Title

Implementation Of The Regulation Layer Using Shift

Authors

Gulick, David
Alvarez, Luis
Horowitz, Roberto

Publication Date

1998-07-01

CALIFORNIA PATH PROGRAM
INSTITUTE OF TRANSPORTATION STUDIES
UNIVERSITY OF CALIFORNIA, BERKELEY

Implementation Of The Regulation Layer Using Shift

David Gulick, Luis Alvarez and Roberto Horowitz

California PATH Research Report
UCB-ITS-PRR-98-26

This work was performed as part of the California PATH Program of the University of California, in cooperation with the State of California Business, Transportation, and Housing Agency, Department of Transportation; and the United States Department of Transportation, Federal Highway Administration.

Report for MOU-287

July 1998

ISSN 1055-1425

CALIFORNIA PARTNERS FOR ADVANCED TRANSIT AND HIGHWAYS

Implementation of the Regulation Layer Using SHIFT [1]

David Gulick[2], Luis Alvarez[3] and Roberto Horowitz[4]

Department of Mechanical Engineering
University of California at Berkeley
Berkeley CA 94720-1740

[1]Research supported by UCB-ITS PATH grant MOU-287
[2]Graduate Student. Email: dgulick@me.berkeley.edu
[3]Postdoctoral researcher. Email: alvar@me.berkeley.edu
[4]Professor. Email: horowitz@me.berkeley.edu

Abstract

The implementation in SHIFT of regulation layer control laws for the hierarchical PATH AHS architecture is presented. SHIFT is a programming language developed at PATH for the simulation of hybrid systems. The implemented regulation layer control laws are derived after the safe-feedback based maneuvers designed in previous PATH projects. These maneuvers were modified to use the acceleration of vehicles as the control signal. Simulation results are included along with a summary of the code developed.

Keywords

Automated Highway Systems, hybrid systems, regulation layer control, safe vehicle maneuvers, feedback based maneuvers.

Acknowledgments

The authors wish to thank the team that developed SHIFT for their support to implement these regulation layer controllers, in particular the assistance of Akash Despande, Marco Antoniotti and Alex Göllü was very important in the debugging phase.

Executive Summary

The design and implementation of Automated Highway Systems requires intensive simulation studies to verify a priori the most important issues about the safety and capacity of a given design.

In this report the language SHIFT (Deshpande et al., 1997), developed in PATH to simulate the behavior of large scale hybrid systems, is employed to simulate regulation control laws for the hierarchical PATH AHS architecture in (Varaiya and Shladover, 1991; Varaiya, 1993).

The regulation layer control laws that were implemented are: lead, join, accelerate to enter and follow. These maneuvers are described in (Hsu et al., 1991; Hsu et al., 1993). The design of the lead, join and accelerate to enter control laws follow the approach presented in (Li et al., 1997) with two modifications. First, in order to match the controller that is implemented in the instrumented vehicles is PATH, an acceleration based controller is used instead of the jerk based controller used in (Li et al., 1997). The second change is in the procedure to estimate the acceleration of the vehicle ahead of the one under control. The problem is reformulated and the position of the vehicles is not longer used to estimate this acceleration. This greatly reduces the complexity of the observer used for the estimation of the acceleration and provides a faster response to sudden changes in acceleration. The changes in the stability analysis in (Li et al., 1997) are included in this report. The follow control law is implemented according to (Swaroop and Hedrick, 1996).

For the implementation, the regulation layer controller is divided into three automata. The first one is the supervisor automaton whose task is to communicate with the coordination layer to receive the order to execute a particular maneuver. When instructed the supervisor automaton will generate a second automaton, the maneuver controller automaton. This automaton is dependent on the particular maneuver being controlled and will be in charge of executing the feedback based control law. The last automaton, the desired velocity profile automaton, is designed to guarantee that the maneuver controller automaton will command accelerations that always keep the vehicle inside a dynamic safety region. The design of this region is explained in (Li et al., 1997).

Examples of simulation results are provided to illustrate the use of the code generated. In all cases the results agree with those encountered in previous implementations in Matlab or SmartPath. A summary of the code generated is also included in the appendix.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The verification of the control laws involved in an automated highway system (AHS) is a difficult but essential task. Each highway may contain several thousand automobiles at any one time, and the actions of any one car affect those in the immediately surrounding area. Because of the complexities of the system, computer simulations of the automated highway system are the most appropriate avenue for large scale verification of the control algorithms. Computer simulations are responsible for demonstrating the safety, efficiency, liveness, and fairness of the control laws. This paper discusses the implementation of the regulation layer, one controller layer of the PATH AHS architecture, within SHIFT. Written at the University of California by PATH researchers. SHIFT is a programming language specifically oriented toward handling dynamic networks of hybrid automata, which involve a combination of continuous and discrete behavior. Further information about SHIFT can be found in (Deshpande et al., 1997).

To discuss the structure of the regulation layer, it is first necessary to recall the function of the regulation layer within the PATH architecture (Varaiya and Shladover, 1991; Varaiya, 1993). As seen in Figure 1.1, directly above the regulation layer is the coordination layer. The coordination layer is responsible for issuing and receiving discrete commands from automobiles within a section of highway. Typical commands include instructions to become a leader of a platoon, join with another platoon, or become a follower (Hsu et al., 1991; Hsu et al., 1993). These commands are communicated to the regulation layer, which then determines the necessary controller response to execute the required maneuver. Within the SHIFT code, it is assumed that the acceleration of the automobile is controlled directly, as discussed later in the analysis chapter. Then the physical layer, which is a dynamic model of the actual automobile, attempts to match the actual automobile acceleration with the desired acceleration specified by the regulation layer. Upon completion of a maneuver, the regulation layer notifies the coordination layer that the maneuver is finished, thus completing the communication process between the regulation layer and the coordination layer. The regulation layer also initiates communication in the case of an unsafe situation.

This report is comprised of four chapters. The first is the analysis chapter, chapter 2, where the derivation and stability of the controller algorithm are discussed. Chapter 3 explains the implementation of the controller algorithm within SHIFT. Sample of simulations are given in chapter 4, followed by concluding remarks in chapter 5. The SHIFT and C codes are included in a companion disk.

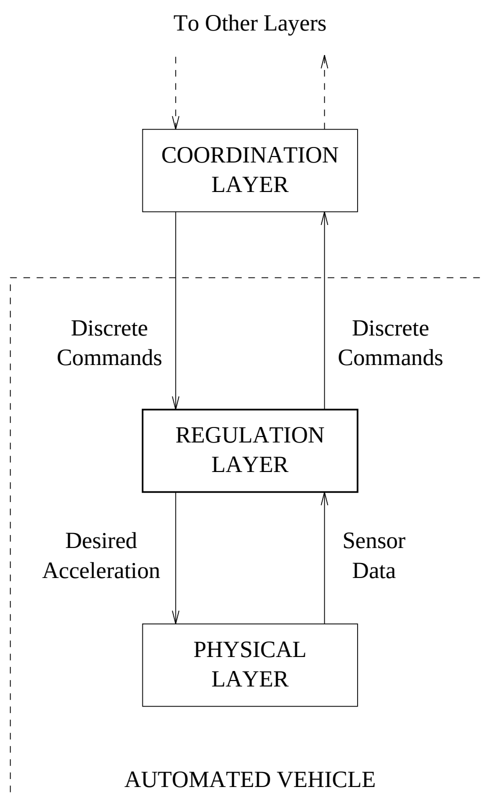


Figure 1.1: PATH architecture.

