor
- b_m Damping constant for the motor
- τ_m Motor torque
- τ_L Load torque
- K_t proportionality constant
- K_m proportionality constant
- v_{wheel} Center velocity of wheel
- r_{wheel} Wheel radius

$$V_s = L_m \frac{di_m}{dt} + R_m i_m + K_m \omega_m \tag{1}$$

$$J \frac{d\omega}{dt} = \tau_m - \tau_L - b_m \omega_m \tag{2}$$

Now we convert gear ratios.

$$\begin{aligned} \omega_m &= 3.4\omega_{wheel} \\ \omega_m &= 3.4 \frac{v_{wheel}}{r_{wheel}} \end{aligned} \tag{3}$$

The velocity of the wheel is equal to the velocity of the corresponding side of the vehicle.

$$v_{wheel} = v_{right} \text{ or } v_{left} \tag{4}$$

Using these equations, we can control the velocity of either wheel.

A.2 Vehicle steering

Given a desired velocity and curvature / change of heading, obtain velocity commands for each wheel. Two mathematical models of the vehicle are presented.

Newtonian kinematics model Using the force model of the vehicle shown in Figure 3, the derivation follows.

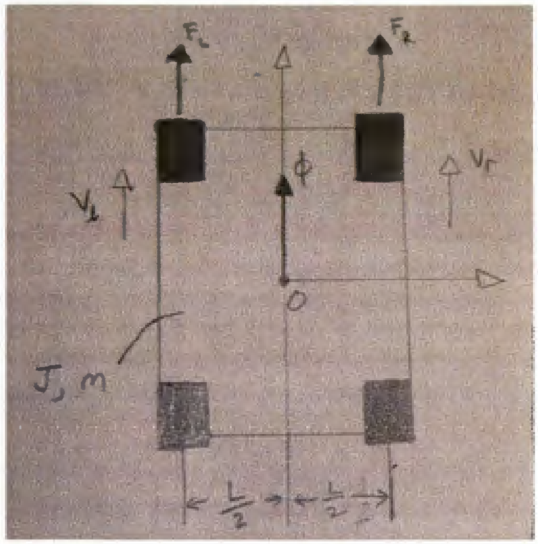


Figure 3: Vehicle force model.

If the velocities are equal, there is no change in heading.

$$v_L = v_R, \rightarrow \frac{d\phi}{dt} = 0 \tag{5}$$

If the forces are equal, there is no change in heading.

$$F_L = F_R, \rightarrow \frac{d\phi}{dt} = 0 \tag{6}$$

Now we sum the moments of inertia.

$$\begin{aligned}
\sum M_o &= J_c\ddot{\phi} \\
\frac{L}{2}F_R - \frac{L}{2}F_L &= J_c\ddot{\phi} \\
\int \left[\frac{L}{2}F_R - \frac{L}{2}F_L\right] dt &= \int J_c\ddot{\phi}dt \\
\frac{tL}{2J_c}(F_R - F_L) &= \dot{\phi}
\end{aligned} \tag{7}$$

After the change in heading has been obtained, we connect force with velocity to obtain change in heading with respect to velocities of the left and right sides. The analysis of the right wheel follows, as shown in Figure 4.

$$\begin{aligned}
F &= ma \\
\int Fdt &= \int madt \\
F_R t &= \frac{1}{2}mv_r \\
F_R &= \frac{mv_r}{2t} \\
F_L &= \frac{mv_l}{2t}
\end{aligned} \tag{8}$$

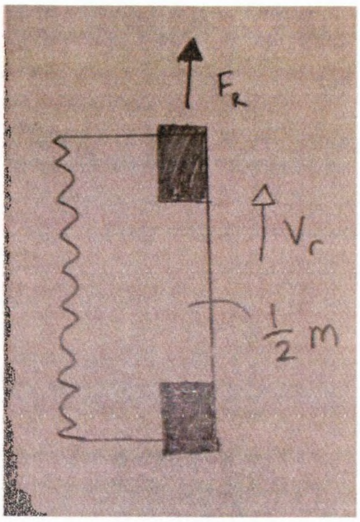


Figure 4: Force model of the right side of the vehicle.

Use the equation from equation 7.

$$\frac{tL}{2J_c}\left(\frac{mv_r}{2t} - \frac{mv_l}{2t}\right) = \dot{\phi}$$
$$\frac{mL}{4J_c}(v_r - v_l) = \dot{\phi} \tag{9}$$

A.3 Curvature

Vehicles can be represented as traveling in a circle with a variable radius. A radius of 1 meter would cause the vehicle to turn in a small circle, while a radius of ∞ would leave the vehicle traveling in a straight line. Vehicles are more likely to travel in circles of larger radii, so the curvature $\kappa = \frac{1}{r}$ will be used as a steering control.

- ϕ heading
- v velocity
- r path radius
- κ path curvature

Consider a vehicle traveling around a circle with constant radius r at constant velocity v. In order to find the heading of the vehicle, the velocity is used. A circle has 2π radians, or measures of the radius. The heading angle is equivalent to the circle angle to the point, offset by 90°.

$$C = 2\pi r$$
$$vt = \phi r$$
$$\phi = \frac{vt}{r} \tag{10}$$

$$\dot{\phi} = \frac{v}{r} = v\kappa \tag{11}$$

$$\dot{x} = r\cos\phi$$
$$\dot{y} = r\sin\phi \tag{12}$$

B Development of the separable controller

B.1 Low-level skid steer controller

The velocity of a skid-steer vehicle can be described as a synthesis of three velocities and radii of a circle.

- φ vehicle heading
- v_L, v_R, v_C vehicle velocity
- r_L, r_R, r_C path radius
- κ_L, κ_R, κ_C path curvature
- L vehicle width

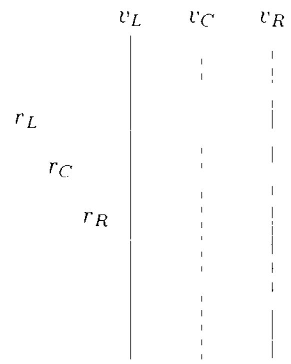


Figure 5: Skid steer velocity and turn radius.

$$\dot{\varphi} = \frac{v_L}{r_L} = \frac{v_C}{r_C} = \frac{v_R}{r_R} \tag{13}$$

$$\begin{aligned} r_L &= r_C - L/2 \\ r_R &= r_C + L/2 \end{aligned} \tag{14}$$

$$\begin{aligned} r_C &= r_L + L/2 \\ r_C &= \frac{v_L}{\dot{\varphi}} + L/2 \\ r_C &= \frac{v_L r_R}{v_R} + L/2 \\ r_C &= \frac{v_L}{v_R}(r_C + L/2) + L/2 \\ r_C\left(1 - \frac{v_L}{v_R}\right) &= \left(\frac{v_L}{v_R} + 1\right) L/2 \\ r_C\,(v_R - v_L) &= (v_R + v_L)\,L/2 \\ r_C &= \frac{v_R + v_L}{v_R - v_L} L/2 \\ \kappa_C &= \frac{v_R - v_L}{v_R + v_L} 2/L \end{aligned} \tag{15}$$

Solve for v_L using κ_C and v_C as inputs.

$$\begin{aligned}
\dot{\phi} &= \frac{v_L}{r_L} = v_C\kappa_C \\
\frac{v_L}{r_L} &= v_C\kappa_C \\
\frac{v_L}{r_C - L/2} &= v_C\kappa_C \\
v_L &= v_C\kappa_C(r_C - L/2) \\
v_L &= v_C(1 - \kappa_C L/2) \\
v_R &= v_C(1 + \kappa_C L/2)
\end{aligned} \tag{16}$$

If desired, the left and right velocities can also be calculated using inputs of v_C and $\dot{\phi}$.

B.2 Pure pursuit controller

The pure pursuit path tracking algorithm as used in land-based navigation [10]. This application was inspired by ASI's Guideline Robotic Convoy System [9]. A circle is drawn with the follower tangent to the circle and the predecessor as a point on the circle. A new vehicle reference frame is used to simplify calculations.

- x inertial frame x coordinate.
- y inertial frame y coordinate.
- ϕ inertial frame vehicle heading.
- θ vehicle frame heading to preceding vehicle.
- d inter-vehicle distance.
- e_x vehicle frame longitudinal error.
- e_y vehicle frame lateral error $e_y = l\cos\theta$.
- r circle radius.
- κ circle curvature.

