Coordinated Adaptive Cruise Control: Design and Simulation

by

Michael Christopher Drew

B.S. (University of Virginia) 1994

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Masters of Science

in

Mechanical Engineering

in the

GRADUATE DIVISION
of the
UNIVERSITY OF CALIFORNIA, BERKELEY

Committee in charge:

Professor Karl Hedrick, Chair
Professor Pravin Varaiya

2002

Coordinated Adaptive Cruise Control: Design and Simulation

by

Michael Christopher Drew

Abstract

Coordinated Adaptive Cruise Control: Design and Simulation

by

Michael Christopher Drew

Masters of Science in Mechanical Engineering

University of California, Berkeley

Professor Karl Hedrick, Chair

Recently, some automotive companies have begun manufacturing cars with Adaptive Cruise Control (ACC) systems. These systems improve on conventional cruise control by using both throttle and braking control to track a desired velocity and to react to the presence of a lead vehicle by following at a safe and comfortable distance. Standard nonlinear control laws require knowledge of lead vehicle acceleration. However the radar systems employed can only measure range and relative velocity of the lead car - acceleration must be estimated by other means. Thus, Coordinated Adaptive Cruise Control (CACC) systems attempt to resolve this problem by using a radio to transmit the lead car velocity and acceleration data.

Here, several distance-tracking control laws are developed based on sliding mode nonlinear control - the most complex requires lead vehicle acceleration and subject vehicle jerk signals, while the simplest of which requires neither. A structure analysis of these

controllers shows that they are actually forms of linear control, and a stability analysis suggest a variant of the complex sliding mode form which allows for independent tuning of the lead vehicle acceleration term.

Simulation results using the Teja hybrid modelling platform show that proper signal processing methods must be employed for good controller performance. While the lead vehicle acceleration term is not necessary for stable control, better performance is achieved when a highly accurate signal is used in the control law. If the signal is degraded, independent tuning of the signal gain can improve performance. The lead vehicle acceleration term also proves useful in designing a stable and functional mode switching logic. Thus, the CACC offers an advantage over ACC because of its superior signal quality.

Contents

List of Figures **iii**

1 Introduction **1**
1.1 Motivation 1
1.2 Background 2
1.3 Thesis Overview 3

2 Vehicle Model Development **4**
2.1 Basic Vehicle Model Dynamics 4
2.2 Vehicle Model Additions 9

3 Communications and Signal Processing **11**
3.1 Signal Uncertainty Modelling 11
3.2 Controller Signal Processing 12

4 Controller Design **15**
4.1 Teja Controller Model Structure Overview 15
4.2 The Switching Control Level 17
4.3 Lower Level Controllers 18
4.3.1 The Engine Controller 18
4.3.2 The Brake Controller 19
4.4 Upper Level Control Law Development 19
4.4.1 Sliding Mode Control Overview 20
4.4.2 Speed Tracking Cruise Control Design 21
4.4.3 Following Mode Controller Design 23
4.4.4 Control Law Structure Analysis 27
4.4.5 Control Law Stability Analysis 31
4.5 Upper Level Mode Switching 36
4.5.1 Mode Switching Background 36
4.5.2 Mode Switching Implementation in the Teja Model 39

5 Simulation Results **44**
5.1 The Performance Index 46
5.2 Signal Processing Results 47
5.3 Control Law Results 52
5.3.1 ACC S_3 Control vs. C/ACC S_1 Control 53
5.3.2 ACC S_3 Control vs. C/ACC S_1 Control: Alternate Lead Car Trajectory 57
5.3.3 The S_1 Variant Control Law 59
5.4 Mode Switching Results 63
5.4.1 Mode Switching Simulation 1 64
5.4.2 Mode Switching Simulation 2 66
5.4.3 Mode Switching Simulation 3 69

6 Conclusions **72**

Bibliography **74**

A Teja Model ACC and CACC Signal Schematics **75**

List of Figures

2.1 Basic Vehicle Model Components in Teja 5

4.1 Controller Model Components in Teja 15
4.2 **Surface 1** Following Mode Control Law Representation of Equation 4.35. . 28
4.3 **Surface 2** Following Mode Control Law Representation of Equation 4.37. . 29
4.4 **Surface 3** Following Mode Control Law Representation of Equation 4.38. . 30
4.5 Following Mode Control Law Representation of Equation 4.42 Using a Linear Vehicle Model. 33
4.6 Simplified Closed Loop Linear Vehicle Model. 34
4.7 A Proposed Cruise Control Mode Switching System: Automation Vs. Human Control [2] 36
4.8 Upper Level C/ACC Switching Logic in Teja 40

5.1 Example Simulation Results 45
5.2 Filtered a_p Signals 48
5.3 A Comparison of Filtered Estimated a_p Signals and the Resulting Controller Performance 50
5.4 A Comparison of Control Command Using Filtered and Non-Filtered Estimated a_p 51
5.5 A Comparison of Filtered Range-Rate Signals. Velocity Sensor Noise = $\pm$.03 m/s. Radar Range-Rate Noise = $\pm$.15m/s. Packet Loss Probability: 1st = 5%, 2nd = 80% 52
5.6 Lead Car Profile for Section 5.3.1. Note the small acceleration magnitude and variance. 54
5.7 Comparison of Control Laws for Section 5.3.1. Note that the addition of a_p does not significantly improve controller performance. 55
5.8 Section 5.3.1 Comparison of a_p Signals Used in Figure 5.7 Plots (b) & (c). . 56
5.9 Lead Car Profile for Section 5.3.2. Note the larger acceleration magnitude. 57
5.10 Comparison of Control Laws for Section 5.3.2. 58
5.11 Comparison of Control Laws for Section 5.3.3. 60

5.12 Surface Behavior of Section 5.3.3 Simulation Set. Note that S_3 converges to zero, but S_1 converges to a_p when the S_1 variant control law is used with $\gamma = 0$. 61
5.13 Comparison of ACC Control Laws for Section 5.3.3. Note the improved performance of the S_1 variant - especially during the last third of the simulation. 62
5.14 Comparison of CACC Control Laws for Section 5.3.3. Note less overall improvement than in ACC mode. 63
5.15 The Upper Level Controller Finite State Machine in Teja. (Only CC $\leftrightarrow$ CACC branch shown.) 64
5.16 Velocity and Range from Section 5.4.1. 65
5.17 Velocity and Range from Section 5.4.2. 67
5.18 Velocity and Range from Section 5.4.3. 70

A.1 **ACC** Signal Flow Paths in the Teja Model. 76
A.2 **CACC** Signal Flow Paths in the Teja Model. 77

Acknowledgements

I want to thank my research advisor Professor Karl Hedrick, and all my fellow students in the Vehicle Dynamics Lab who have offered instruction and guidance in the last two years. Special thanks to Adam Howell for his work in importing the original vehicle models into Teja, and to Pete Seiler for his guidance in the control law analysis. I would also like to thank Anouck Girard for her ongoing help with installing the code on the test vehicle.