Chapter 2

Analysis

This chapter discusses the derivation of the controller for the regulation layer, the derivation of an observer for the lead car, and a proof of stability for the controller. A complete derivation of a previous controller for the regulation layer is given in (Li et al., 1997). The derivation that follows is very similar to this previous work, with the following important differences:

- The new controller assumes that the acceleration of the car can be controlled directly. This is a change from the previous controller, where the jerk was the parameter that could be controlled. This modification reflects the controller onboard the automated cars that are being designed and tested by PATH engineers.
- A full order observer is presented in Section 2.2. This observer is different from the observer introduced in (Li et al., 1997) because the position of the lead car is not included in the analysis. This reduces the complexity of the observer because the absolute position of the lead car does not need to be explicitly known.

These two major differences are highlighted in the following chapters, which are the theoretical basis for the regulation layer controller.

2.1 Controller Derivation

The objective of the regulation layer controller is to keep a vehicle traveling in the highway according with the conditions of relative velocity and relative spacing associated with a given maneuver. The next higher layer in the automated highway hierarchy, the coordination layer, issues commands that select the specific maneuver such as join, follow, or split. When there is a change of maneuver, the automated vehicle's regulation controller attempts to switch from the conditions associated with the present maneuver to the conditions associated with the new one in a quick and safe manner. To accomplish this task, the regulation layer controller tries to follow a desired velocity profile. Calculation of the desired velocity profile depends on three items: (1) the current maneuver, (2) the relative spacing between the trail car and a lead car, and (3) the velocity of a lead car.

Figure 2.1 shows the important geometrical parameters for the derivation. In this analysis, the trail car is assumed to be the automated car that is the target of the controller's action, and the lead

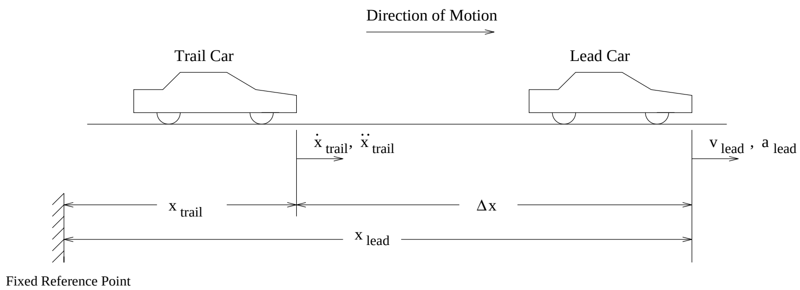


Figure 2.1: Geometry for controller derivation.

car is the car that is directly ahead of the trail car in the same highway lane. Variables associated with the trail car are denoted with the subscript *trail*; likewise, variables associated with the lead car are indicated with the subscript *lead*. Also, derivatives with respect to time are indicated by a dot above a given variable.

The basis for the controller algorithm is to minimize the error between the trail car's velocity, $\dot{x}_{lead}$, and the desired velocity, $v_d(\Delta x, v_{lead})$, which is a function of the relative spacing between the trail car and lead car Δx and v_{lead} is the lead car velocity. It is clear from Figure 2.1 that the relative spacing is

$$\Delta x = x_{lead} - x_{trail}.$$

If we define the velocity error *e* by

$$e := \dot{x}_{trail} - v_d(\Delta x, v_{lead}). \tag{2.1}$$

then taking the derivative of the error with respect to time yields

$$\dot{e} = \ddot{x}_{trail} - \begin{pmatrix} \dfrac{\partial v_d}{\partial \Delta x} & \dfrac{\partial v_d}{\partial v_{lead}} \end{pmatrix} \begin{pmatrix} v_{lead} - \dot{x}_{trail} \\ \dot{v}_{lead} \end{pmatrix}. \tag{2.2}$$

Assuming that the goal is to drive the error to zero exponentially, an appropriate expression for the closed loop error dynamics is

$$\dot{e} = -\lambda_1 e \tag{2.3}$$

Substituting this expression into Eq. (2.2) and solving for the trail car's acceleration results in the following equation:

$$\ddot{x}_{trail} := -\lambda_1 e + \begin{pmatrix} \dfrac{\partial v_d}{\partial \Delta x} & \dfrac{\partial v_d}{\partial v_{lead}} \end{pmatrix} \begin{pmatrix} v_{lead} - \dot{x}_{trail} \\ a_{lead} \end{pmatrix}, \tag{2.4}$$

This equation would drive the velocity error to zero exponentially; unfortunately, the lead car's acceleration is not known exactly. If instead, it is assumed that an estimate of the lead car's acceleration is known[1], then the following is an acceptable control law for the trail car acceleration:

$$\ddot{x}_{trail} := \Gamma(\Delta x, v_{lead}, \dot{x}_{trail}, \hat{a}_{lead}) = -\lambda_1 e + \begin{pmatrix} \dfrac{\partial v_d}{\partial \Delta x} & \dfrac{\partial v_d}{\partial v_{lead}} \end{pmatrix} \begin{pmatrix} v_{lead} - \dot{x}_{trail} \\ \hat{a}_{lead} \end{pmatrix}, \tag{2.5}$$

[1]This approach is known as back-stepping (**?**).

Note that the last term involves the estimate of the lead car's acceleration, $\hat{a}_{lead}$, instead of the true acceleration. In the next section, an observer for the lead car's acceleration will be introduced.

The dynamics for the velocity error *e* are easily shown to be

$$\dot{e} = -\lambda_1 e - \frac{\partial v_d}{\partial v_{lead}} \tilde{a}_{lead}, \tag{2.6}$$

where $\tilde{a}_{lead}$ is the estimation error for the lead car acceleration,

$$\tilde{a}_{lead} := a_{lead} - \hat{a}_{lead}. \tag{2.7}$$

Thus so long as the estimation error remains small, the error approaches zero approximately exponentially. The stability of this solution will be investigated further in Section 2.3.

2.2 Observer for Lead Car Motion

An estimate for the lead car acceleration is necessary for the proposed controller. Assuming that the lead car velocity v_{lead} is known from sensor data, the following is the derivation of a full order observer for the lead car acceleration.

The lead car dynamics are given by

$$\frac{d^3}{dt^3} x_{lead} = \dot{a}_{lead}(t), \tag{2.8}$$

where a_{lead} is the acceleration input to the lead car. Written in state space form where the state matrix is $\mathbf{x} = \begin{pmatrix} v_{lead} & a_{lead} \end{pmatrix}^T$ the above equation becomes

$$\dot{\mathbf{x}} = A\mathbf{x} + B\dot{a}_{lead}(t). \tag{2.9}$$

where

$$A := \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \qquad B := \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{2.10}$$

Also, since the only variable assumed to be known by the trail car is the lead car velocity, the equation for the sensor output is

$$y = C\mathbf{x}, \tag{2.11}$$

where

$$C := \begin{pmatrix} 1 & 0 \end{pmatrix}. \tag{2.12}$$

A full order observer for the state $\mathbf{x}$ is

$$\dot{\hat{\mathbf{x}}} = A\hat{\mathbf{x}} + L(y - C\hat{\mathbf{x}}) + \mathbf{q}(t), \tag{2.13}$$

where $\hat{\mathbf{x}}$ is the state estimate, L is the observer gain matrix, and $\mathbf{q}(t)$ is a tuning function to be determined in the stability analysis. This is a standard full order observer with the addition of one term, $\mathbf{q}(t)$, which accounts for the nonlinearities inherent in the system. By subtracting Eq. (2.13) from Eq. (2.9), the dynamics for the state estimator error, $\tilde{\mathbf{x}} = \mathbf{x} - \hat{\mathbf{x}}$, is found to be

$$\dot{\tilde{\mathbf{x}}} = (A - LC)\tilde{\mathbf{x}} + B\dot{a}_{lead}(t) - \mathbf{q}(t). \tag{2.14}$$

If both $a_{lead}(t)$ and $\mathbf{q}(t)$ remain bounded, the state estimates will approach the actual states so long as the both of the eigenvalues of $A_F \equiv (A - LC)$ have negative real components. A simple calculation shows that this occurs when the components of *L* are both positive.