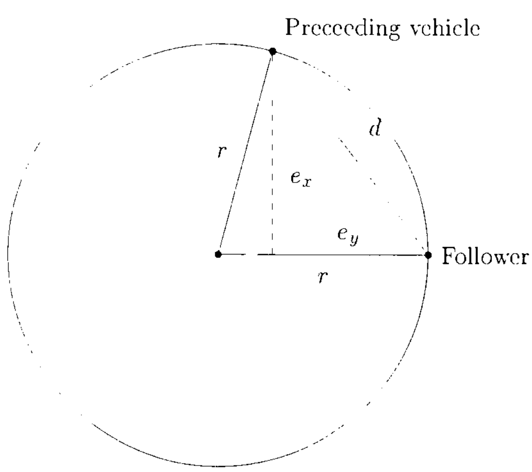


Figure 6: The coordinate frame for the pure pursuit controller.

$$\begin{aligned}
e_x &= \cos(\phi)(x_{n+1} - x_n) + \sin(\phi)(y_{n+1} - y_n) \\
e_y &= -\sin(\phi)(x_{n+1} - x_n) + \cos(\phi)(y_{n+1} - y_n)
\end{aligned} \tag{17}$$

$$\begin{aligned} \theta &= \arcsin(e_x/d) \\ \theta &= \arccos(e_y/d) \\ \theta &= \arctan(e_x/e_y) \end{aligned} \tag{18}$$

$$\begin{aligned} e_x &= d\sin\theta \\ e_y &= d\cos\theta \end{aligned} \tag{19}$$

$$e_x^2 + e_y^2 = d^2 \tag{20}$$

$$(r - e_y)^2 + e_x^2 = r^2 \tag{21}$$

$$\begin{aligned} (r - e_y)^2 + e_x^2 &= r^2 \\ r^2 - 2re_y + e_y^2 + e_x^2 &= r^2 \\ e_y^2 + e_x^2 &= 2e_y \\ d^2 &= 2re_y \\ r &= \frac{d^2}{2e_y} \end{aligned} \tag{22}$$

$$\kappa = \frac{1}{r} = \frac{2e_y}{d^2} = \frac{2\cos\theta}{d} \tag{23}$$

Steering is not independent of velocity. As velocity increases, the available curvature decreases.

B.3 Longitudinal controller

Any longitudinal controller can be used in this system. Here are some potential systems.

Bidirectional impedance model I'm going to use Soo-Yeong Yi's Impedance control with some basic alterations. This system uses forces between vehicles as acceleration inputs.

- d_n is the desired inter-vehicle distance.
- v_n is the velocity.
- d_{des} is the desired inter-vehicle distance.

$$d_n = \sqrt{(x_{n+1} - x_n)^2 + (y_{n+1} - y_n)^2} \tag{24}$$

$$\dot{v} = \frac{k}{m}(d_n - d_{des}) - \frac{k}{m}(d_{n-1} - d_{des}) + \frac{c}{m}(\dot{v}_{n+1} - \dot{v}_n) - \frac{c}{m}(\dot{v}_n - \dot{v}_{n-1}) \tag{25}$$

Unidirectional constant distance

$$\dot{v} = \frac{k}{m}(d_n - d_{des}) + \frac{c}{m}(\dot{v}_{n+1} - \dot{v}_n) \tag{26}$$

This system is the simplest to implement, but the platoon isn't string stable.

Unidirectional constant time

$$\dot{v} = \frac{k}{m}(d_n - h * v_n) + \frac{c}{m}(\dot{v}_{n+1} - \dot{v}_n) \tag{27}$$

C Code to interface with the vehicle platoon

```
# This program listens to an xbee module on COM14, and sends
# necessary commands to the xbee module on COM15.
import serial
import time
import sys
import numpy as np
import signal

ser = serial.Serial('COM15',115200,timeout=0.017)
back = serial.Serial('COM14',115200,timeout=2)
def signal_handler(signal,frame):
        print ('\nControl C was pressed. Now sending the kill signal.\n')
        t = time.time()
        while (time.time() - t < 10):
                ser.write('c\n')
        print 'Really done now\n'
        sys.exit(0)
signal.signal(signal.SIGINT, signal_handler)


l =[0] * 6
ser.write('t\n')
ser.write('t\n')
ser.write('t\n')
ser.write('t\n')
ser.write('t\n')
t0 = time.time()
t=t0
while 1:
        outstring = back.readline()
        outstring = outstring[1:-1]
        counter=1
        for i,k in zip(outstring[0::2], outstring[1::2]):
                l[counter]= np.frombuffer(buffer(i+k),dtype=np.float16)[0]
                counter = counter + 1
        l[0] = time.time() - t
        print "\ntime,\tcarTime,\tv,\tx,\ty,\theading"
        print ["%0.2f" % i for i in l]
print "Finished\n"
```

A References

[1] R. Rajamani, *Vehicle dynamics and control.* Springer, 2011.

[2] C. Chien and P. Ioannou, "Automatic vehicle-following," in *American Control Conference, 1992.* IEEE, 1992, pp. 1748–1752.

[3] M. Y. Abualhoul, M. Marouf, O. Shagdar, and F. Nashashibi, "Platooning control using visible light communications: A feasibility study," in *Intelligent Transportation Systems-(ITSC), 2013 16th International IEEE Conference on.* IEEE, 2013, pp. 1535–1540.

[4] T. Fujioka and M. Omae, "Vehicle following control in lateral direction for platooning," *Vehicle System Dynamics*, vol. 29, no. S1, pp. 422–437, 1998.

[5] T. C. Ng, J. I. Guzman, and M. D. Adams, "Autonomous vehicle-following systems: A virtual trailer link model," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on.* IEEE, 2005, pp. 3057–3062.

[6] P. Petrov, "Nonlinear adaptive control of a two-vehicle convoy," *Open Cybernetics & Systemics Journal*, vol. 3, pp. 70–78, 2009.

[7] D. Yanakiev and I. Kanellakopoulos, "A simplified framework for string stability analysis in ahs," in *Proceedings of the 13th IFAC World Congress*, 1996, pp. 177–182.

[8] D. Cripps, "Unmanned ground vehicles: High level system model," 2015, date reflects date obtained, not the creation date.

[9] J. L. Ferrin, "Controls of a tether-based robotic convoy," Master's thesis, Utah State University, 2007.

[10] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," DTIC Document, Tech. Rep., 1992.

Chapter 4

System Handbook

SATS Vehicle Platoon Handbook

Secure Automated Transportation Systems Group
Electrical and Computer Engineering
Utah State University
Logan, Utah

April 30, 2015

Contents

List of Figures

List of Tables

Revision History

Rev	Description	Author	Contact	Date
1		Daniel Dunn	d.dunn@aggiemail.usu.edu	1/1/2015
2	Added instructions chapter	Samuel Mitchell	samuel.mitchell@aggiemail.usu.edu	4/24/2015
3				
4				
5				
6				
7				
8				
9				
10				

Chapter 1

System Design

1.1 Project Description

The intent of this project is to construct a platoon of autonomous vehicles. The lead vehicle is under the direct control of an operator via 2.4GHz radio transmitted command signals. Each vehicle after the lead vehicle (follower vehicles) will run in a completely autonomous mode. Each follower vehicle will track the vehicle directly in front of it. The follower vehicles must maintain a given time headway separation from the preceding vehicle. The time headway input is given in seconds, the relative distance between vehicles is then a function of the current velocity of that vehicle multiplied by the time headway. The vehicle platoon must be able to travel in excess of 20mph while maintaining the proper inter-vehicle spacing as well as maintain stability as the platoon accelerates and decelerates.

Initially the main focus of this project is to maintain the inter-vehicle spacing and platoon stability. Therefore, the required longitudinal controller has been designed first. However, before the platoon can operate properly a lateral control system will need to be implemented as well.

The purpose of building this vehicle platoon is to test various vehicle platooning control measures that have been suggested in industry and academia. The objective is to bring attention to the flaws in many of these suggested designs by physically demonstrating conditions that will cause instability in the platoon. Instability could be introduced by a vehicle (attacker) that does not follow the control law of the platoon, perhaps someone trying to cause destructive collisions.

Initially this project will consist of one lead vehicle and one follower vehicle. This design must be scalable. The intent is to fully implement a platoon of 10 vehicles for demonstration purposes. The vehicles chosen for this project are differential steer with two 4 diameter wheel on each side. Each vehicle has two dc motors, each motor drives two wheels on one side of the vehicle. The motors are driven by a two channel motor controller that accepts PWM input signals. The duty cycle of this signal will be the output from the longitudinal and lateral controllers. These controllers are implemented discretely in the C coding language.

Initially ultrasonic sensors will be used to measure the relative distance between vehicles. This distance is the main feedback to the longitudinal controller. Two encoders are installed on each vehicle, one per motor, the output is used as a minor velocity feedback loop to the controllers. Additionally a Doppler radar sensor on each vehicle is used to measure the relative velocity between the current and preceding vehicles. The output of this sensor will be used in a feed forward configuration to the longitudinal controller.

To ensure that the platoon can be remotely shut down each vehicle will have an XBee wireless module on-board. The modules will transmit a pulse signal that will be monitored by a watchdog timer aboard each vehicle. If the pulse is interrupted the platoon will be stopped and set in a safe off state.

Each vehicle must also have additional onboard memory. This will allow the data output from each sensor to be stored aboard each vehicle. This data can be uploaded to a computer and used to create plots of the real time performance of the controllers aboard each vehicle as well as the interaction between vehicles.