Chapter 1

Introduction

1.1 Motivation

Conventional cruise control has been offered commercially by automobile manufacturers for decades. The basic operation is simple - the user sets a desired speed, and the controller modulates the throttle in an attempt to track the desired speed. This type of cruise control is most useful on high-speed freeways with minimal traffic and grade. The system is inexpensive and reliable, but there are drawbacks. Since the system does not have braking control, it must rely on engine and air/friction drag to reduce speed. In downhill situations, for example, the car may exceed the desired speed. Furthermore, if another car cuts in front of the subject car, or if the subject car approaches a slower moving car, conventional cruise control has no means of responding. Thus, Adaptive Cruise Control (ACC) was developed. ACC controls both throttle and braking, plus it uses radar to detect the presence of a leading automobile and attempts to maintain a comfortable following distance between the two vehicles. The radar on ACC-equipped vehicles provides range and

range-rate signals, whereas range acceleration must be estimated.

An extension of Adaptive Cruise Control, Coordinated Adaptive Cruise Control (CACC) performs an identical function as ACC but instead of relying entirely on radar for lead vehicle state detection, it receives velocity and acceleration data directly from the leading vehicle via a radio communication link. This arrangement eliminates the need for range acceleration estimation, plus range-rate can be more accurately measured with less filtering. Additional information, like maximum forward acceleration and braking ability may also be shared between vehicles so that the control can be optimized for efficiency and safety. The radar is still necessary for range measurement since neither vehicle can "know" its position relative to another.

1.2 Background

ACC has been an active area of research both in industry and in academia. The multiple-layered sliding surface control approach has been developed for use on commercial applications and for precise control of vehicle platoons for the California PATH[1] project [3]. Automobiles are nonlinear systems with complexities that are difficult to model precisely. Thus, the sliding surface method of nonlinear control is chosen because of its robustness and stability properties.

This research is done in conjunction with the ongoing MoBIES[2] project. The Teja hybrid system platform is used for both the vehicle model and the controller which will be imported directly onto the actual test vehicle.

[1]PATH: Partners for Advanced Transit and Highways.
[2]MoBIES: Model Based Integration of Embedded Software.

1.3 Thesis Overview

This report covers the development of a C/ACC controller which is simulated in Teja using a vehicle model. We will begin with a discussion of the vehicle model and its governing equations in Chapter 2. Next, we will cover the modelling of the signal dynamics and filtering in Chapter 3. In Chapter 4 we will focus on the controller design and consider several control laws for distance-tracking following mode cruise control. We will also consider controller stability and suggest an alternate form of control based on the results. Finally, in Chapter 5 we will see the results of several simulations and discuss how to interpret the results.

Chapter 2

Vehicle Model Development

2.1 Basic Vehicle Model Dynamics

Since the controller will be designed and tuned in simulation, we require a vehicle model with reasonable accuracy. The basic longitudinal vehicle dynamics model was imported into Teja from a previously designed model by Adam Howell of the VDL[1]. Figure 2.1 shows the component view of the basic vehicle model within Teja. Vehicle parameters such as engine map data, mass, rolling resistance, etc., are based on data from a 1998 Buick LeSabre test vehicle.

We can see that the model includes the following primary components:

1. **SprungMass:** Contains simple motion dynamics based on the tractive force produced at the wheels (from braking or accelerating), drag forces from rolling resistance and air drag, and gravitational force due to grade.

[1]VDL: Vehicle Dynamics Lab.

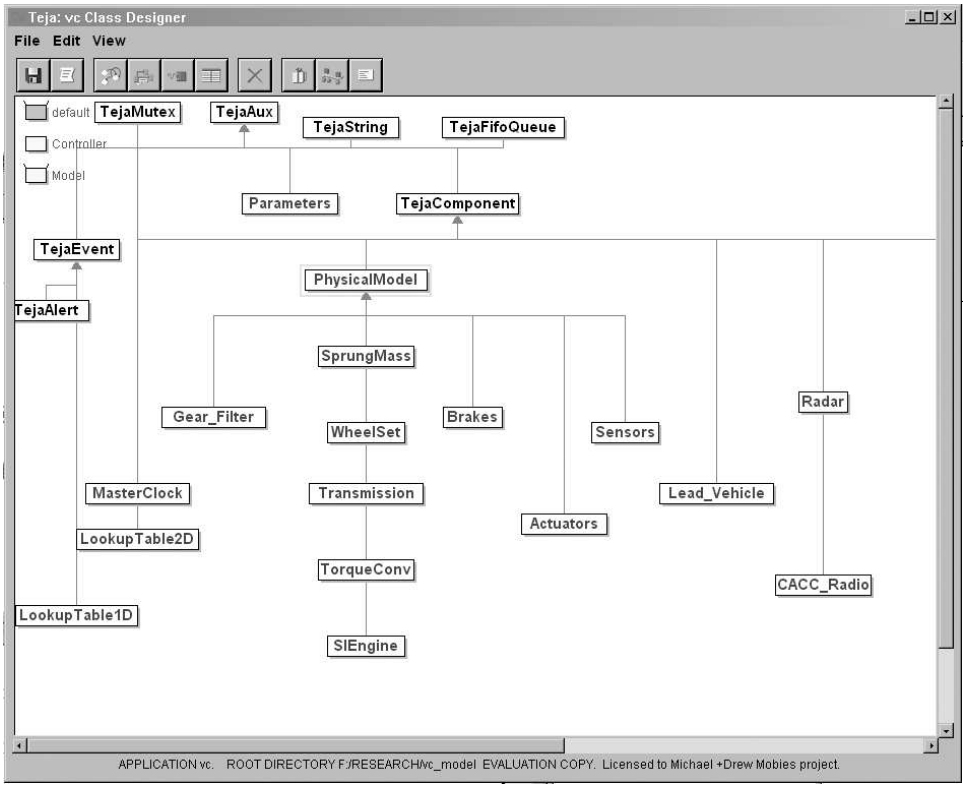

Figure 2.1: Basic Vehicle Model Components in Teja

2. **WheelSet:** Models wheel slip and inertia characteristics according to each axel.

3. **Transmission:** Contains shifting dynamics based on vehicle speed. Provides gear ratios.

4. **Gear_Filter:** This component is an addition to the original model. It is a first-order filter for the gear ratio value. This is necessary to avoid acceleration spikes due to the discontinuities of a modelled gear shift.

5. **TorqueConv:** Representing the torque converter, this component contains two modes: coupled and uncoupled according to speed ratios and gear selection. It also contains

a simplified model of transferred torque.

6. **SIEngine:** Simulates spark ignition engine dynamics using a two-state model of engine angular velocity and manifold mass flow rate. Incorporates lookup tables for manifold and throttle characteristics.

7. **Brakes:** Simple pressure dynamics model using a first-order filter to capture pressure lag.

8. **Actuators:** Models the throttle dynamics with a first-order filter, and max throttle rate limit.

9. **LookupTables:** These components handle the engine map data which is contained in external data files.