2.3 Stability Analysis

Since the controller involves the estimate of the lead car acceleration, which is calculated by the full order observer, the dynamic responses of the controller and the observer are coupled. Consider the following candidate for a Lyapunov function:

$$V(e, \tilde{x}_{lead}) = \frac{1}{2}Qe^2 + \frac{1}{2}\gamma\tilde{\mathbf{x}}_{lead}^T P\tilde{\mathbf{x}}_{lead}, \tag{2.15}$$

where *Q* and γ are both positive constants and *P* is a positive definite matrix. This candidate function includes terms involving both the controller error e and the observer error $\tilde{x}$. The most difficult part of this analysis is the choice of *P* such that it satisfies the relationship

$$A_F{}^T P + PA_F = -2C \tag{2.16}$$

where C is also a positive definite matrix. This relationship is necessary for proving that *V* is indeed a Lyapunov function. Prior to discussing *P* further, several other derivations are required.

Consider the real decomposition of A_F,

$$A_F = T\Lambda T^{-1}, \tag{2.17}$$

where *T* is real and invertible and the diagonal of Λ contains the real parts of the eigenvalues of A_F. Λ can be further decomposed into two components,

$$\Lambda = \Lambda_1 + \Lambda_2, \tag{2.18}$$

where Λ_1 is symmetric (i.e. $\Lambda_1 = \Lambda_1{}^T$) and Λ_2 is skew-symmetric (i.e. $\Lambda_2 = -\Lambda_2{}^T$).

One possibility for the choice of *P* is

$$P = (T^{-1})^T T^{-1}, \tag{2.19}$$

where *T* is the matrix introduced above. In this case, we find that the matrix *C* that satisfies Eq. (2.16) is

$$C = -(T^{-1})^T \Lambda_1 T^{-1}, \tag{2.20}$$

which is positive definite if every diagonal element of the matrix Λ_1 is negative. Since the diagonal elements of Λ_1 are the real parts of the eigenvalues of A_F, *C* is positive definite if the full-order observer state matrix A_F has stable eigenvalues. Thus choosing P according to Eq. (2.19) guarantees a positive definite solution to C in Eq. (2.16) given an appropriate full order observer. This fact is essential in completing the next portion of the analysis.

For the function V to be an acceptable Lyapunov function, its derivative with respect to time, $\dot{V}$, must be negative definite. Taking the time derivative of Eq. (2.15), and using the relationship between P and C given in Eq. (2.16), yields

$$\dot{V} = -\lambda_1 Q e^2 - \gamma \tilde{\mathbf{x}}_{lead}^T C \tilde{\mathbf{x}}_{lead} + \gamma \tilde{\mathbf{x}}_{lead}^T P B \dot{a}_{lead}(t) - Qe \frac{\partial v_d}{\partial v_{lead}} \begin{pmatrix} 0 & 1 \end{pmatrix} \tilde{\mathbf{x}}_{lead} - \gamma \mathbf{q}^T P \tilde{\mathbf{x}}_{lead}. \tag{2.21}$$

Because the fourth term involves $\frac{\partial v_d}{\partial v_{lead}}$, a nonlinear function of Δx and v_{lead}, it is convenient to choose the tuning function $\mathbf{q}$ such that the fourth term is eliminated from the equation. Thus, an appropriate choice for $\mathbf{q}$ is

$$\mathbf{q} = \frac{Qe}{\gamma} \frac{\partial v_d}{\partial v_{lead}} P^{-1} \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \tag{2.22}$$

Substituting this expression into the equation for $\dot{V}$ yields

$$\dot{V} = -\lambda_1 Q e^2 - \gamma \tilde{\mathbf{x}}_{lead}^T C \tilde{\mathbf{x}}_{lead} + \gamma \tilde{\mathbf{x}}_{lead}^T P B \dot{a}_{lead}(t). \tag{2.23}$$

A derivation that closely follows the methods presented in (Li et al., 1997) shows that for any initial condition $e(0)$, and for any $\epsilon > 0$, there is a time T_1 such that if $t \geq T_1$ then

$$\sqrt{\frac{Q}{2}} |e(t)| \leq V^{\frac{1}{2}}(t) \leq \frac{j_{max}}{\zeta} \sqrt{\frac{\delta}{\gamma}} (1 + \epsilon), \tag{2.24}$$

where

$$|\dot{u}| \leq j_{max},$$

and

$$\delta := 2 P_{\cdot 2}^T P^{-1} P_{\cdot 2}\,,$$

where $P_{\cdot 2}$ is the second column of P.

Therefore, $|e(t)| \leq \frac{j_{max}}{\zeta} \sqrt{\frac{\delta \rho}{\gamma}} (1 + \epsilon)$ after a long enough time.

Chapter 3

Regulation Layer Structure in SHIFT

The organization of the regulation layer within SHIFT is shown in Figure 3.1 for the sample case where the automobile is executing a join maneuver. At the top of the regulation layer structure is the supervisor automaton. This automaton is the primary interface between the coordination layer and the regulation layer (Eskafi, 1996). In addition this automaton is responsible for creating the appropriate maneuver controllers. Just below the supervisor automaton is the maneuver controller. This automaton is specific to the automobile's current maneuver and is the primary location for the calculation of the desired acceleration, the continuous-time output of the regulation layer controller. There are other automata that support the maneuver controller. these are the region automaton and the acceleration bounds automaton.

The following sections discuss the automata with respect to their actual implementation in SHIFT. In order to establish a common vocabulary for these sections, several definitions are required. When discussing the code, *type* refers to the basic programming structure for automata. Types establish a general structure for an automata; each instance of the same structure is denoted as a *component*. Within each type, it is necessary to establish the automaton's continuous and discrete properties. Statements that define the continuous-time progression of variables are *flows*, whereas discrete states progress through discrete *transitions*.

3.1 Supervisor Automaton

The supervisor automaton is the top most automaton in the structure of the regulation layer. Operating almost entirely in the discrete domain, this automaton is responsible for the following tasks:

- The supervisor automaton communicates with the coordination layer. This involves receiving and transmitting discrete commands. In SHIFT these commands are actually implemented as synchronous events, which force separate automata to transition simultaneously.

- This automaton is responsible for the creation of the maneuver controller automaton. As an example, when the coordination layer requests that the car execute a join maneuver, the regulation layer creates a maneuver controller called *JoinController* specific to the join maneuver.