All of the functions of the vehicle described above are implemented on a Texas Instruments EK-TM4C123GXL ARM Cortex M4F micro-controller. There will be one microcontroller aboard each vehicle running with a clock frequency of 20MHz. The following resources of the microcontroller are being used to implement this project:

- Pulse Width Modulation (PWM) modules 0 and 1
- I2C module 0
- Quadrature Encoder Input (QEI) modules 0 and 1
- UART modules 0 and 1
- GPIO ports for TTL input and output

1.2 Hardware

1.2.1 Vehicle Specifications

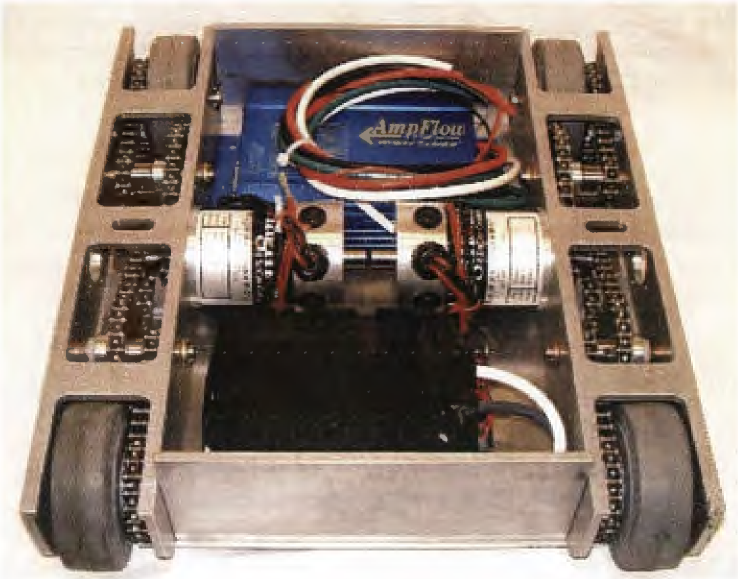


Figure 1.1: Light Weight BattleKit

Standard Vehicle Configuration (battlekits.com)
Light weight kit Length: 45 cm Width: 39.6 cm Weight: 16.78 kg (including motors, controller, and 2 batteries)
Gear ratio 3.4 (high speed ratio)
Wheel diameter 10.13 cm (4in) (measured circumference 31.8 cm)
System voltage 36V (3 - 12V sealed lead acid (SLA) batteries in series) (../data_sheets/UB1250_D5741.pdf)
2 - E30-150 AmpFlow motors (battlekits.com/robot_motors.htm)
AmpFlow 160 motor controller (../data_sheets/AF160_Manual.pdf)
Differential steer
Average acceleration 7.8 m/s^2 Theoretical top speed 13.32 m/s (29.8 mph) (members.toast.net/joerger/squid/newtorquecalc.htm)

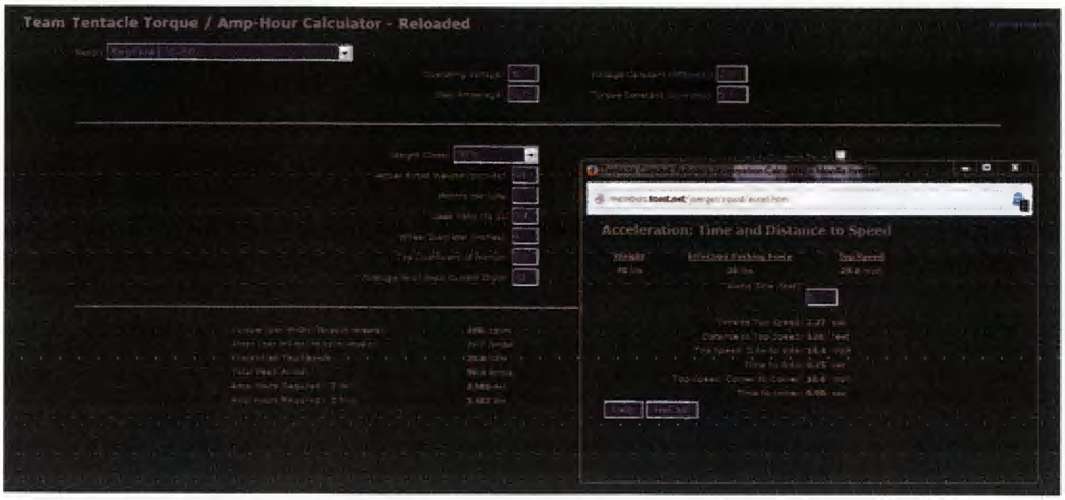


Figure 1.2: Output of the Team Tentacle Torque / Amp-Hour Calculator for the standard vehicle configuration

Enhanced Vehicle Configuration
Light weight kit Dimensions and weight same as standard configuration
Gear ratio 3.4 (high speed ratio)
Wheel diameter 12.7 cm (5in)
System voltage 36V (3 - 12V sealed lead acid (SLA) batteries in series) (./data_sheets/UB1250_D5741.pdf)
2 - F30-150 AmpFlow motors (battlekits.com/robot_motors.htm)
AmpFlow 160 motor controller (./data_sheets/AF160_Manual.pdf)
Differential steer
Average acceleration 10.0 m/s^2 Theoretical top speed 20.21 m/s (45.2 mph) (members.toast.net/joerger/squid/newtorquecalc.htm)

The main wiring diagram for each vehicle is shown in Figure1.3. As per the recommendations in the AF160 data sheet a flyback diode has been specified across the terminals of the main disconnect switch. The purpose of this diode is to dissapate voltage spikes caused when the motors turn off and generate reverse RMS.

1.2.2 Sensors

A variety of sensors are used for this project. These sensors provide the feedback necessary to implement a control scheme to maintain desired inter-vehicle separations and velocities.

- Lidar range finder provides inter-vehicle seperation distance and relative velocity between vehicles.
- Encoders, provide velocity of left and right wheel pairs, and cumulative distance traveled.
- Ultrasonic (optional), provides inter-vehicle seperation distance. The difference between subsequent sensor readings can be used to estimate relative velocity between vehicles.
- Radar (optional), Doppler shift is used to calculate the relative velocity between vehicles.
- Accelerometer, provides acceleration in X, Y, Z directions. The acceleration can be integrated to give estimates of velocity for short time durations.
- XBee, provides wireless communication based on the IEEE 802.15.4 protocol.

ROBOT SENSORS		
Sensor	**Manufacturer**	**Part No.**
Lidar	Pulsed Light	LL-905PIN-01
Encoders	US Digital	E2-500-375-IE-D-G-B
Ultrasonic	PARALLAX INC.	PING #28015
Radar	PARALLAX INC.	X-Band Motion Detector #32213
Accelerometer	DFROBOT	SEN0032
XBee	Digi International	XBP24BZ7SIT-004