Since the emphasis of this project is the development of a vehicle controller, the detailed vehicle model dynamics will not be covered. However, a brief discussion of some of the model's basic governing equations is necessary for relating to the controller design. Thus, we start with the overall longitudinal vehicle dynamics equation [3] which is the basis for relating brake and engine torque to vehicle acceleration.

$$T_e = R_g(T_b + M_{rr} + hF_a + mgh\sin\theta) = \beta a \tag{2.1}$$

where

$$\beta = \frac{[J_e + R_g^2(J_{wr} + J_{wf} + mh^2)]}{R_g h} \tag{2.2}$$

and

$$F_a = C_a v^2 \tag{2.3}$$

β: Lumped Inertia

F_a: Aerodynamic Drag Force

T_e: Net Engine Torque

T_b: Brake Torque

R_g: Gear Ratio

M_{rr}: Rolling Resistance Torque

h: Effective Wheel Radius

m: Total Vehicle Curb Weight

J_e: Engine Inertia

J_{wf}: Front Axle Inertia

J_{wr}: Rear Axle Inertia

C_a: Aerodynamic Drag Coefficient

v: Subject Car Velocity

a: Subject Car Acceleration

θ: Road Slope

Note that although we use the road slope term θ in our equations, in simulation and implementation we currently assume the slope is zero since this quantity is hard to measure. While it is certainly possible to use estimation techniques or parameter adaption schemes to account for non-zero θ, we will not discus these here.

Turning to the engine dynamics, we simplify the system using the two-state model

proposed by Cho and Hedrick [1].

$$T_e = T_e(\omega_e, P_m) \tag{2.4}$$

where

ω_e: Engine Angular Velocity

P_m: Intake Manifold Pressure

Engine torque is calculated from both these states using an experimentally determined engine map. Also, we assume that the air in the intake manifold behaves as an ideal gas so we can easily relate manifold pressure to manifold mass flow rate. This is convenient since our control input (the throttle) directly controls the mass flow rate. Thus, the two state equations featured in the **SIEngine** component are:

$$\dot{m}_a = MAX\ TC(\alpha)PRI(m_a) - \dot{m}_{ao} \tag{2.5}$$

$$\dot{\omega}_e = \frac{(T_e - T_{pump})}{J_e} \tag{2.6}$$

where

α: Throttle Angle

MAX: Flow Rate at Full Throttle

$TC(\alpha)$: Empirical Throttle Characteristic

$\dot{m}_{ao} = \dot{m}_{ao}(m_a, \omega_e)$: From an Empirical Map

T_{pump}: Torque at the Torque Converter Pump

$PRI(m_a)$: Pressure Influence Function for Compressible Flow:

$$PRI = 1 - \exp\left(9(\frac{P_m}{P_{atm}} - 1)\right) \tag{2.7}$$

P_{atm}: Atmospheric Pressure

For the brake model, we take a simple approach by neglecting all hydraulic fluid dynamics and treat the system as a first-order filter. This captures the time lag transient between pressure at the master cylinder to the pressure at the wheel. All wheels are assumed to have the same brake pressure at their calipers.

$$\dot{P}_{wheel} = (P_{mc} - P_{wheel})/\tau \tag{2.8}$$

$$\tau = 0.1$$

$$T_b = K_b(P_{mc} - P_{po}) \tag{2.9}$$

where

K_b: Experimentally Determined Brake Gain

P_{po}: Push-out Pressure - required to engage brake

2.2 Vehicle Model Additions

With the basic components of the vehicle model in place, additional components representing the vehicle's sensors and communications are added as follows:

1. **Sensors:** Produces subject vehicle velocity and acceleration signals. Noise is added independently to both signals to simulate real sensor characteristics.

2. **Radar:** Simulates the radar installed on C/ACC-equipped vehicles. Produces range (relative vehicle distance), and range-rate (relative vehicle velocity) signals. Again,

noise is added to each signal. Also provides an indicator flag for the presence of a lead vehicle.

3. **CACC_Radio:** Simulates the radio installed for CACC communication. Lead vehicle velocity and acceleration is produced. Noise is added to simulate the lead car sensor behavior, and a probabilistic packet loss model is used to simulate the RF communication link. An indicator flag signals the presence of a CACC-equipped lead car. We will discuss this in more detail in the following section.

4. **Lead_Vehicle:** Simulates the behavior of a lead vehicle for simulation purposes. Purely simulated maneuvers can be specified, or velocity and acceleration profiles can be imported.

5. **Data_Acquisition:** This component acts as the channel through which all data is transferred between the controller and the model. The only computation performed is that of β (the lumped inertia parameter) which must be continually updated to be used by the lower controllers because it is a function of the current gear ratio.

In the following chapter we will discuss the specifics of signal noise and uncertainty dynamics in the **Sensors**, **Radar**, and **CACC_Radio:** components in more detail.

Chapter 3

Communications and Signal Processing

Indeed, the area of communications and signal processing is a topic unto itself. Specifically, in the realm of controls, care must be taken when designing a controller which relies on signals having their own channel dynamics. In this chapter, we discuss the modelling of the communication between the controller and the subject and lead vehicles.

3.1 Signal Uncertainty Modelling

Since an actual control system installed on a vehicle must be robust enough to handle signal uncertainties like noise and packet loss, we attempt to capture these uncertainties in the model. The components which simulate noise (**Sensors**, **Radar**, and **CACC_Radio**) do so by adding randomly generated values to the nominal signal at every time step in the simulation. Thus, the noise is random, uncorrelated (white) noise which is

a reasonable assumption for many sensor signals. The upper and lower bounds of the noise can be specified and adjusted independently for all these "noisy" signals. Typically, we add more noise to acceleration signals than to velocity/range-rate signals since acceleration measurement is more difficult.

In the case of the **CACC_Radio** component, noise is added to the lead vehicle velocity and acceleration signals to simulate the noise of the lead vehicle's sensors. The additional uncertainties of packet loss are also simulated using a second-order Markov chain probabilistic model. We prescribe two probabilities: first, the probability of loosing a packet given that the previous packet was *not* lost, second, the probability of losing a packet given the previous packet *was* lost. Typically the first probability is much lower than the second. This arrangement simulates the tendency to loose multiple packets at a time due to radio uncertainties like multipath and interference. In other words, a packet is not lost often, but when it is, there is a higher chance of loosing the next one due to the presence of physical and/or environmental conditions.

3.2 Controller Signal Processing

Just as we have added noise on the vehicle side of the model to simulate the actual signal dynamics, we now desire to remove as much of the noise as possible on the controller side. For ACC control, we also have to estimate the lead car acceleration signal a_p since the radar can only detect distance and rate. While there are many techniques of differentiating a signal we have elected to use a simple but crude method of numerical differentiation. In the Teja simulation, there are 3 components which handle the task of filtering the radar

and CACC radio signals and estimating a_p from the radar:

1. **A_p_Estimator:** This component estimates lead vehicle acceleration by numerically differentiating the unfiltered range-rate signal and adding it to the unfiltered subject vehicle acceleration. ($a_p = \ddot{r} + a$)

2. **Radar_Filter:** This component filters the range signal (which is used in both ACC and CACC control), and the range-rate signal which both come directly from the radar. It also filters the estimated a_p signal which comes from the **A_p_Estimator** component. A first-order filter is used for range filtering, and a second-order filter is applied to the range-rate and a_p signals. All three of these signals are used in ACC mode.

3. **CACC_Filter:** Similar to the **Radar_Filter** component, this component filters both the v_p and a_p signals coming from the **CACC_Radio** component using second-order filters. Both of these signals are used in CACC mode only.