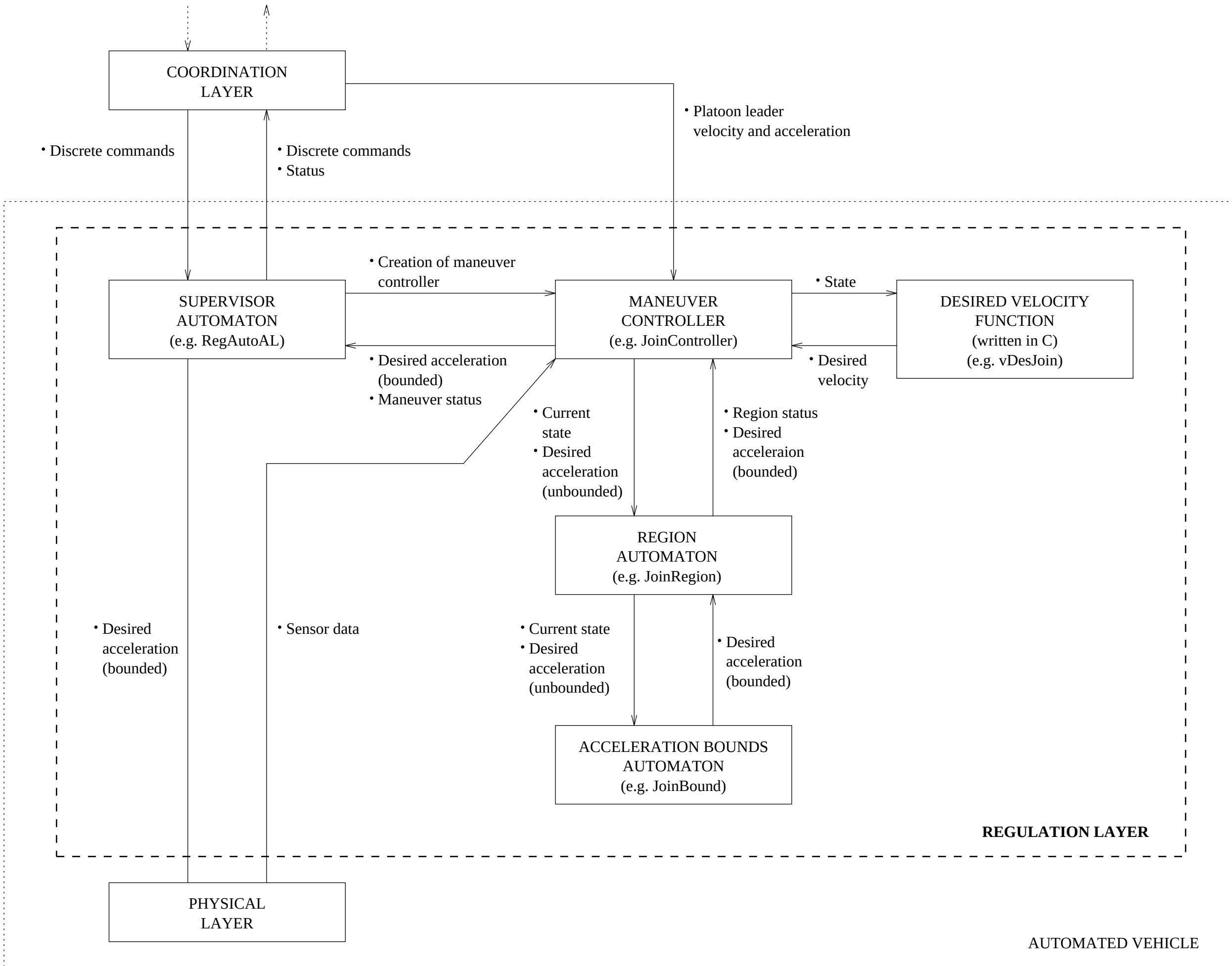


Figure 3.1: Structure of the regulation layer in SHIFT.

- Since the type of maneuver controller changes over the course of a simulation and would thus be a difficult point of contact between the physical layer and the regulation layer, the supervisor acts as an intermediate link in the process of communicating the desired acceleration to the physical layer.

Supervisor Automaton: Flow

The only flow statements within the supervisor automaton are ones that create a local copy of the desired acceleration, which is originally calculated in a maneuver controller.

Supervisor Automaton: Discrete Transitions

The discrete finite state machine associated with the supervisor automaton enables (1) the coordination layer to specify a new maneuver to execute, (2) the coordination layer to abort a maneuver, and (3) the maneuver controller to report the status of a current maneuver. A small portion of the state diagram is shown as an example in Figure 3.2. Illustrating the discrete transitions associated with the finite state *JOIN*, this diagram shows that the transitions occur simultaneously with transitions in the coordination layer and in the maneuver controller automaton.

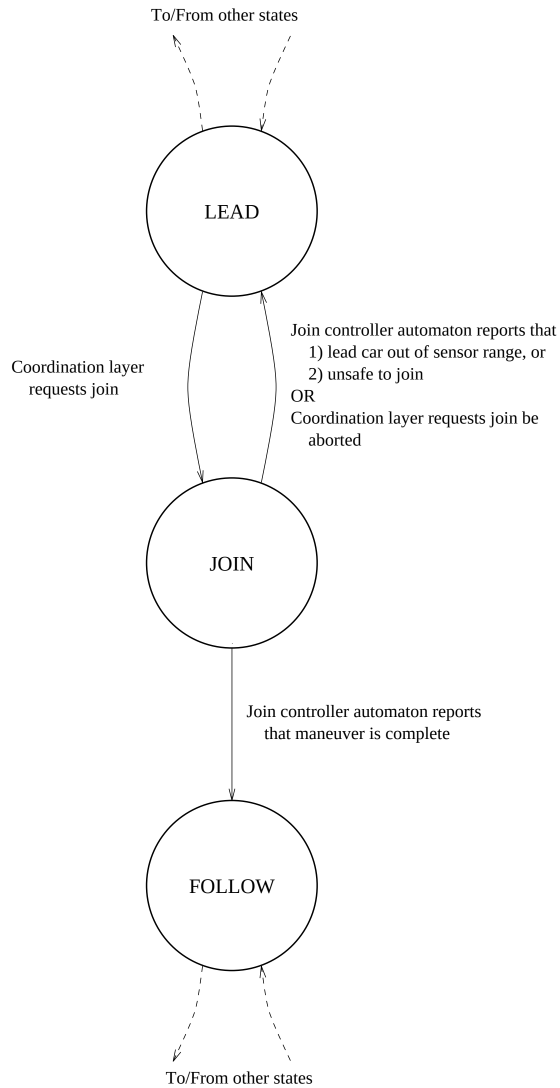


Figure 3.2: Partial state diagram of the supervisor automaton.

When a discrete transition occurs within the supervisor automaton, a new maneuver controller is created and initial connections between components are defined. The list given below summarizes these actions.

1. Create a new maneuver controller automaton.

2. Establish a connection between the new maneuver controller and the supervisor automaton.

3. Establish a connection between the maneuver controller and the automated vehicle automaton, which acts as a top-level automaton for the entire car, including both the physical layer and the regulation layer.

4. Establish a connection between the maneuver controller and the coordination layer automaton.

This concludes the discussion of the supervisor automaton. The next section continues with a discussion of the next structural component, the maneuver controller.

3.2 Maneuver Controller

The maneuver controller is the automaton that is responsible for determining the controller action for a given maneuver. Thus, the controller derived in the analysis is located within this automaton. In addition to the continuous calculation of the controller output, there are discrete transitions within the maneuver controller that indicate the current status of the maneuver. The primary functions of the maneuver controller are

- To calculate the controller response, using the results of the analysis chapter as a model for the calculations.
- To create and then communicate with a region automaton, which (1) monitors the current state of the trail and lead car and (2) places bounds on the desired acceleration.

Maneuver Controller: Flow

As stated above, the continuous time portion of the maneuver controller calculates the desired acceleration for the trail car using the previously derived controller law. The procedure for this calculation is outlined below.

1. Obtain sensor data from physical layer.
2. Call a function written in C to calculate the desired velocity. Use approximation techniques to calculate the first and second partial derivatives of the desired velocity with respect to the relative spacing, Δx, and the lead car velocity, v_{lead}. As an example of the approximation method, consider the following approximation for the first partial derivative of the desired velocity with respect to Δx:

$$\frac{\partial v_d}{\partial \Delta x}(\Delta x, v_{lead}) \approx \frac{v_d(\Delta x + d\Delta x, v_{lead}) - v_d(\Delta x - d\Delta x, v_{lead})}{2d\Delta x}, \tag{3.1}$$

   where $d\Delta x$ is a small finite length. Clearly, if $d\Delta x$ were allowed to approach zero, then this would in fact be the exact partial derivative. Although the approximate calculation in Eq. (3.1) requires two additional calls to the C-function for calculation of the desired velocity, this technique eliminates the need for an explicit solution to the partial derivative, which is quite cumbersome. The approximation also avoids the potential problem that the desired velocity profile may not be differentiable at every point within the domain.