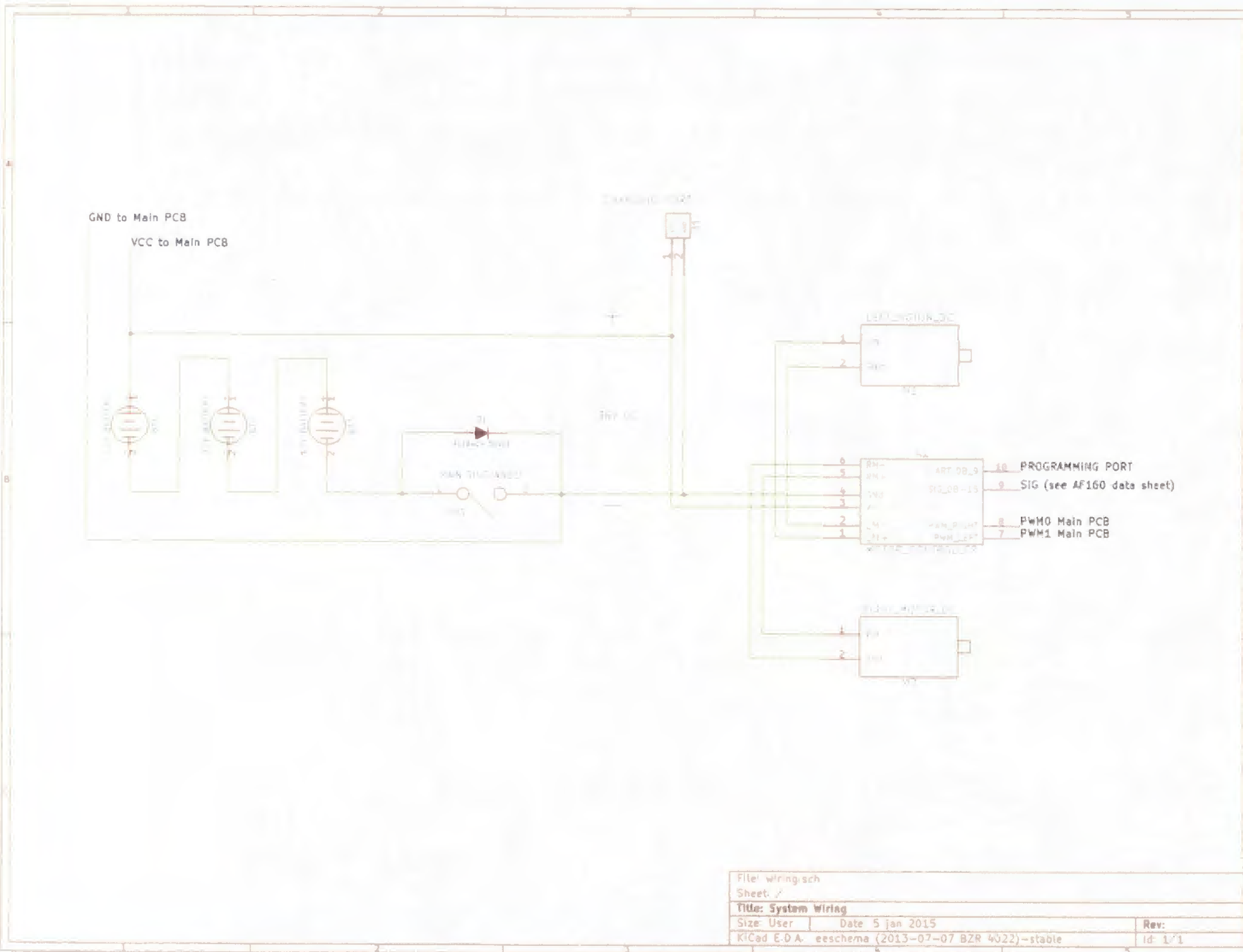
Table 1.1: Sensors

Sensor Cables

The battle kit robot platform used for this project uses the standard servo cable scheme for the PWM inputs. Because of this I decided to keep this wiring scheme for the rest of the sensors.

SENSOR CABLES		
PIN 1	GND	BROWN WIRE
PIN 2	+5V	RED WIRE
PIN 3	SIGNAL	ORANGE WIRE
PIN 5	OPTIONAL	YELLOW WIRE

Table 1.2: Sensor cable standard



GND to Main PCB
VCC to Main PCB
PROGRAMMING PORT
SIG (see AF160 data sheet)
PWM0 Main PCB
PWM1 Main PCB
File: wiring.sch
Title: System Wiring
Size: User
Date 5 jan 2015
Rev:
KiCad E.D.A. eeschema (2013-07-07 BZR 4022)-stable
Id: 1/1

1.3 System Architecture

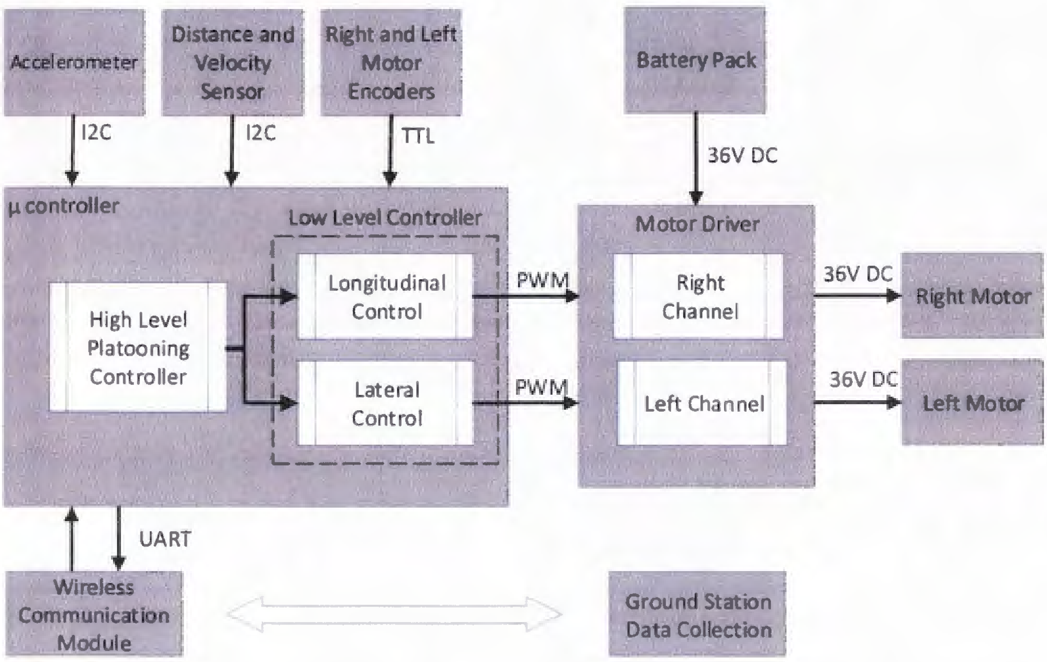


Figure 1.4: Top level system architecture

1.3.1 Micro-Controller Configuration

As mentioned before the micro-controller use for this project is the Texas Instruments TivaC EK-TM4C123GXL ARM Cortex M4F. The on-board peripheral driver library is utilized to configure the required peripheral devices. The complete software developer package EK-TM4C123GXL-KEIL which includes, TivaWare for C Series and Keil RVMDK for the Tiva C Series TM4C123G LaunchPad, can be found at: http://www.ti.com/tool/sw-ek-tm4c123gxl. Once the package is downloaded concise installation instructions can be found in the Documentation folder. Follow the instructions in the Quickstart-Driver-Installation-spmu287.pdf to setup the drivers required to flash the micro-controller. Follow the instructions in the Quickstart-Eval-Kit-Keil-spmu355.pdf to flash a simple program to the board and learn how to configure your own project. Utilizing this development package the micro-controller has been configured as follows:

- System clock set at 20Mhz using PLL with crystal reference.
- GPIOA PIN6 set to input. (discontinued)
- (discontinued) GPIOA PIN6 interrupt set to rising edge.
- (discontinued) GPIOB PIN4 used as input and output.

- (discontinued) GPIOB PIN4 interrupt set to both edges.
- Timer0 set periodic count-down, the count value is set in a function used to set the main loop sample time.
- Timer0 interrupt set for to interrupt on a timeout event.
- (discontinued) Timer1 set periodic count-up, used to determine ultrasonic sensor time of flight.
- (discontinued) Timer2 set periodic count-up, used to determine Doppler sensor time of flight.
- UART0 enabled for 15200 baud, this UART operates over the USB programming port.
- UART1 enabled for 15200 baud, this UART is used to send data through the XBee wireless link.
- PWM0 and PWM1 enabled, PWM0 is used to output longitudinal control signals, PWM1 for lateral control signals.
- QEI0 and QEI1 enbled, these peripherals are used to gather distance and velocity input from the motor encoders.
- IC20 is enabled and used to program the lidar and accelerometer and receive data from these sensors.

PB5 is for possible future expansion with the addition of a second ultrasonic sensor. It has currently not been configured. The following tables show the pin assignments of the TivaC micro-controller:

FIRST ROW OF DOUBLE HEADER PINS			
Pin #	**Device**	**Pin #**	**Device**
+3.3v	Sensor Power	VBUS	
PB5	Ultrasonic1 Trigger/Signal	GND	
PB0	UART1 U1Rx	PD0	Lidar Power Enable
PB1	UART1 U1Tx	PD1	Lidar Mode
PE4	Radio RX Input	PD2	
PE5	Radio RX Input	PD3	QEI0 Index
PB4	Ultrasonic0 Trigger/Signal	PE1	Radio RX Input
PA5		PE2	Radio RX Input
PA6	Doppler Radar Out	PE3	Radio RX Input
PA7	Doppler Radar Enable	PF1	

Table 1.3: TivaC EK-TM4C123GXL pin assignments

SECOND ROW OF DOUBLE HEADER PINS			
Pin #	**Device**	**Pin #**	**Device**
PF2		GND	
PF3		PB2	I2C0 SCL
PB3	I2C0 SDA	PE0	Radio RX Input
PC4	QEI1 Index	PF0	
PC5	QEI1 PhA	RST	
PC6	QEI1 PhB	PB7	PWM1
PC7		PB6	PWM0
PD6	QEI0 PhA	PA4	
PD7	QEI0 PhB	PA3	
PF4		PA2	

Table 1.4: TivaC EK-TM4C123GXL pin assignments

NOTE: PA0 and PA1 correspond to UART0 UORx, and UART0 U0Tx respectively. These pins are not broken out.

Sensor Data

The readings from the various sensors are stored in a custom data structure **VEHICLE** which is an instance of a custom data type struct **VEHICLE_t** defined in **VP_globals.h**. These measurements will be used as the input to the system control loops.

VEHICLE.{variable}			
State	**System Variable**	**Units**	**Type**
Velocity of right wheels	right_velocity	meter/sec	float
Velocity of left wheels	left_velocity	meter/sec	float
Velocity center of mass	velocity	meter/sec	float
Relative velocity	delta_velocity	meter/sec	float
Absolute distance	traveled_distance	meter	float
Relative distance	delta_distance	meter	float
Acceleration along X	x_accel	meter/sec/sec	float
Acceleration along Y	y_accel	meter/sec/sec	float
Acceleration along Z	z_accel	meter/sec/sec	float

Table 1.5: Table of system states available and the associated software variables.