The first-order range filter is specified by its time constant τ. Typically, however, since the noise of the range signal is relatively small, the time lag of the filter is a greater detriment to performance than the noise, so this filter is usually turned off.

For the second-order filters in the **Radar_Filter** and **CACC_Filter** components, separate Matlab files were created for both of these components which create filters using one of the many available digital filters (eg. Butterworth, Chebyshev type I or II, etc.). The filters are then applied to data from a previous Teja simulation so they can be tuned. Since Teja simulates continuous-time state-space dynamics, the Matlab file converts the digital filter into a continuous-time transfer function based on the sampling rate. Finally, it writes

the transfer function coefficients to a data file which Teja parses during the simulation. Each of the 4 second-order filters is constructed in a canonical state-space form so the continuous-time filter results match the results from the Matlab functions.

Each of the second-order filters are designed this way by trial error using the Matlab programs. There is always a trade-off between reduced noise and signal resolution and lag. Certainly a more rigorous approach to filter design could be applied, but it is not the emphasis of this project. Within the realm of simulation, these basic signal processing models are sufficient as a base-line for the testing and designing of the controller. They effectively demonstrate the importance of noise and uncertainty and the role of filtering as applied to this problem as we will see in Chapter 5. For a detailed schematic of the signal connections used in ACC and CACC control, refer to Appendix A.

Chapter 4

Controller Design

4.1 Teja Controller Model Structure Overview

We now consider the development of the vehicle controller. In general, we can think of a controller as a system which attempts to force another system (plant) to follow a desired trajectory by directing inputs to the system. Often, the controller my also have to calculate the trajectory in real time in order to achieve a higher-level, preset desired result.

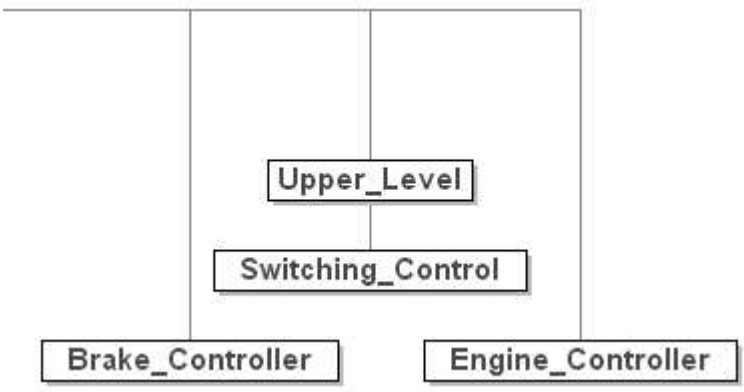

Figure 4.1: Controller Model Components in Teja

In the case of a longitudinal vehicle controller, our plant is the vehicle whose only inputs are throttle angle and brake master cylinder pressure. The trajectory is the acceleration of the vehicle which may be positive or negative. The desired result for C/ACC control is a desired speed and a desired following distance when following a leading car travelling slower than the desired speed.

Accordingly, the implementation of the controller in Teja has been divided into three levels as shown in Figure 4.1. The **Upper_Level** component calculates a desired acceleration trajectory based on a desired velocity or following distance using defined control laws. It also contains algorithms for switching between the various controller modes.

The **Switching_Control** component represents an intermediate level of the controller. Based on the desired acceleration calculated by the **Upper_Level** component, it decides whether braking input, or throttle input is necessary to achieve the desired result.

Finally, the lower levels of the controller are contained in the **Brake_Controller** and **Engine_Controller** components. These components first convert the desired acceleration into a desired torque, then they produce an input signal of brake pressure or throttle angle using vehicle-specific models.

There are many advantages to this type of control structure. Perhaps the greatest is that once the lower and intermediate level controllers are developed specifically for the vehicle, the upper level can be designed and tuned without much consideration of the lower level details. Thus, a robust high-level controller may be designed and implemented on one vehicle, then translated to another with only minor tuning. In the following sections we will cover the details involved in the design of all three levels of control. We will begin our

discussion with the lower-level controllers, but we will cover the upper-level controller in the greatest depth, since it is the most interesting and complex level.

4.2 The Switching Control Level

The Switching Control component shown in Figure 4.1 contains switching logic based on the a_{des} calculated from the Upper Level component. The switching criteria has an offset based on the fact that at closed throttle, there is a residual acceleration due to engine, air/friction drag, and road grade. Thus, determining whether to use engine or brake inputs involves more than just determining if a_{des} is positive or negative. For instance, if a_{des} is negative, but the residual acceleration is large enough in magnitude, we may not require any brake input. Residual acceleration can be derived from the vehicle longitudinal dynamics equation 2.1 as follows [3] :

$$a_{resid} = \frac{1}{\beta}[T_{ect} - R_g(M_{rr} + hF_a + mgh\sin(\theta))] \tag{4.1}$$

where T_{ect} is the closed throttle torque of the engine. This value is typically determined experimentally according to engine speed and torque converter dynamics. For this simulation, we have estimated T_{ect} as zero since, for efficiency reasons, the transmission/engine controller on the Buick LeSabre was designed to reduce the effect of engine drag.

Once the residual acceleration is calculated, the Switching Control uses the following criteria to determine whether engine or braking control is required:

$$a_{des} \geq a_{resid} \quad \Rightarrow \text{Engine Control} \tag{4.2}$$

$$a_{des} < a_{resid} \quad \Rightarrow \text{Brake Control} \tag{4.3}$$

To prevent rapid chattering between both control modes (especially during steady-state), hysteresis is added to this scenario to "damp" the response.

4.3 Lower Level Controllers

Once the Switching Control level has determined whether engine or brake control is necessary, the selected lower level controller must convert this acceleration into the appropriate input to the car.

4.3.1 The Engine Controller

If engine control is required, the Engine Controller component first converts the desired acceleration into a desired engine torque. Again, we derive this value from the longitudinal vehicle dynamics equation 2.1 [3].

$$T_{edes} = \beta a_{des} + R_g(M_{rr} + hF_a + mgh\sin(\theta)) \tag{4.4}$$

Next, the desired engine torque is converted into a throttle angle command α by inverting an engine map based on the current engine speed ω_e. This is accomplished by interpolating the data from a specific engine map lookup table.

$$\left.\begin{array}{r} T_{edes} \\ \\ \omega_e \end{array}\right\} \Rightarrow \{\text{Inverse Engine Map}\} \Rightarrow \quad \alpha \tag{4.5}$$

Even though this method is only as accurate as the engine map data, it is sufficient for this application since an appropriately robust controller will compensate for these inaccuracies.

4.3.2 The Brake Controller

Similarly, if braking control is required, the Brake Controller component computes a desired brake torque by using another form of equation 2.1 [3].

$$T_{bdes} = -\frac{(\beta a_{des} + T_{ect})}{R_g} - (M_{rr} + hF_a + mg\sin(\theta)) \tag{4.6}$$

Then, equation 2.9 is used to find the brake control input P_{mc}.