3. Use the full order observer developed in Section 2.2 to estimate the lead car acceleration and jerk.

4. Calculate the unbounded desired acceleration $\ddot{x}_{trail}$ using Eq. (2.5). Then calculate an estimate of the time derivative of this value using the following approximation:

$$\begin{aligned}\hat{j}_d = \quad & \Delta v \left(\lambda_1 \frac{\partial v_d}{\partial \Delta x} + \frac{\partial^2 v_d}{\partial \Delta x^2} \Delta v + \frac{\partial^2 v_d}{\partial v_{lead} \partial \Delta x} \right) \\ & + \hat{a}_{lead} \left(\lambda_1 \frac{\partial v_d}{\partial v_{lead}} + \frac{\partial v_d}{\partial \Delta x} + \frac{\partial^2 v_d}{\partial \Delta x \partial v_{lead}} \Delta x + \frac{\partial^2 v_d}{\partial \Delta x \partial v_{lead}} \hat{a} \right) \\ & + a_{trail} \left(-\lambda_1 - \frac{\partial v_d}{\partial \Delta x} \right) \\ & + \dot{\hat{a}} \frac{\partial v_d}{\partial v_{lead}},\end{aligned}$$

   where

$$\Delta v = v_{lead} - v_{trail}.$$

   This equation is arrived at by taking the time derivative of Eq. (2.5) and substituting in estimated values where appropriate.

5. Finally, use the region automaton to place limits on the desired acceleration.

Maneuver Controller: Discrete Transitions

In addition to the implementation of the controller law in the continuous time domain, the maneuver controller has discrete states that indicate the current status of a maneuver. The general structure is similar from one maneuver to another. An example is presented for the join maneuver in Figure 3.3.

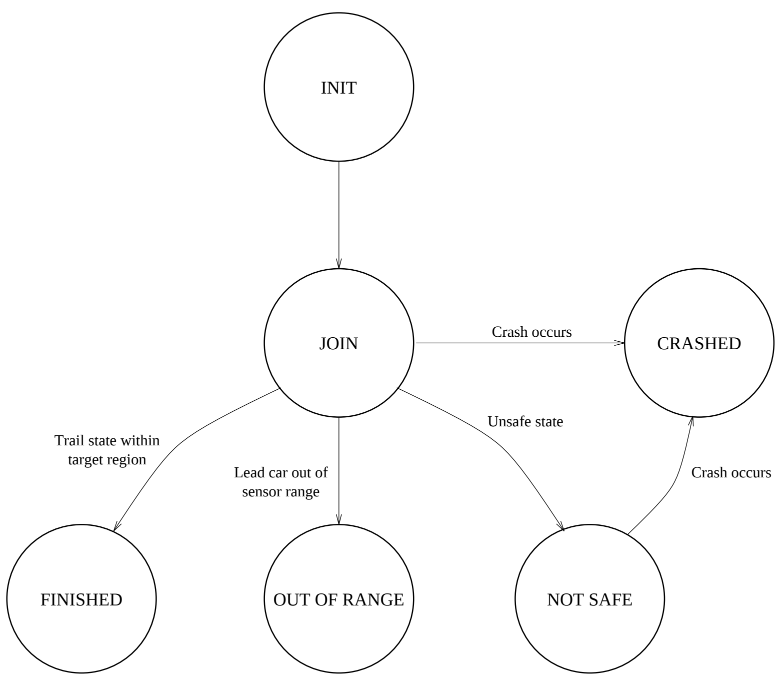


Figure 3.3: Maneuver controller state diagram for merging.

3.3 Desired Velocity Profile

The desired velocity profile is calculated in a function that is called from the maneuver controller. Because of the manner in which SHIFT evaluates numerical statements, certain calculations are much more easily written in C. For each maneuver, the function receives the relative spacing Δx and the lead car velocity v_{lead} as arguments, and returns the value of the desired velocity v_d.

3.4 Region Automaton

The purpose of the region automaton is twofold: (1) to monitor the car's current location within the state space and (2) to place limits on the desired acceleration which was calculated in the maneuver controller. The procedure for accomplishing these goals is to use a finite state machine to indicate the current region of the state space, and then dependent on the region, calculate the limits for the acceleration.

A basic diagram of the regions of the state space is shown in Figure 3.4. As is evident in the figure, the state space is divided into six regions: *NORMAL*, *TOO FAR NO COMFORT*, *BRAKE*, *UNSAFE*, *CRASHED*. See (Li et al., 1997) for a full explanation of the regions. Discrete transitions in the region automaton correspond to movements of the state from one region to another.

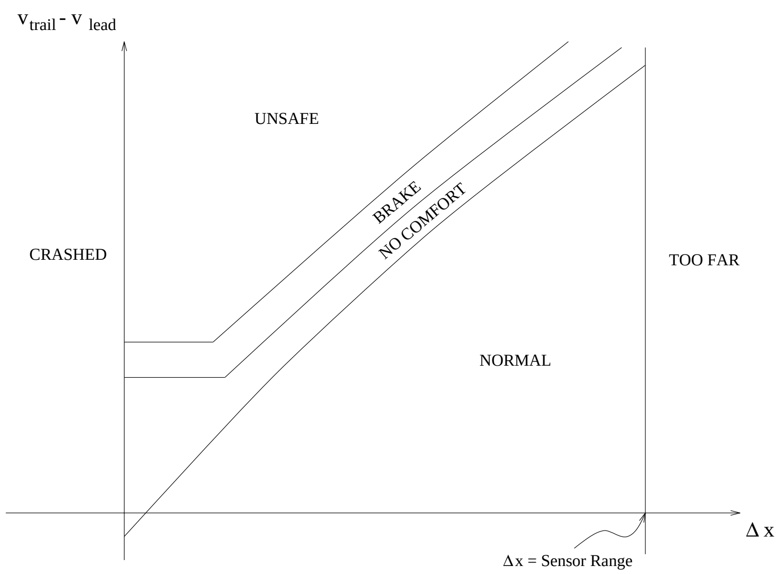


Figure 3.4: Diagram of regions (not to scale).

The continuous time flow within the region automaton establishes the bounds for the desired acceleration. These bounds depend on the current discrete state and are summarized in Table 3.1

3.5 Acceleration Bounds

This automaton is a complicated combination of continuous time flows and discrete state transitions that ensures that the velocity, acceleration, and jerk of the car are maintained within set

Region	Acceleration Characteristics
NORMAL	Calculated by Eq. (2.5) with comfortable limits
NO COMFORT	Calculated by Eq. (2.5) with modified limits (allows deceleration and acceleration outside the comfortable limits)
BRAKE	Decelerate as quickly as possible
UNSAFE	Decelerate as quickly as possible

Table 3.1: Bounds for acceleration.

limits. For each maneuver, two instances, or components, of this type are created. One is used when the state is within the *NORMAL* or *TOO FAR* regions, where bounds are set assuming that the motion should remain within a comfortable range. The other instance is used when the state is within the *NO COMFORT* region, when the car is able to execute trajectories that are outside of the comfortable limits.

Chapter 4

Simulation

This chapter discusses an example simulation from the SHIFT code, viewed using a graphical user interface called *TkShift*. Of the three cars modeled within the simulation, only the trail car (*Car 1*) is an automated car with a complete regulation layer controller. For the other two cars, *Car 2* and *Car 3,* the velocity is held constant for all time. The details of simulation setup are given below.