1.3.2 PWM Configuration

The AF160 motor controller used for this project determines the period of the PWM signal as well as what duty cycle corresponds to reverse, stop, and forward. By default the AF160 recognizes the following ranges of PWM input, These values correspond with the standard pulse widths used for servo actuators:

- 1.0mSec pulse width, motor output full reverse

- 1.5mSec pulse width, zero motor output
- 2.0mSec pulse width, motor output full forward
- Maximum PWM signal frequency 300Hz

Input to AF160 Motor Controller		
PWM Pulse Width (300Hz)	**Duty Cycle**	**Motor Output**
1.0 mSec	-100%	Full Reverse Velocity
1.5 mSec	0%	Zero Velocity
2.0 mSec	100%	Full Forward Velocity

Table 1.6: Default PWM input values for AF160 DC motor controller.

The maximum PWM frequency of 300Hz is used to insure the output is updated as quickly as possible. This gives this component of the system the highest possible bandwidth. It is important that the low level components of the control system have high bandwidth because the bandwidth of each progressive level of control must be lower than its subsystems. The higher the bandwidth of the subsystems, the higher the bandwidth of the top level system. The top level system bandwidth will determine the overall time response of the system as a whole.

To achieve the desired signal pulse width the timer reload value for the PWM modules of the micro-controller is calculated as follows:

$$PWM_{timer\_reload} = \frac{system\ clock\ frequency}{2} * (desired\ pulse\ width\ in\ mSec)$$

For a system clock speed of 20MHz the PWM timer reload value for zero motor output is 15000. This is used as the default base value any required control effort will add to this base value up to a maximum PWM timer reload value of 20000. This corresponds to the max 2mSec pulse width.

Module PWM0 is used to output the control effort for longitudinal motion. PWM1 will be used to output the lateral control effort. As of the writing of this document only the longitudinal control is being designed. It will be important to verify the interaction between the longitudinal and lateral control schemes. I believe that the AF160 will take half the PWM1 control effort output and add this to one motor output and subtract from the other. This will result in the overall net longitudinal output being equal in the forward travel direction of the robot. As of yet this theory has not been verified. This will detirmine how the robot turns, whether it pivots about the center of mass or skews to the left or right when turning.

The relationship between control input to the AF160 and the system response will be discussed in later sections. A system model will then be developed and used to design appropriate control schemes.

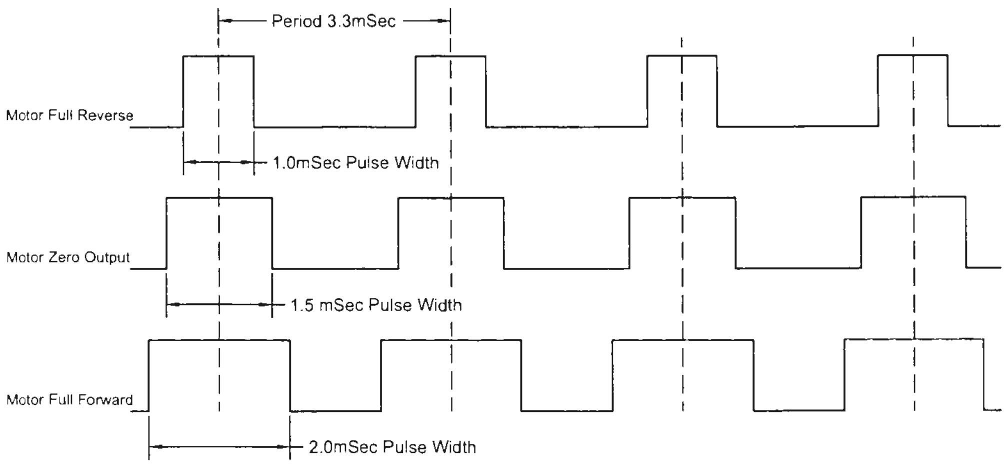


Figure 1.5: PWM output at frequency of 300Hz

1.3.3 Encoder Configuration

Two quadrature encoders are used for this project. One encoder for the right motor and one for the left motor. The encoders are used to track the rotation of the motor shafts. The number of rotations during a specific time period is linearly related to the velocity of the motor shaft.

The index of each encoder can be used to track the number of rotations. This can be used to track estimate the cumulative distance traveled by the robot. However, if the wheels slip or spin on the operating surface this measurement will be innacurate.

The output of the encoders is connected to the quadrature encoder input modules of the TivaC. The velocity of the left and right sets of wheels on the robots can be calculated as follows:

$$\frac{meters}{sec} = QEICOUNT * \frac{samples}{sec} * \frac{rev}{encoder\ PPR} * \frac{1}{gear\ ratio} * \pi * (diameter\ of\ wheel\ in\ meters)$$

The QEI modules can be configured for quadrature encoder input. In this case the Pulses Per Revolution (PPR) will be four times the Cycles Per Revolution (CPR). I have however, found a descrepancy in this process. The particular encoder used for this project is 500 CPR. Using quadrature encoding the PPR should be 2000. When I use this divisor the velocity calculated using the above equation is half what I expect to see. I performed a simple experiment by running the motor for a short period of time and capturing the number of encoder pulses counted by the micro-controller during the test period. I also attached a piece of string to one of the wheels of the robot and allowed it to wrap around the wheel as it rotated. By comparing the linear measurement of the string wrapped around the wheel versus number of encoder counts I calculated that I am getting 1000 PPR not the 2000 expected. I am not sure at this point if the micro-controller is improperly configured or the encoders are not actually 500 CPR. This dilemma will requre further investigation.

For future reference I have included drawings of the pin layout for the US Digital encoders used for this project shown in Fig. 1.6. The phase relationship between the encoder signal outputs is shown in Fig. 1.7.

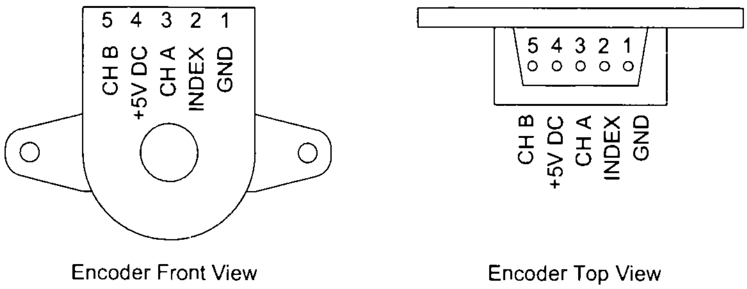


Figure 1.6: US Digital encoder pin layout

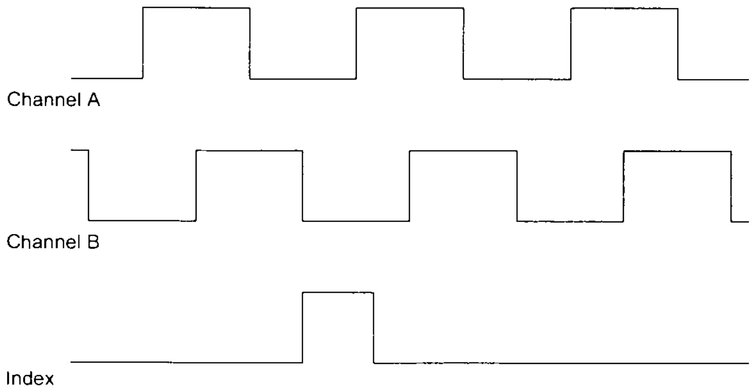


Figure 1.7: Quadrature encoder signal output

1.3.4 Lidar

The Lidar-LITE range sensing module is being used for this project. The Lidar module is preferable to the Ultrasonic Sensors due to the 40m range. This section will be completed as characterization is performed.

1.3.5 Ultrasonic Sensors

An ultrasonic sensor is used for this project to measure the distance between vehicles. Currently the main program loop on the TivaC is based off the maximum time required to recieve a reading from the PING ultrasonic sensor. I have allowed 20mSec which is just longer than the time required to recieve a maximum distance reading.

Theory of Operation (from PING ultrasonic documentation). The PING sensor detects objects by emitting a short ultrasonic burst and then "listening" for the echo. Under control of a host microcontroller (trigger pulse), the sensor emits a short 40 kHz (ultrasonic) burst. This burst travels through the air at about 1130 feet per second, hits an object and then bounces back to the sensor. The PING))) sensor provides an output pulse to the host that will terminate when the echo is detected, hence the width of this pulse corresponds to the distance to the target.