4.4 Upper Level Control Law Development

When designing a C/ACC controller, we are actually designing two controllers. If our subject vehicle cannot detect a leading car in the same lane travelling slower than our desired speed, we are not in a following mode and need only track a desired speed. Thus, we need at least two control law regimes: one for desired velocity tracking (what we will call **CC** mode), the other which tracks a desired following distance between our subject vehicle and a detected lead vehicle. Since we are considering coordinated adaptive cruise control, we must further divide the distance-tracking regime into two modes: the coordinated adaptive mode (which we will call **CACC**) which actively communicates with the lead car, and the default following mode which detects the lead car using radar (which we will call **ACC** mode). The ACC mode is necessary since we cannot assume that all other cars will have communication abilities. Also, if the CACC signal becomes degraded, we can use the ACC mode as a backup.

Therefore, the upper level controller will consist of at least three modes. Additional modes have been added which smooth the transitions between the three primary

modes. However, it is important to point out that there are only two basic control regimes: speed tracking, and distance tracking - each having their own control laws. The only difference between ACC and CACC is how the pertinent signals are detected (radar vs. radio communication).

Each of these modes can utilize throttle *and* braking control of the vehicle. For this project we have primarily used sliding surface control for both modes. In the following sections we will first provide an overview of sliding surface control, then we will discuss the details of the various control laws used in this project. Based on the structure and formulation of our sliding surface controllers, we will see an interesting result relating nonlinear surface control to linear control techniques. We will then analyze the stability of these controllers, and finally, we will touch on the complexities of designing the criteria used for switching among the upper level control modes.

4.4.1 Sliding Mode Control Overview

Since an automobile is an inherently complex nonlinear system, we would naturally choose a nonlinear control technique with robustness properties capable of dealing with the many uncertainties of the system. The simplicity and robustness properties of sliding surface control make it appealing for use in both of the control regimes.

Sliding mode control requires choosing a proper surface S where S is a function of the plant's states, or pseudo-states. The condition of $S = 0$ must define a manifold on which the closed-loop system is asymptotically stable. Also, a reachability condition must be defined which forces S towards zero either in a finite time, or asymptotically. In general,

this reachability condition is defined as [5]

$$\dot{S} = -\gamma(S) \tag{4.7}$$

where $\gamma(S)$ satisfies:

1. $\gamma(S)$ is continuous if $S \neq 0$.

2. 4.7 is globally asymptotically stable.

To satisfy condition 2, we must have that

$$S\gamma(S) > 0, \quad S \neq 0 \, .$$

Also, $\gamma(S)$ must be bounded for $S \in \mathcal{N}_\delta(0)$, where $\mathcal{N}_\delta(0)$ is a δ neighborhood of the origin.

Thus, one can view sliding mode control as consisting of two parts. The first mode involves S approaching zero according to its defined dynamics. Once $S = 0$ is achieved, the second mode (the so-called sliding mode) begins. The state trajectories are then on the stable manifold and "slide" towards the origin.

4.4.2 Speed Tracking Cruise Control Design

Although the sliding surface definition is not unique, one conventional method of defining it is as follows:

$$S = \left[\frac{d}{dt} + \lambda\right]^{r-1} \varepsilon \tag{4.8}$$

where

r: Relative degree of the system

λ: Sliding surface gain (strictly positive)

ε: System error which we want to drive to 0

In the above equation, r represents the relative degree of the system. This is the number of times the output appearing in ε requires differentiation for the control input to show up. The sliding surface gain, λ is a gain which tunes the rate of convergence to $\varepsilon = 0$. With this method, the control input will be in the first derivative of the S function. Thus, solving for this input gives us a control law.

For the speed tracking mode $r = 1$, so we can define our error and the surface as

$$\varepsilon = v - v_{target} \tag{4.9}$$

$$S_{cc} = \varepsilon = v - v_{target} \tag{4.10}$$

Now we must decide how we want to force $S = 0$, ie., we need to define $\gamma(S)$. Common definitions of $\gamma(S)$ in the sliding surface control literature include the discontinuous equation $\gamma(S) = kS + k_0 sgn(S)$ and its continuous round-off variants. In this project, we have chosen $\gamma(S) = KS$ which forces $S = 0$ asymptotically. This is easier to implement because of its simplicity and because it is continuous. We have

$$\dot{S}_{cc} = -KS_{cc} \tag{4.11}$$

which satisfies the global asymptotic stability requirement since $\gamma(S)$ is bounded and satisfies $S\gamma(S) > 0$. Now we can solve for a desired acceleration:

$$a_{des} = \dot{v}_{target} - K(v - v_{target}) \tag{4.12}$$

This control law formulation for the speed tracking mode is rather simple. In fact, if we assume that $\dot{v}_{target} = 0$ it is just negative feedback proportional control. Also, since

velocity is the only required state for this controller, implementation is easy. However, this controller may suffer from being to harsh for the vehicle's occupant, so other methods may be employed to improve the "feel" of the ride. Since this project concentrates on C/ACC design, we will not consider these methods. Yet, the observation that the sliding mode approach produced a linear control law is worth noting. We will discuss this in further detail in the following sections.

4.4.3 Following Mode Controller Design

Prior to developing a surface formulation for the following regime (for use in both ACC and CACC modes), we need a definition of the system. We can define a two-state model using the following distance r and $\dot{r}$. Namely,

$$r = x_p - x \tag{4.13}$$

$$\dot{r} = v_p - v \tag{4.14}$$

We have defined x and v as the subject vehicle position and velocity, and x_p and v_p as the lead vehicle position and velocity. Now, we can define a following distance law based on the subject vehicle's velocity as

$$r_{des} = \sigma v + L \tag{4.15}$$

$$\dot{r}_{des} = \sigma a \tag{4.16}$$

where L is typically a car's length and σ may be viewed as a following time, or headway time between the two vehicles. Defining error and using a proper surface definition, we can define a sliding surface.

Surface Definition 1

For reasons which we will discuss momentarily, we will investigate a number of surface definitions. The first surface follows directly from equation 4.8. Together with the asymptotic reachability condition discussed above, we have:

$$\varepsilon = r - r_{des} \tag{4.17}$$

$$S_1 = \dot{\varepsilon} + \lambda\varepsilon \tag{4.18}$$

$$S_1 = \dot{r} - \dot{r}_{des} + \lambda(r - r_{des}) \tag{4.19}$$

$$\dot{S}_1 = \ddot{r} - \dot{r}_{des} + \lambda(\dot{r} - \dot{r}_{des}) \tag{4.20}$$

$$\dot{S}_1 = -KS_1 \tag{4.21}$$

As a simple test of stability for this surface choice, we can set $S_1 = 0$ in equation 4.18, and we get $\varepsilon(t) = e^{-\lambda t}$ which is asymptotically stable for all positive λ. Since the control input to the car (plant) is in the form of throttle or braking control, we combine equations 4.20 and 4.21 and solve for a desired acceleration, a_{des}:

$$a_{des} = \frac{1}{1 + \lambda\sigma}\left[KS_1 + \lambda\dot{r} - \ddot{r}_{des} + a_p\right] \tag{4.22}$$

Once a_{des} is computed, it is passed on to the lower controllers where it is converted to either a throttle angle or brake pressure input. However, a quick assessment of equation 4.22 reveals that it requires knowledge of leading vehicle acceleration and subject vehicle jerk. These values are difficult to measure in actual implementation, and even in simulation, jerk can only be estimated with numerical differentiation. If we ignore both a_p and $\ddot{r}_{des}$, we end up with a disturbance Δ_1 in equation 4.21:

$$\dot{S}_1 = -KS_1 + \Delta_1 \tag{4.23}$$

where $\Delta_1 = a_p - \ddot{r}_{des}$. Fortunately, we can expect that a_p and $\ddot{r}_{des}$ are reasonably bounded, so while S_1 may not reach zero, it will reach a reasonable boundary layer.