- The constant parameters such as vehicle and highway characteristics are listed in Appendix A. In addition, parameter values that are specific to individual maneuvers are given in Appendix B.

- The trail car, *Car 1,* begins with initial conditions $x_{trail}(0) = 0m$ and $\dot{x}_{trail}(0) = 20m/s$ and is initially in a lead maneuver. At $t = 10s$, the coordination layer requests that the trail car join with *Car 2*.

- *Car 2* is the previous car in front of *Car 1*. *Car 2* begins with an initial position of $x_{lead}(0) = 25m$ and maintains a constant velocity of $20m/s$ throughout the simulation.

- *Car 3*, the platoon leader for the platoon that includes *Car 2*, has an initial position of $x_{platoon}(0) = 50m$ and also maintains a constant velocity of $20m/s$ during the entire simulation.

Figure 4.1 shows the continuous-time results of the simulation. These graphs encompass three different maneuvers: the lead maneuver from 0 to 10 seconds, the join maneuver from 10 to 28 seconds, and the follow maneuver from 28 to 30 seconds. The transition from the join maneuver to the follow maneuver occurs automatically when the *JoinController* determines that trail car has completed joining with *Car 2*. It is also illustrative to view the discrete transitions of the hybrid automata; however, due to the large number of transitions during the course of one simulation, this paper does not discuss these transitions further.

During the first ten seconds of the results in Figure 4.1, the lead maneuver is executed. This maneuver successfully increases the relative spacing between *Car 1* and *Car 2* to $35m$, which is the desired interplatoon spacing. Figure 4.2 shows that the actual velocity approaches the desired velocity as time progresses.

At 10 seconds, the trail car begins joining with *Car 2*. At this time the trail car velocity increases in order to reduce the relative spacing between between itself and *Car 2*. Comparing the time response in Figure 4.1 and the relative velocity plot in Figure 4.3, it is possible to explain the results. First the trail car speeds up to reduce the error between the actual velocity and the desired velocity. At about 14 seconds, the trail car obtains its maximum velocity and then begins to decelerate, following the desired velocity profile. The trail car continues to approach the previous car, until at about 24 seconds the trail car suddenly decelerates. This deceleration is the result of the overshoot seen in Figure 4.3 where the trail car attempts to quickly decelerate to match the speed of the previous car. At 28.0 seconds, the maneuver controller determines that the trail car has completed the join and is ready for the follow maneuver.

Unlike the two previous maneuvers, the follower law does not attempt to explicitly follow a desired velocity profile. Instead, the controller minimizes the errors associated with (1) the spacing between the trail car and the previous car, (2) the relative velocity of these two cars, and (3) the relative velocity between the trail car and the platoon leader. Additionally, the controller accounts for the accelerations of the previous car and the platoon leader. Figure 4.4 shows the time response of the relative spacing between the trail car, *Car 1*, and the previous car, *Car 2*, as well as the time response for their relative velocity. The controller quickly adjusts the relative spacing toward the desired intraplatoon spacing of $2m$ and zero relative velocity.

This example shows the action of the three maneuvers that are currently incorporated into the regulation layer in SHIFT. In addition, other simulations have shown that the general controller algorithm successfully reduce the velocity error over time and that the full-order observer is a good estimator for the acceleration of the lead car. Simulations have included cases where the lead car either has an oscillatory velocity or stops suddenly.

One item which has not been previously discussed in this paper is the integration of the regulation layer code with the physical layer and the coordination layer. Presently, the regulation layer and the physical layer are fully integrated, allowing the regulation layer to control the acceleration of a complex vehicle model within the physical layer. Further work is required for the complete integration of the regulation layer and the coordination layer.

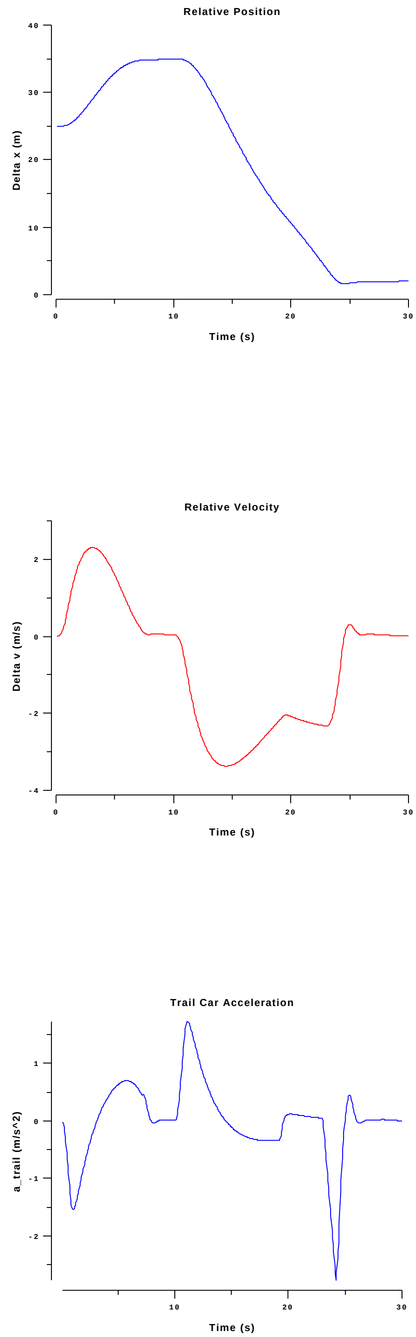


Figure 4.1: Simulation results.

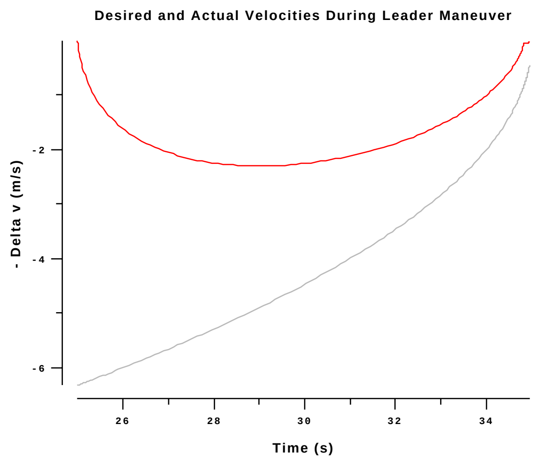


Figure 4.2: Desired and actual velocities for lead maneuver. As time increases, the velocities approach the upper right corner of the plot.

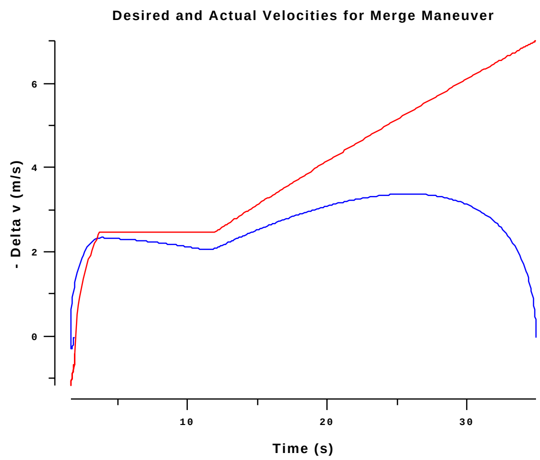
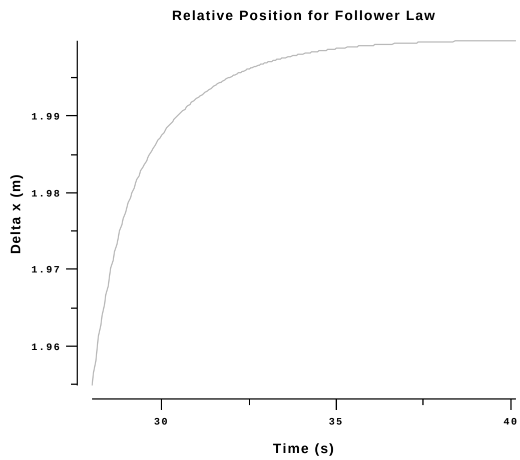


Figure 4.3: Desired and actual velocities for join maneuver. As time increases, the velocities approach the lower left corner of the plot.