To receive the signal from the ultrasonic sensor GPIOB PIN4 is used as an input/output. The pin must first be cofigured as output to send the t_{out} pulse to the sensor. The pin is then configured as input and the GPIOB interrupt is enabled to detect the rising and falling edge of the input pulse. The width of this pulse is captured using timer1 configured as a single 32-bit wide timer. A 32-wide timer must be used to capture the maximum pulse width corresponding to $(.0185) * (20000000) = 370000$, while using a 20MHz clock speed. The pulse width is used to calculate the real value in meters:

$$distance\ (meters) = \frac{(speed\ of\ sound) * t_{IN}}{2}$$

$$speed\ of\ sound\ (meters/sec) = 331.5 + 0.6 * (temp\ of\ air\ (celcius))$$

The surrounding air temperature will change the distance measurement by 4 % in a temperature range of 50°F to 90°F. This can cause measurement error in the range of centimeters. For now I have used the speed of sound corresponding to an air temperature of 70°F. An air temperature sensor could be added to correct this issue.

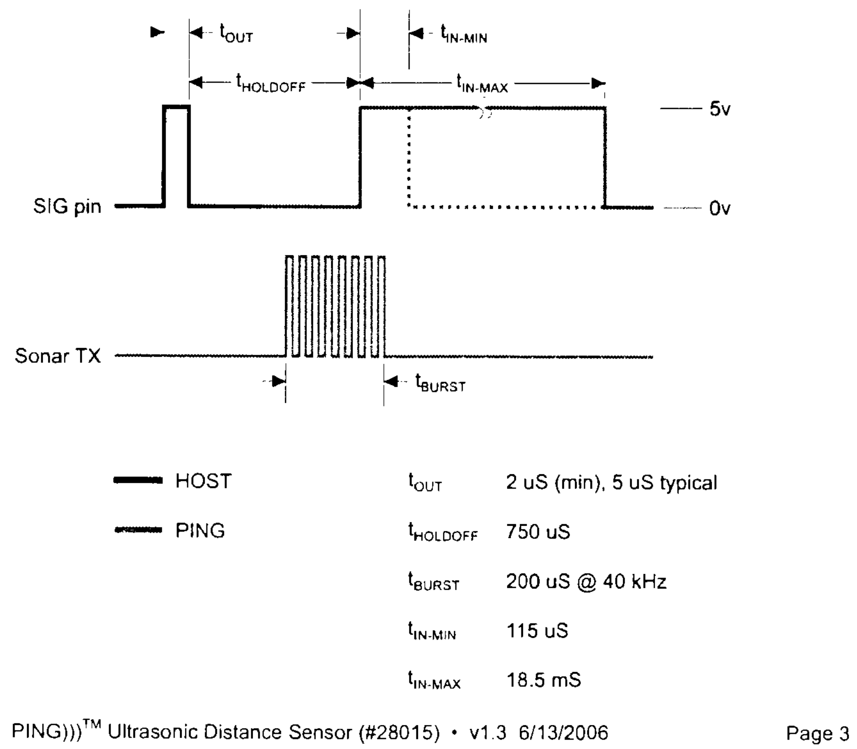


Figure 1.8: PING ultrasonic theory of operation

1.3.6 Doppler Radar

This section will be completed when a Doppler radar has been selected.

1.3.7 Accelerometer

The ADXL345 accelerometer is being used for this project. This section will be completed as characterization is performed.

1.3.8 XBee Wireless Module

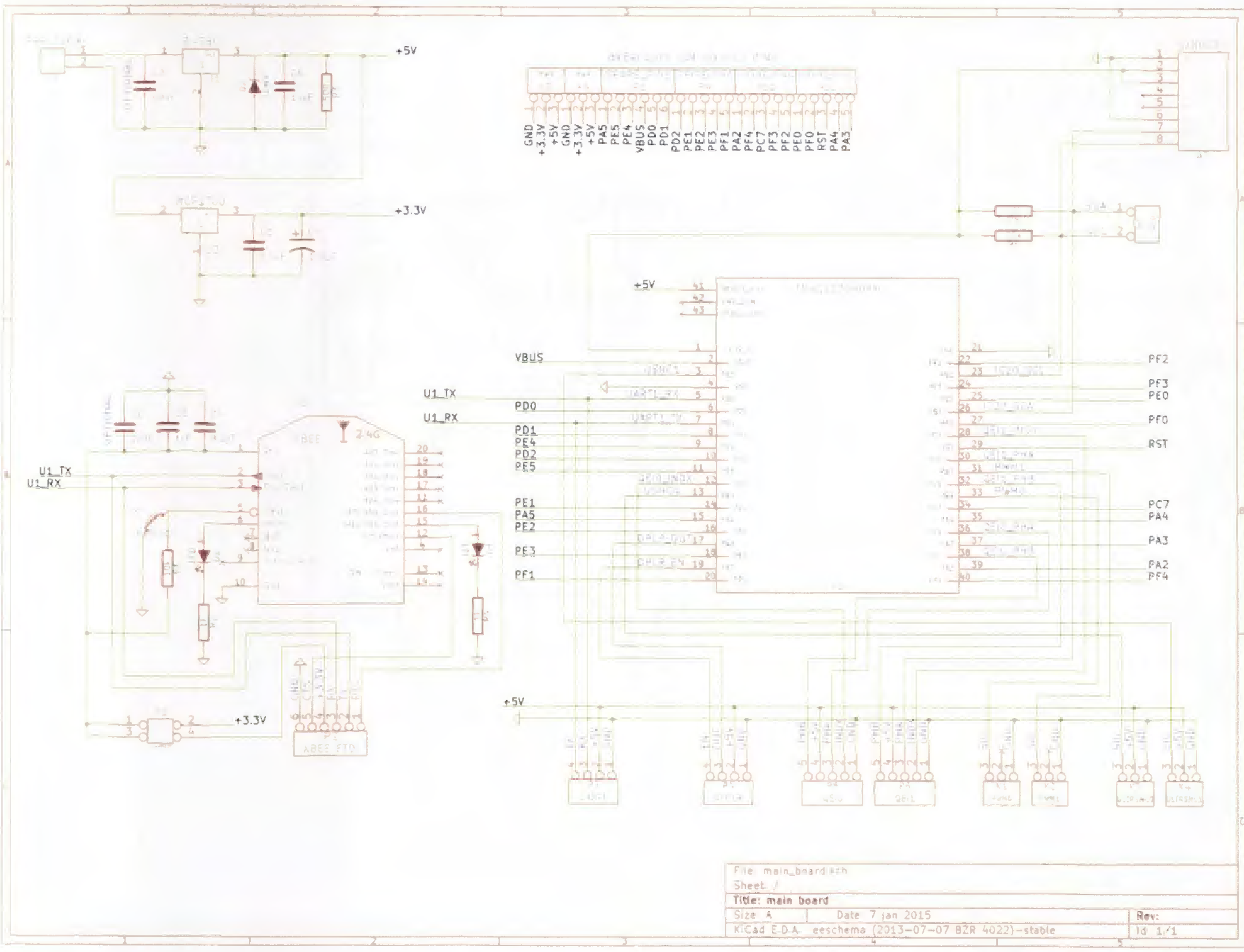
The XBee

1.3.9 Main PCB Design

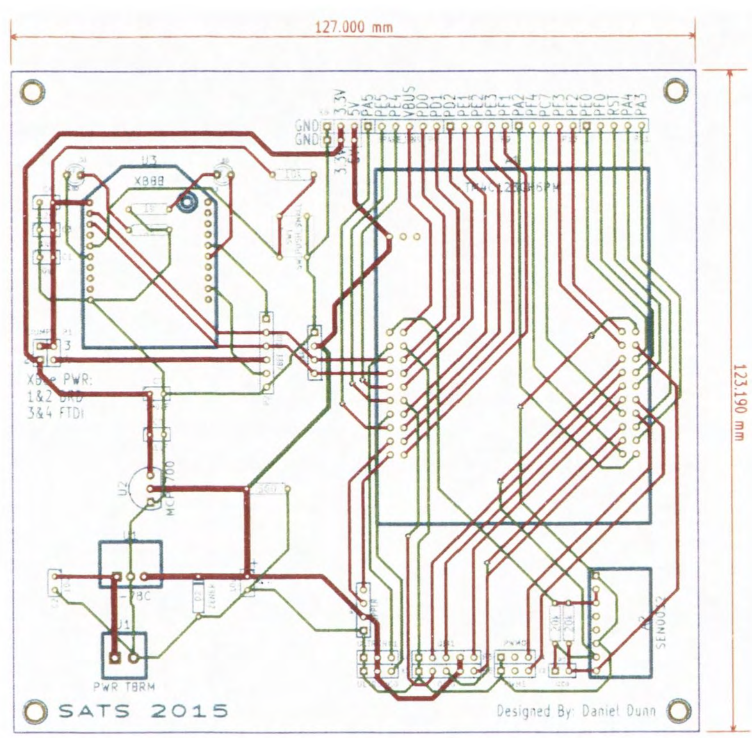
The PCB was designed to support the following components:

- XBee Wireless Module

- Accelerometer
- Lidar
- 2 Quadrature Encoders
- 2 Pulse Width Modulation Channels
- Tiva C Launchpad Micro-Controller
- I2C Expansion
- 2 Ultrasonic Sensors
- Doppler



+5V
+3.3V
GND
+3.3V
+5V
GND
+3.3V
+5V
PA5
PE5
PE4
VBUS
PD0
PD1
PD2
PE1
PE2
PE3
PF1
PA2
PF4
PC7
PF3
PF2
PF0
PE0
RST
PA4
PA3
2.4G
U1_TX
U1_RX
VBUS
PD0
PD1
PE4
PD2
PE5
PE1
PA5
PE2
PE3
PF1
PF2
PF3
PE0
PF0
RST
PC7
PA4
PA3
PA2
PF4
+5V
+3.3V
File: main_board.sch
Sheet: /
Title: main board
Size: A
Date: 7 jan 2015
Rev:
KiCad E.D.A. eeschema (2013-07-07 BZR 4022)-stable
Id: 1/1



File: main_board.brd		
Sheet:		
Title:		
Size: A	Date: 7 jan 2015	Rev:
KiCad B.D.A		Id: 1/1

Chapter 2

Instructions

2.1 Vehicle Assembly

The purpose of this section is to detail the vehicle assembly procedure as used by the Secure Autonomous Transportation Systems (SATS) Group. If any alterations to the vehicle platoon are performed, the updates should be detailed in this document.