Surface Definition 2

The presence of unmeasurable terms in the first surface definition is motivation for an alternative surface which neglects both subject vehicle jerk and leading vehicle acceleration. We can define S_2:

$$S_2 = \dot{r} + \lambda(r - r_{des}) \tag{4.24}$$

$$\dot{S}_2 = \ddot{r} + \lambda(\dot{r} - \dot{r}_{des}) \tag{4.25}$$

$$\dot{S}_2 = -KS_2 \tag{4.26}$$

Considering the stability of this surface, we can rewrite equation 4.24 as

$$S_2 = \dot{\varepsilon} + \lambda\varepsilon + \dot{r}_{des} \tag{4.27}$$

and, if we assume that $\dot{r}_{des}$ is constant (an assumption we have already made by assuming that vehicle jerk ≈ 0), we find that at $S_2 = 0$, the solution to 4.24 is $\varepsilon(t) = e^{-\lambda t} - \dot{r}_{des}/\lambda$. Thus, S_2 is stable, but not asymptotically as before since $\varepsilon \to -\dot{r}_{des}/\lambda$ which is bounded. Now we have a new definition of a_{des}:

$$a_{des} = \frac{1}{(1 + \lambda\sigma)}\left[KS_2 + \lambda\dot{r} + a_p\right] \tag{4.28}$$

Note that this definition of a_{des} is not equivalent to dropping the $\ddot{r}_{des}$ term from equation 4.22 since we have defined a different surface from the start. Also, this time a_{des} does not have a jerk term, but still requires knowledge of lead vehicle acceleration. If we assume $a_p = 0$, we will again have a bounded error, and equation 4.26 will become $\dot{S}_2 = -KS_2 + \Delta_2$

where $\Delta_2 = a_p$. It is important to realize, however, that although $||\Delta_2|| \leq ||\Delta_1||$ for the surface trajectories, the error trajectory for S_1 is asymptotically stable whereas for S_2 it is not. Hence, both of these surface choices involve bounded errors when setting vehicle jerk and lead vehicle acceleration equal to zero. It is difficult to see whether S_2 offers any advantage over S_1 other than being slightly easier to compute.

Surface Definition 3

Is it possible to define a surface which does not require knowledge of either subject vehicle jerk ($\dot{a}$), or lead vehicle acceleration (a_p)? Looking again at equation 4.8, and noticing that we have defined $r_{des} = \sigma v + L$, we find that the relative degree of our system is actually $r = 1$ since differentiating r_{des} produces σa. Thus, we can legally define a third surface S_3:

$$S_3 = \varepsilon = r - r_{des} \tag{4.29}$$

$$\dot{S}_3 = \dot{r} - \sigma a \tag{4.30}$$

$$\dot{S}_3 = -KS_3 \tag{4.31}$$

This surface is obviously stable since at $S_3 = 0,\ \varepsilon = 0$. Solving for a_{des}, we get

$$a_{des} = \frac{1}{\sigma}(KS_3 + \dot{r}) \tag{4.32}$$

This time a_{des} only requires knowledge of lead and subject vehicle velocity and following distance. We have a viable, stable control law which does not require any immeasurable terms or any assumptions to avoid those terms.

Nevertheless, intuition tells us that knowing lead vehicle acceleration should be advantageous for better performance, since it translates directly to our desired acceleration;

ie. it allows for a quicker response to the behavior of the leading vehicle. In the next section we will analyze why this intuition is true, and consider alternative methods of constructing a control law.

4.4.4 Control Law Structure Analysis

Recall that in Section 4.4.2 we found that applying sliding surface control to the velocity tracking controller simplified to become simple proportional negative feedback control. Similarly, for the following mode controllers, let us consider surface definition 1 where equation 4.22 can be written as:

$$a_{des} = K_1 S_1 + K_2 \dot{r} - K_3 \ddot{r}_{des} + K_3 a_p \tag{4.33}$$

where

$$K_1 = \frac{K}{(1+\lambda\sigma)}, \quad K_2 = \frac{\lambda}{(1+\lambda\sigma)}, \quad K_3 = \frac{1}{(1+\lambda\sigma)} \tag{4.34}$$

Substituting for S_1 and using $\dot{r} = \dot{\varepsilon} + \dot{r}_{des}$ gives us

$$a_{des} = (K_1 + K_2)\dot{\varepsilon} + K_1\lambda\varepsilon + K_2\dot{r}_{des} - K_3\ddot{r}_{des} + K_3 a_p \tag{4.35}$$

The first two terms of equation 4.35 are recognizable as proportional and derivative feedback, while the last 3 terms are typically viewed as feedforward from the r_{des} and x_p signals. However, since we are relating r_{des} to vehicle velocity by $r_{des} = \sigma v + L$, the r_{des} signal is actually generated by the plant. This is made clear by constructing the system in transfer function form as shown in Figure 4.2. This construction allows us to view the system in single-input-single-output form, where the only input is considered to be the lead vehicle position x_p, and the output is the error ε. While in real life, only relative position

has meaning, this form is convenient and mathematically correct for analysis purposes. Note that the plant block P represents all the nonlinearities of the vehicle - the output of which is the actual acceleration a. Also, if we wish to neglect subject vehicle jerk, we simply eliminate the $K_3 s^2$ portion of the r_{des} loop.

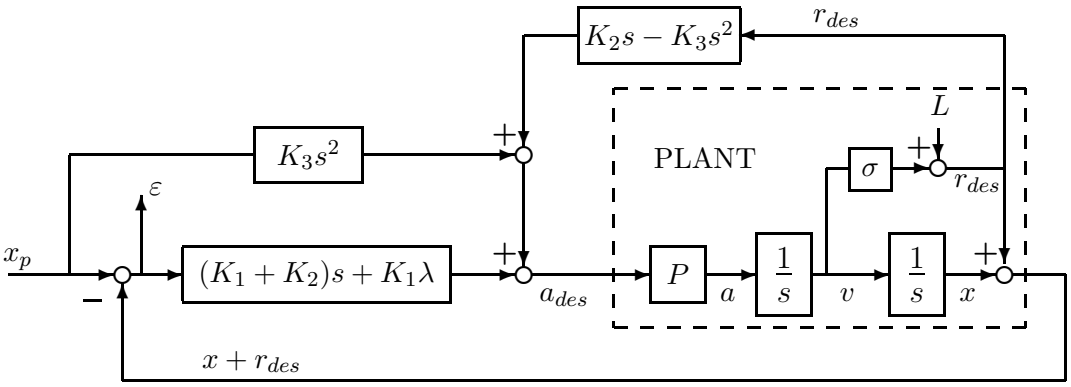

Figure 4.2: **Surface 1** Following Mode Control Law Representation of Equation 4.35.