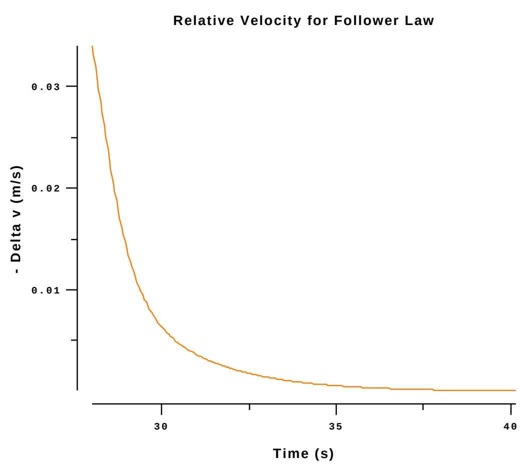


Figure 4.4: Follow maneuver time response.

Chapter 5

Concluding Remarks

This paper discussed the implementation of the regulation layer within SHIFT, a software language written specifically to handle the simulation of hybrid automata. The analysis chapter formulated a controller which explicitly determined the automated car's acceleration. Within this controller, it was necessary to include a full-order observer to estimate the previous car's acceleration. The stability of the controller algorithm was proven through the use of a Lyapunov function in the final section of the analysis. Next, the structure of the regulation layer within SHIFT was explained. For each hybrid automaton, key aspects were identified, such as the automaton's objectives, its continuous-time flow, and the accompanying finite state machine. As an example of the code's operation, a simulation which included the execution of three maneuvers was presented.

The current implementation of the regulation layer in SHIFT serves as a strong foundation for future work. Several items that need to be addressed include (1) the addition of other maneuvers, (2) continuation of simulation testing, and finally (3) integration of the regulation layer with the coordination layer. These and other advancements will facilitate testing of the automated highway architecture proposed by PATH.

Bibliography

Deshpande, A. R., Göllü, A., and Semenzato, L. (1997). SHIFT *Reference Manual*. California PATH.

Eskafi, F. (1996). *Modeling and Simulation of the Automated Highway Systems*. PhD thesis, Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

Hsu, A., Eskafi, F., Sachs, S., and Varaiya, P. (1991). The design of platoon maneuver protocols for IVHS. Technical Report UCB-ITS-PRR-91-6, University of California, Berkeley.

Hsu, A., Eskafi, F., Sachs, S., and Varaiya, P. (1993). Protocol design for an automated highway system. *Discrete Event Dynamic Systems*, 2:183–206.

Krstic, M., Kanellakopoulos, I., and Kokotovic, P. (1995). *Nonlinear and Adaptive Control Design*. John Wiley and Sons, Inc.

Li, P., Alvarez, L., and Horowitz, R. (1997). AVHS safe control laws for platoon leaders. To appear in IEEE Transactions on Control Systems Technology.

Swaroop, D. and Hedrick, J. K. (1996). String stability of interconnected systems. *IEEE Transactions on Automatic Control*, AC-41(3):349–357.

Varaiya, P. (1993). Smart cars on smart roads: problems of control. *IEEE Transactions on Automatic Control*, AC-38(2):195–207.

Varaiya, P. and Shladover, S. E. (1991). Sketch of an IVHS systems architecture. Technical Report UCB-ITS-PRR-91-3, Institute of Transportation Studies, University of California, Berkeley.

Appendix A

Summary of Global Parameters

Parameter	Description	Value
$\Delta x_{SensorRange}$	Sensor range distance	$60.0m$
v_{max}	Maximum vehicle velocity	$40.0\frac{m}{s}$
a_{max}	Maximum vehicle acceleration	$2.5\frac{m}{s^2}$
a_{min}	Minimum vehicle acceleration	$-5.0\frac{m}{s^2}$
j_{max}	Maximum vehicle jerk	$2.5\frac{m}{s^3}$
j_{min}	Minimum vehicle jerk	$-50\frac{m}{s^3}$
v_{link}	Optimum velocity - set by link layer	$25.0\frac{m}{s}$

Table A.1: Vehicle Parameters

Parameter	Description	Value
$a_{com,max}$	Maximum comfortable acceleration	$2.0\frac{m}{s^2}$
$a_{com,min}$	Minimum comfortable acceleration	$-2.0\frac{m}{s^2}$
$j_{com,max}$	Maximum comfortable jerk	$2.5\frac{m}{s^3}$
$j_{com,min}$	Minimum comfortable jerk	$-2.5\frac{m}{s^3}$
Δv_{allow}	Maximum safe relative velocity for crash	$3.0\frac{m}{s}$

Table A.2: Comfort and Safety Parameters

Parameter	Description	Value
v_{fast}	Maximum velocity on highway	$35.0\frac{m}{s}$
v_{slow}	Minimum velocity on highway	$10.0\frac{m}{s}$

Table A.3: Highway Parameters

Parameter	Description	Value
$d\Delta x$	Position step	$0.25m$
$d\Delta v$	Velocity step	$0.1\frac{m}{s}$

Table A.4: Derivative Estimation Parameters

Appendix B

Maneuver Details

B.1 Lead Maneuver

The purpose of the leader maneuver is to establish a car as the leader of a platoon. In this position, the optimum speed of the car is v_{link}, a velocity specified by the link layer of the automated highway system. However, the leader law also must maintain a desired distance, Δx_{leader}, between the car and a previous platoon. (Li et al., 1997) suggest one possible desired velocity profile to satisfy these criteria. These velocity profile has a velocity discontinuity at Δx_{leader}. It is possible to smooth this discontinuity by including a term depending on a comfort acceleration. Therefore, a new velocity profile is adopted herein. Let the velocity profile for a comfortable acceleration with final conditions $\dot{x}_{trail} = v_{lead}$ and $\Delta x = \Delta x_{leader}$ be denoted by $v_{com,accel}$. Likewise let the velocity profile for a comfortable deceleration with the same final conditions be denoted by $v_{com,decel}$. These two quantities are

$$v_{com,accel} = \max\left(0, v_{lead} - \sqrt{2a_{com,min}(\Delta x - \Delta x_{leader})}\right) \tag{B.1}$$

and

$$v_{com,decel} = \min\left(v_{fast}, v_{lead} + \sqrt{2a_{com,max}(\Delta x - \Delta x_{leader})}\right) \tag{B.2}$$

where $a_{com,min}$ and $a_{com,max}$ are the minimum and maximum comfortable accelerations, respectively. Then the desired velocity profile is

$$v_d(v_{lead}, \Delta x) = \begin{cases} \min\left(v_{com,accel}, v_{link}\right) & \text{if } \Delta x < \Delta x_{leader} \\ \min\left(v_{com,decel}, v_{link}\right) & \text{if } \Delta x_{leader} \leq \Delta x < \Delta x_{SensorRange} \\ v_{link} & \text{if } \Delta x \geq \Delta x_{SensorRange} \end{cases} . \tag{B.3}$$

B.2 Join Maneuver

While maintaining safety and passenger comfort, the join maneuver attempts to join an individual car to the end of an existing platoon in minimal time. The SHIFT code adopts a desired velocity