2.1.1 Quadrature Encoders

Installation process of the E2-500-375-IE-D-G-B Optical Kit Encoder.

1. Remove one motor and gear assembly using a 3/32" allen wrench on the 9 bolts that attach the assembly to the base.

2. Attach the encoder base to each motor. Groove side out. The U shaped groove should face up. 3/8 length screws.

3. Slide the counter tab (round, clear) onto the motor shaft.

4. Move the encoder (brown with pins) onto the encoder base. The pins face up, with the groove on the motor side.

2.1.2 Battery Padding

Cut foam padding to fit the dimensions in Table 2.1. Carefully insert the batteries with the terminals pointed toward the right side of the vehicle, as shown in Figure 2.1.

Using hot glue, affix the 22 x 3.5 x 2.5 pad to the underside of the electronics platform. The piece should be 6cm from the front with 1cm on either side.

Table 2.1: Battery padding dimensions (cm)

Qty.	Length	Width	Depth
2	21.5	9	2.5
2	14	9	1.5
1	22	3.5	2.5

2.1.3 Wire Connections

Motor

Attach the motor cables to the corresponding spaces on the motor controller board. The motor controller board is at the rear of the vehicle. This will assist in locating the right and left channels.

Table 2.2: Required wires

Qty.	Name	Color	Length (cm)	Gauge	Connector 1	Connector 2
4	Motor	R, Blk	-	-	Ring terminal	-
2	Motor controller	R, Blk	45	8	Ring terminal	Wire
1	Battery long	R	90	8	Female quick slide	Wire
1		Blk	70	8	Female quick slide	Wire
2	Battery short	Blk	20	8	Female quick slide	Female quick slide
2	Kill switch	R, Blk	35	8	Large ring terminal	Wire
2	Charger	R, Blk	25	12	Anderson	Wire
2	PCB	R, Blk	30	12	Wire	Wire
2	PWM ribbon	Blu, V, Gry	70	40	Female pin header	Male pin
2	QEI ribbon	Br,R,O,Y,Gn	70	40	Female pin header	Female pin header

Battery

Attach the female quick slides of the battery long and short cables to the terminals of the batteries, as shown in Figure 2.1.

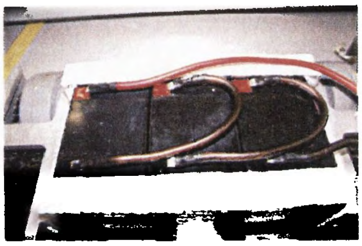


Figure 2.1: Battery wiring configuration.

Attach the wire end of the battery long cables to the power block, as shown in Figure 2.2.

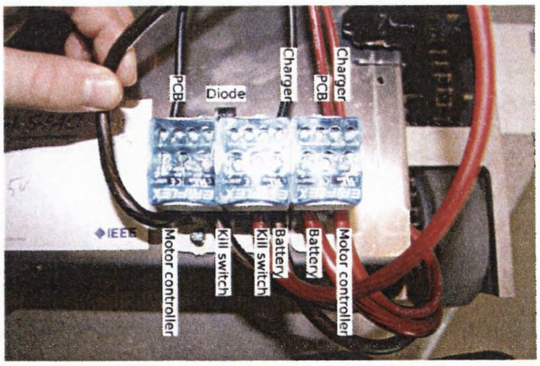


Figure 2.2: PCB wiring configuration.

Motor Controller

Attach the ring terminals of the motor controller cables to the motor controller board, as shown in Figure 2.3
Attach the wire ends of the motor controller cables to the power block, as shown in Figure 2.2.

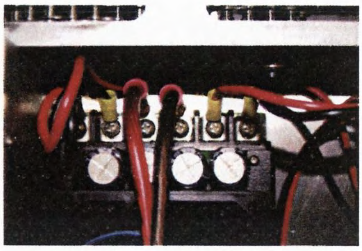


Figure 2.3: Motor control wiring configuration

Kill Switch

Attach the ring terminals of the kill switch cables to the kill switch, as shown in Figure 2.4. Wrap the ring terminals in electric tape.

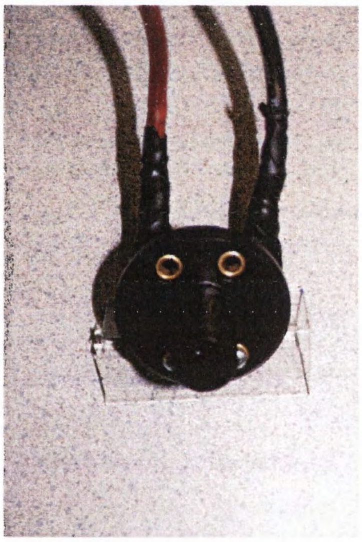


Figure 2.4: Kill switch wiring configuration

Attach the ring terminals of the kill switch cables to the power block, as shown in Figure 2.2.

Charger

Attach the wire ends of the charger cables to the power block, as shown in Figure 2.2.

PCB

Attach the wire ends of the PCB cables to the power block, as shown in Figure 2.2.

PWM Ribbon

Attach each PWM ribbon to the corresponding PWM wire from the motor controller board (right, left), as shown in Table 2.3. Attach each PWM ribbon to the corresponding PWM port on the PCB, as shown in Table 2.4.

Table 2.3: PWM wire to wire connection

MCB PWM	PWM Cable
O	Blu
R	V
Br	Gry

Table 2.4: PWM and QEI connection to the PCB

Cable	PCB Port
PWM Left	PWM 0
PWM Right	PWM 1
QEI Left	QEI 1
QEI Right	QEI 0

QEI ribbon

Attach each QEI ribbon to the its corresponding QEI module, as shown in Table 2.4.

2.1.4 PCB Assembly

Refer to the PCB drawing in Figure 1.10. The only correction for the system is to replace R10 with a wire.

2.1.5 Electronics Platform

Place all items on the platform using Figure and Table 2.5 as guide. After the components are placed, perform the wire connections described in Section 2.1.3.

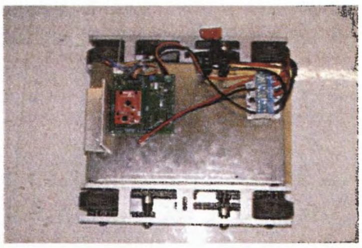


Figure 2.5: Electronics platform configuration.

Table 2.5: Electronics platform connection

Item	Qty	Diameter	Length (inch)
Angle iron bolt	2	#8	1/2
Kill switch bolt	2	#8	1/2
Power block bolt	2	#8	1/2
Electronics platform mounting bolt	4	#8	1/2
PCB spacer	4		
PCB bolt	4	#4-40	3/8

2.2 Platoon Operation

The vehicle platoon is configured to begin on a command from the leader. In order to prevent injury, follow these instructions in order.

1. Place vehicles in a straight line with 1 meter between the rear tires of each vehicle. See Figure 2.6.

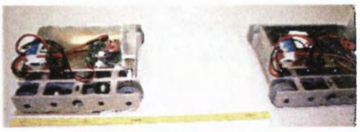


Figure 2.6: Vehicles with 1m spacing between rear tires.

2. Turn off all kill switches (horizontal position). See Figure 2.7.

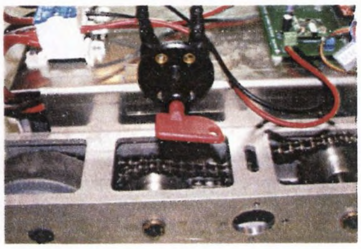


Figure 2.7: Vehicle with the kill switch in the off (horizontal) position.

3. Set the microcontroller to debug mode. See Figure 2.8.

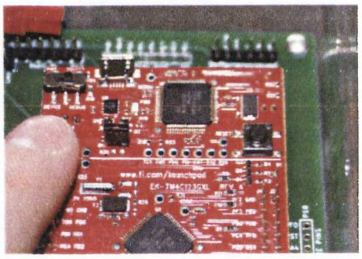


Figure 2.8: Tiva C microcontroller in debug mode.