Similarly, we can reconstruct equation 4.28 of the second surface definition into

$$a_{des} = K_1 \dot{r} + K_1 \lambda \varepsilon + K_2 \dot{r} + K_3 a_p \tag{4.36}$$

where we have used the identical gain definitions from equations 4.34. Simplifying, and substituting $\dot{r} = \dot{\varepsilon} + \dot{r}_{des}$ we have

$$a_{des} = (K_1 + K_2)\dot{\varepsilon} + K_1 \lambda \varepsilon + (K_1 + K_2)\dot{r}_{des} + K_3 a_p \ . \tag{4.37}$$

Again, we have similar feedback and feedforward terms to those in equation 4.35 of S_1. This version of the controller is shown in Figure 4.3 which proves that the only

difference between these two control laws is in the r_{des} feedforward loop. Furthermore, if we neglect the $K_3 s^2$ term in Figure 4.2, we have essentially identical controllers except for the magnitude of the gain ($K_1 + K_2$ instead of K_2).

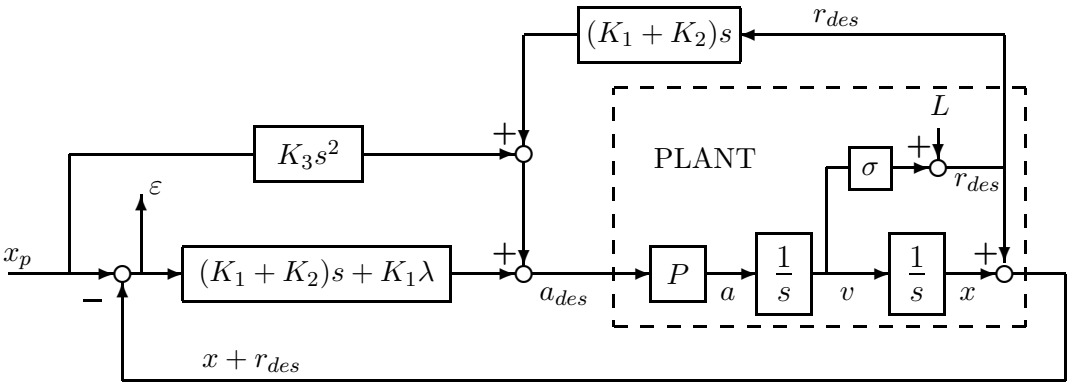

Figure 4.3: **Surface 2** Following Mode Control Law Representation of Equation 4.37.

Finally, for the 3rd surface definition, S_3, equation 4.32 can be written as

$$a_{des} = \frac{1}{\sigma}[\dot{\varepsilon} + K\varepsilon + \dot{r}_{des}] \tag{4.38}$$

which can be constructed as in Figure 4.4. Again, we find that this controller contains PD feedback, and we still have a derivative term from the r_{des} signal. The only difference is the lack of the x_p feedforward loop (the a_p term) which, as we discussed in the previous section, should provide improved performance. Nevertheless, although the initial form of the S_3 control law may have appeared to differ greatly from the first two controllers, this analysis proves they are remarkably similar.

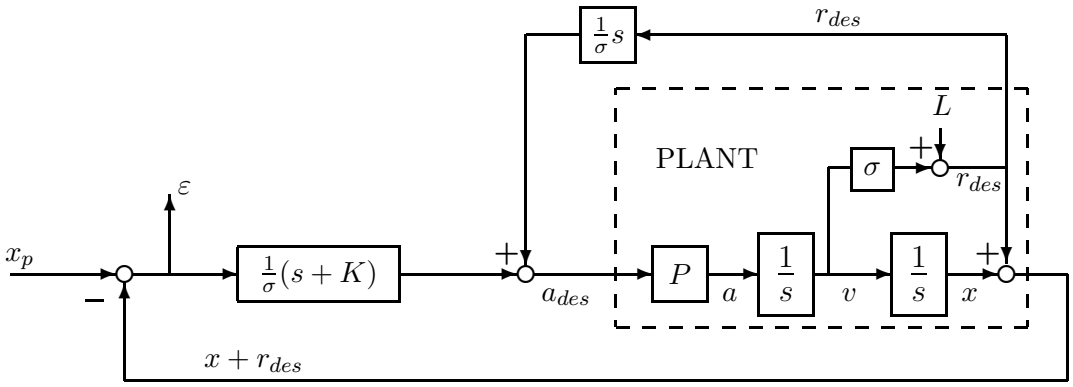

Figure 4.4: **Surface 3** Following Mode Control Law Representation of Equation 4.38.

Not only are these three controllers similar to each other, it is also clear that they are all forms of *linear* control. Furthermore, since the lower-level controllers are simply conversions from a desired acceleration to throttle and brake input, we are not really using the sliding mode controller to cancel any nonlinearities. In essence, we are assuming that our plant is linear in that whatever acceleration the upper-level request we will receive - albeit with some lag and possible saturation.

This realization raises an important question: Have we gained anything by using the sliding mode technique for this application? Since this form of nonlinear control is a subset of linear control, would not a linear controller be better suited for this application? Looking again at equation 4.35 we see that gains K_1, K_2, and K_3 are all coupled to our sliding mode gains K and λ, and our desired following distance headway parameter σ. Intuition suggests that we would be better off having the flexibility to tune each of the

control law terms individually. For example, we can see that raising the gain of a_p, (K_3) involves lowering λ since σ is fixed. But this comes at the expense of lowing the gain of ε which may not be beneficial. We will consider these questions in further detail in the following section as we consider controller stability, and later, where we analyze some simulation results.

4.4.5 Control Law Stability Analysis

In this section we will consider the closed loop stability of the following mode controllers. The results of the previous section demonstrate that the 3 control laws are nearly identical in structure. Therefore, we will begin our discussion with the first control law based on the S_1 surface since it is the most general.

Initially, we might be tempted to assume that the S_1 control law is stable based solely on the properties of its construction (Section 4.4.3). However, we have defined the desired following distance to be a function the vehicle's velocity ($r_{des} = \sigma v + L$) which is a state of the plant. Thus, in essence, we have a positive feedback loop in our control structure (see Figure 4.2). Considering only the dynamics of the $S_1 = 0$ is not sufficient; we must consider the remainder dynamics of the system on the sliding manifold. To do this, we can transform the plant dynamics into $(\bar{r}, S_1)$ coordinates [5]. Based on equation 4.18 we have

$$\sigma\ddot{\bar{r}} + (1 + \lambda\sigma)\dot{\bar{r}} + \lambda\bar{r} = S_1 + \sigma a_p + \lambda\sigma v_p \tag{4.39}$$

$$\dot{S}_1 = -KS_1 \tag{4.40}$$

$$\bar{r} = r - L \tag{4.41}$$

The transformed system has de-coupled the r_{des} terms into lead car states and range states. Since $S_1 \rightarrow 0$ asymptotically, we can consider the remainder dynamics by setting $S_1 = 0$ in equation 4.40 and we find its poles are

$$\lambda_1 = -\lambda$$

$$\lambda_2 = -\frac{1}{\sigma}$$

Thus, the closed loop system is globally stable for all $\lambda > 0$, and all $\sigma > 0$. Similar results can be shown for the S_2 control law (see [5]) which can be extended for the S_3 control law.