Parameter	Description	Value
λ_1	Controller gain	0.3
γ	Tuning function gain	1.0
Q	Tuning function gain	0.1
L	Observer gain	$\begin{pmatrix} 19 & 90 \end{pmatrix}^T$
Δx_{leader}	Minimum spacing for leader	$35.0m$

Table B.1: Lead Maneuver Parameters

profile proposed by (Li et al., 1997). The following is a summary of their calculations. First, the maximum safe velocity curve v_safe of the trail platoon for a given v_lead and Δx is

$$v_{safe}(v_{lead}, \Delta x) = \\ \max \begin{cases} -(a_{max} - a_{min})d + \sqrt{-2a_{min}\Delta x + v_{lead}^2 + v_{allow}^2 - a_{min}(a_{max} - a_{min})d^2}\ , \\ -(a_{max} - a_{min})d + v_{lead} + v_{allow}\ , \end{cases} \tag{B.4}$$

where a_{max} and a_{min} are the maximum and minimum vehicle accelerations, respectively, d is the delay for maximum deceleration to be achieved when a maximum braking command is issued, and v_{allow} is the maximum relative speed between the lead and trail platoons at which an impact can occur safely. For the join to be completed in the minimum possible time, while still maintaining comfortable accelerations and jerks, the appropriate velocity is

$$v_{min}(v_{lead}, \Delta x) = \min \begin{cases} v_{lead} + \sqrt{2a_{com}(\Delta x - \Delta x_{join})}\ , \\ v_{fast}\ \ , \end{cases} \tag{B.5}$$

where a_{com} is the magnitude of the comfort acceleration and deceleration, Δx_{join} is the desired intraplatoon distance, and v_{fast} is the maximum recommended velocity on the highway. In order to satisfy both safety and the objective of completing the maneuver in minimum time, the desired velocity profile should be

$$v_d(v_{lead}, \Delta x) = \min(v_{min}, v_{safe})\ .$$

Parameter	Description	Value
λ_1	Controller gain	0.3
γ	Tuning function gain	1.0
Q	Tuning function gain	0.1
L	Observer gain	$\begin{pmatrix} 19 & 90 \end{pmatrix}^T$
Δx_{join}	Desired following distance	$2.0m$
$d\Delta x_{join}$	Allowable error in join finish position	$0.05m$
$d\Delta v_{join}$	Allowable error in join finish velocity	$0.1\frac{m}{s}$

Table B.2: Join Maneuver Parameters

B.3 Accelerate To Enter Maneuver

The process of entering the automated lanes from the manual lanes involves several steps. The SHIFT code includes one portion of this procedure: while in a transition lane, the car accelerates from a complete stop to the speed of the traffic in the automated lane, simultaneously a desired longitudinal spacing is created between the accelerating car and a designated car within the automated lane. Safety must also be maintained for the automated car at all times.

The objective for the accelerating car is to match the velocity of the car in the automated lane, which is designated as v_{AL}, while also obtaining a desired relative spacing with the same car. The spacing, which places the car in the transition lane in a position to change lanes and be a follower, is denoted as Δx_{accel}. These conditions are referred to as the final conditions of the maneuver. A further goal of the maneuver is to reach these final conditions before the accelerating car passes out of the designated acceleration lane. As in the lead maneuver and the join maneuver, the accelerating car attempts to minimize the error between the car's velocity and a desired velocity, which is determined according to a specific profile. However, in this maneuver, the desired velocity depends on the relative spacing between the accelerating car and a designated car in the automated lane and the velocity of the car in the automated lane. These quantities are denoted as Δx_{AL} and v_{AL}, respectively. In order to easily specify the desired velocity profile, two velocity functions are introduced. First, a velocity function to be used when $\Delta x_{AL} < \Delta x_{accel}$ is

$$v_{accel} = \max\left(0, v_{AL} - \sqrt{2ra_{com,min}(\Delta x_{AL} - \Delta x_{accel})}\right) \tag{B.6}$$

where r is a constant less than one. This constant, r, is included so that the acceleration associated with Eq. (B.6) is slightly less than the comfortable acceleration $a_{com,min}$. Likewise, when $\Delta x_{AL} \geq \Delta x_{accel}$ then the following velocity applies:

$$v_{decel} = \min\left(v_{fast}, v_{AL} + \sqrt{2ra_{com,max}(\Delta x_{AL} - \Delta x_{accel})}\right). \tag{B.7}$$

As before, the constant r is introduced so that the deceleration is slightly less than the comfortable deceleration in magnitude. Using these velocity functions, the desired velocity profile is defined to be

$$v_d(v_{AL}, \Delta x_{AL}) = \begin{cases} \min(v_{accel}, v_{link}) & \text{if } \Delta x_{AL} < \Delta x_{accel} \\ \min(v_{decel}, v_{link}) & \text{if } \Delta x_{AL} \geq \Delta x_{accel} \end{cases}. \tag{B.8}$$

Safety for the accelerating car is determined with respect to a previous car in the transition lane. If there is a previous car within sensor distance, then safety is determined by a region automaton as described in Section 3.4. To ensure that safety of the accelerating car is maintained, if the *NO COMFORT* region is entered, then the current maneuver is aborted and the car begins to execute another maneuver as directed by the coordination layer.

B.4 Follow Maneuver

The follower law is different than the controller law for the other maneuvers because vehicle to vehicle, or string-stability, must be satisfied within a platoon. (Swaroop and Hedrick, 1996) proposed a follower control for the vehicles within an arbitrarily large platoon. The solved the problem

Parameter	Description	Value
λ_1	Controller gain	0.3
γ	Tuning function gain	1.0
Q	Tuning function gain	0.1
L	Observer gain	$\begin{pmatrix} 19 & 90 \end{pmatrix}^T$
Δx_{accel}	Desired spacing	$2.0m$
r	Scaling constant	0.9

Table B.3: Accelerate To Enter Maneuver Parameters

by relaying the platoon leader velocity $v_{platoon}$ and acceleration $a_{platoon}$ to the trail car. By using this additional information, the error spacing between individual cars within a platoon is stable. Within the following equations, variables associated with the platoon leader are denoted by the subscript *platoon* and those associated with the car directly ahead of the follower are denoted by *lead*. Swaroop's ??? solution is

$$\ddot{x}_{trail} = \frac{a_{lead} + q_3 a_{platoon} + l_1 s_1 + q_1 \Delta\dot{x}}{1 + q_3} \tag{B.9}$$

where

$$s_1 = \Delta\dot{x} + q_1(\Delta x - \Delta x_{Follow}) + q_3(v_{platoon} - \dot{x}_{trail}). \tag{B.10}$$

Within these equations, q_1, q_3, and l_1 are constant gains and Δx_{Follow} is the desired intraplatoon vehicle spacing. The values of parameters for the follower law are listed in Table B.4.

Parameter	Description	Value
q_1	Follower law gain	1.0
q_3	Follower law gain	1.0
l_1	Follower law gain	1.0
Δx_{Follow}	Desired following distance	$2.0m$

Table B.4: Follow Maneuver Parameters

Appendix C

SHIFT and C Code

C.1 Summary of SHIFT Files

File	Description
reg.hs	Supervisor automaton *regAutoAL*
reglead.hs	Automata for the leader controller
regmerge.hs	Automata for the join controller
regaccel2enter.hs	Automata for the accelerate to enter controller
regfollow.hs	Automata for the follower controller

Table C.1: SHIFT Files Summary

C.2 Summary of C Files

File	Description
vDesLead.c	Velocity profile for the leader law
vDesMerge.c	Velocity profile for the join law
vDesAccel.c	Velocity profile for the accel. to enter law

Table C.2: C Files Summary