4. Compile the uVision project with the macro LEADER defined. With the kill switch off, program the lead vehicle.
5. Compile the uVision project with the macro LEADER undefined. With the kill switch off, program the follower vehicles.
6. Start follower vehicles:
    - Turn on the kill switch (vertical position).
    - Set the microcontroller to device mode.

7. Start lead vehicle:
   - Connect the Right PWM channel to the ELEV channel of the AR400 Receiver. Ground is the gray wire.
   - Connect the Left PWM channel to the AILE channel of the AR400 Receiver. Ground is the gray wire.
   - Turn on the vehicle kill switch (vertical position).
   - Set the microcontroller to device mode.
8. Turn on the RC Radio. The lead vehicle is controlled via the right joystick.
9. Enjoy!

2.3 Controller Manipulation

This section outlines the process to insert or alter a control law. The super-loop structure is described, followed by a description of how to insert or alter various control laws. The software utilizes TivaWare 1.1, programmed in Keil uVision 4.

The main loop (shown in Figure 2.9) consists of 5 distinct sections:

- Update sensors
- Lateral and longitudinal control
- Low level control
- Communication
- Timing

When altering a controller, it is very important to refrain from altering anything besides the controller. Refer to the SATS Vehicle Platoon API for more information on the software flow.

2.3.1 Longitudinal control

The longitudinal controller accepts the required inputs and returns a velocity. If the control law utilized gives an acceleration, this must be manipulated to result in a velocity command.

2.3.2 Lateral control

The example lateral controller accepts the required inputs and returns a curvature. If the control law utilized returns a change in heading ($\dot{\phi}$), the low level lateral controller 2.3.3 must be altered to accept $\dot{\phi}$ as an input.

2.3.3 Low level lateral control

The low level lateral control accepts velocity and curvature commands, returning desired right and left velocity set points.

2.3.4 Low level velocity control

The low level velocity control utilizes a PID to control the desired velocity. This function returns a PWM command in range (-1,1). The gains in Table 2.6 result in a maximum settling time of 0.5s.

Table 2.6: Low level velocity control gains

Gain	Value
Proportional	1.5
Integral	3.7
Derivative	0.01

```
while(!kill_car)
{
  // Update sensors
  updateEncoders(&ENCODER, &VEHICLE);
  readLidar();

  // Lateral and longitudinal control
  longitudinal_constant_distance(&VEHICLE, &PRECEDING_VEHICLE, &
     CONTROLLER_SET_POINT);
  pure_pursuit_control(&VEHICLE, &PRECEDING_VEHICLE, &
     CONTROLLER_SET_POINT);

  // Low level control
  differential_control(&CONTROLLER_SET_POINT);
  u1 =  PID_control(CONTROLLER_SET_POINT.right_velocity, VEHICLE.
     right_velocity, pid_kp, pid_ki, pid_kd, -1.0, 9.0, SAMPLE_TIME,
      .01, &R_integrator, &R_differentiator, &R_error_dl);
  u2 =  PID_control(CONTROLLER_SET_POINT.left_velocity, VEHICLE.
     left_velocity, pid_kp, pid_ki, pid_kd, -1.0, 9.0, SAMPLE_TIME,
     .01, &L_integrator, &L_differentiator, &L_error_dl);
  // Send steering command
  PWMPulseWidthSet(PWM0_BASE, PWM_OUT_0, (PWM_BASE_PULSE + u2 *
     5000.0));
  PWMPulseWidthSet(PWM0_BASE, PWM_OUT_1, (PWM_BASE_PULSE - u1 *
     5000.0));

  // Communication

  // Execute timing operations
  time_stamp += SAMPLE_TIME;
  milliSecDelayUntil();
}
```

Figure 2.9: Vehicle platoon main loop.

Chapter 5

Reflection

Lateral Control of a Vehicle Platoon: Reflection

Sam Mitchell

Spring 2015

Overview

My honors thesis was inspired by an interest in working with embedded computers. As an engineer, I'm very interested in creating devices that don't need to be tied to a desktop computer to operate. My original plan was to produce a clever organizational device for food storage, but I was approached by Dr. Gerdes to assist him on his NSF-funded vehicle platooning research with the Secure Autonomous Transportation Systems (SATS) Group. So I took on a vehicle platooning project.

The focus of the SATS Group is to analyze the security and reliability of the automated highways of tomorrow. My project is a subset of the overarching goal - I assume that standard lanekeeping algorithms won't work, and vehicles on the road must rely on vehicle following to maintain stability. In other words, the main goal of the project is to create a line of vehicles that follow one after another. This was especially appealing to me because I could interface with hardware and software to produce an interesting product.

The basic setup of the project was to implement a following system on a platoon of vehicles. This started by analyzing three available control systems. Once that analysis was completed, I assembled 10 vehicles and wrote software to run one of the controllers.

Research

The first big challenge of working on my thesis was my utter and complete lack of experience with control systems. I started by diving into the current literature on vehicle following; my goal was to implement an available controller in Matlab. It took months before I had a working simulation. As my deadlines flew by, I started to get nervous about whether I was going to have a product to present by the end of the year.

I finally got one of the controllers working by November, followed by another in December. After looking at those two controllers, I determined that no easy solution was readily available, so I developed a third controller that would be easy to work with. This was based largely off of the thesis of a recent graduate (Ferrin, 2007). This controller outperformed the others in simulation, so it was selected for implementation.

It was a proud moment when the controller that I designed worked better than the systems I had found from professional journals. It is quite apparent that most of the papers I had read were more concerned with producing a novel method than developing a controller that simply works. This was the first time that I saw that my research mattered outside of my lab. I realized what I was doing was something where I could contribute to the scientific community at large.

At this point I prepared a poster for the Student Research Symposium that outlined the work that I had done up to this point. The symposium was a strange experience for me because I had prepared with the poster, but I hadn't thoroughly prepared a speech to give to the interested crowd of people who approached me. This lack of preparation ensured that the judge was the first person to approach my poster. I mangled the presentation, and the judge thought I didn't do any worthwhile work. This was a stark contrast to the elation I had felt at producing the working controller. This showed me that it's very important to make sure that any product I present must be sold well.

While I was doing research on controllers, I still had to assemble the vehicles to be used for implementation. On top of my many hours working on the cars, I led a team of six people in construction efforts, resulting in roughly 150 hours of work. The most difficult part of this was to ensure that everyone had a meaningful part of the project to work on.

Writing the software for my thesis was the most frustrating time. Interfacing with sensors and other peripherals that I was working with was very buggy.

After I was having troubles with excess delays, I spoke with Dr. Gerdes about methods to fix the problem. He suggested that I rewrite the software to perform operations as needed instead of in a sequential manner. This made me nervous, because I had to present the working vehicles in a matter of days. Instead of fretting about the request. I made the modifications to the code. This addition worked more reliably than the original program structure.

Another big struggle with the project development was the communication protocol. The cars would get hung up on inter-vehicle communication, and anything sent from my computer would cause the vehicles to freeze up. I tried to alter the pre-built interface for the communication and reduce the message size, but nothing worked. This was the final result of the system — a vehicle would occasionally receive a position vector from the preceding vehicle and rush to compensate.

I was wrestling with the communication problem through finals week. This was in preparation for the second public forum where I presented my research, Electrical and Computer Engineering Senior Design Night. After many late nights in preparation for this event, the system didn't reliably function as intended. It was very frustrating to do so, but I stopped development the day before the presentation to allow time for documentation. I took the cars out to USU's test track and captured some video of the vehicles behaving in an erratic manner.

Since completing the requirements for this project, I have continued forward with the work, and I intend to utilize the system I developed as a platform to build future research on. My master's thesis will focus on utilizing signals of opportunity to maintain platoon stability in adverse environments.

Final Reflection

Overall, the work I did on my thesis gave me insight into what researchers might do in my field. Through most of the effort, I didn't enjoy the process. I felt like I didn't understand what was going on, and I couldn't figure out how to make progress. After months of working that way, suddenly I understood what was going on, and I enjoyed the work much more.

Despite the many struggles over the past year, this experience helped me become a better engineer. Because of my project, I feel confident that I will be able to approach various robotics projects in the future and understand how to structure a plan for and complete a large project. The thesis development process truly has been a capstone of my undergraduate experience.

Chapter 6

Professional Author Bio

Sam Mitchell recently completed the requirements for the USU Bachelor's of Science Degree in Electrical Engineering, with a Minor in Mathematics. During his time at USU, Sam served as the USU IEEE Student Branch Chair. While serving in this capacity, he organized two outreach conferences with over 100 high school and college participants. Sam was also the recipient of the following awards: USU's Presidential Scholarship, Dean's List five times, D. Walter Dansie Engineering Scholarship, and an URCO Grant for his honors thesis. Sam has started pursuing a Master's Degree at USU, studying the security and stability of automated highway systems.