In the previous section we claimed that the sliding mode controllers are simply forms of linear feedback and feedforward control. Let us now investigate the stability of this generalized form beginning with equation 4.35 which can be written as

$$a_{des} = A\dot{\varepsilon} + B\varepsilon + C\dot{r}_{des} + Da_p \tag{4.42}$$

where we have dropped the $\ddot{r}_{des}$ term since it involves knowledge of vehicle jerk. Since we are considering stability of the closed loop structure shown in Figure 4.2, we need to make some assumptions of the plant P. For this analysis we will use a common technique of approximating the vehicle model as a double integrator with an input first-order lag having a time constant τ. This closed loop structure of equation 4.42 can be represented as in Figure 4.5.

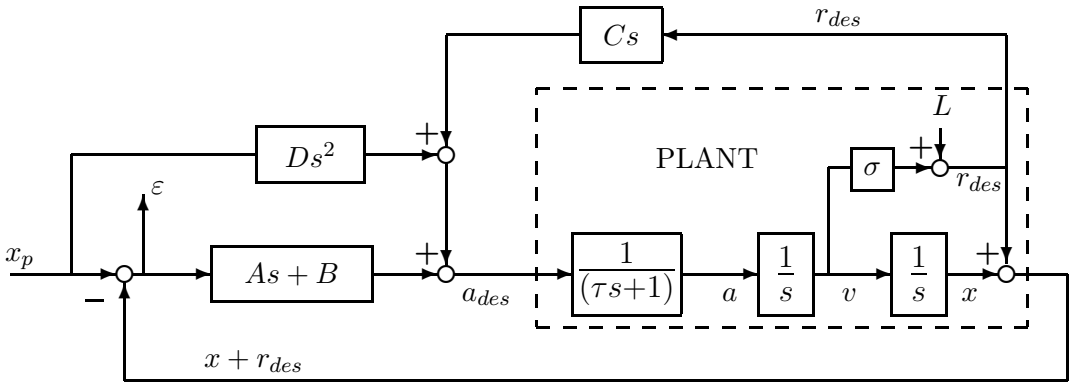

Figure 4.5: Following Mode Control Law Representation of Equation 4.42 Using a Linear Vehicle Model.

To analyze the stability of this closed loop system, we first simplify the loop as shown in Figure 4.6 where

$$G_1(s) = \frac{1}{s(\tau s + 1 - C\sigma)} \tag{4.43}$$

and

$$G_2(s) = \frac{(\sigma s + 1)}{s} \tag{4.44}$$

This structure is further simplified giving us the transfer function from $X_p(s) \rightarrow \varepsilon(s)$

$$\frac{\varepsilon(s)}{X_p(s)} = \frac{1 - Ds^2 G_1(s) G_2(s)}{1 + (As + B) G_1(s) G_2(s)} \tag{4.45}$$

$$= \frac{s^2[(\tau - D\sigma)s + 1 - C\sigma - D]}{\tau s^3 + (1 - C\sigma + A\sigma)s^2 + (A + B\sigma)s + B} \tag{4.46}$$

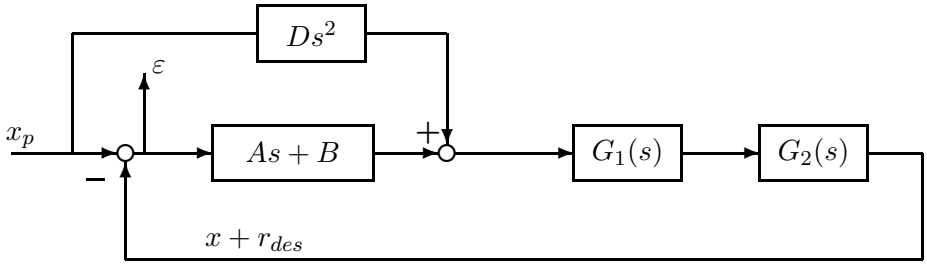

Figure 4.6: Simplified Closed Loop Linear Vehicle Model.

Now we can utilize the Routh-Hurwitz criterion on the denominator of 4.46 which provides bounds for stability,

$$\tau > 0$$

$$1 + \sigma(A - C) > 0$$

$$A + B\sigma - \frac{\tau B}{1 + \sigma(A - C)} > 0$$

$$B > 0$$

The first of these inequalities is obvious, and the last is not surprising if we relate B to the definition of K_1 in equation 4.35. However, the second and third are considerably more complex. Thus, whereas the sliding mode controllers may be more restrictive due to their coupled gain structure, this generalized form of feedforward/feedback linear control requires more tuning effort because there are more gains to deal with and more restrictions on their regions of stability. Though our plant has effectively been linearized by the modular form of the overall controller design, there is a significant advantage of simplicity and elegance of the sliding mode control which is not evident in the general linear control

structure.

Note that the feedforward gain D is not involved in the above stability criterion. This is because we may think of the feedforward signal as a disturbance; as long as x_p is bounded and the transfer function (Ds^2) is stable, it will not de-stabilize the system. This fact is actually quite beneficial. We have made the claim that if we can know the state of the lead vehicle with better accuracy - especially the lead vehicle acceleration - we can improve the performance of our controller.

The S_3 controller lacks this feedforward term, so perhaps it will not offer the CACC controller much of an advantage over ACC control. In addition, the S_1 and S_2 controllers are limited in their flexibility of the a_p signal since K_3 (of equations 4.35 and 4.37) is dependent on λ and σ(which is not a controller gain). It would seem logical to improve the transient response of the controller by raising the a_p gain, but K_3 can not be increased without decreasing λ at the possible expense of steady state performance. In fact, the maximum value of K_3 is 1. Perhaps we can get the best of both the generalized controller, and the sliding mode control laws by modifying the sliding mode controller so that it has an independent gain on just the a_p term. We can do this without adversely affecting stability. Thus, returning to the original form of the control laws (equations 4.22 and 4.28) the S_1 and S_2 control laws become:

$$S_1 :\Rightarrow \quad a_{des} \quad = \quad \frac{1}{1+\lambda\sigma}[KS_1 + \lambda\dot{r} - \ddot{r}_{des}] + \gamma a_p \tag{4.47}$$

$$S_2 :\Rightarrow \quad a_{des} \quad = \quad \frac{1}{1+\lambda\sigma}[KS_2 + \lambda\dot{r}] + \gamma a_p \tag{4.48}$$

We will investigate the performance of these modified control laws in the next chapter.

As a final note, we should point out that, although the feedforward term γa_p is

in theory bounded, the a_p signal is typically noisy and difficult to measure in practice - especially if a_p is determined from the radar's range-rate ($\dot{r}$) signal as it is in ACC mode. Even if we pre-filter a "clean" a_p signal from the CACC radio, we must be cautious about tuning γ. The residual noise and lag of a filtered a_p signal will impose upper bounds on γ.

4.5 Upper Level Mode Switching

4.5.1 Mode Switching Background

The criteria used for switching between the controller's several modes can be extremely complex, and subject to the opinion and philosophy of the designer. Compromises between automation and human interface quickly arise. For example, which transitions should be controlled by the user, and which should be done automatically? One such approach for error handling switching logic is outlined in Figure 4.7 as proposed by Girard [2].

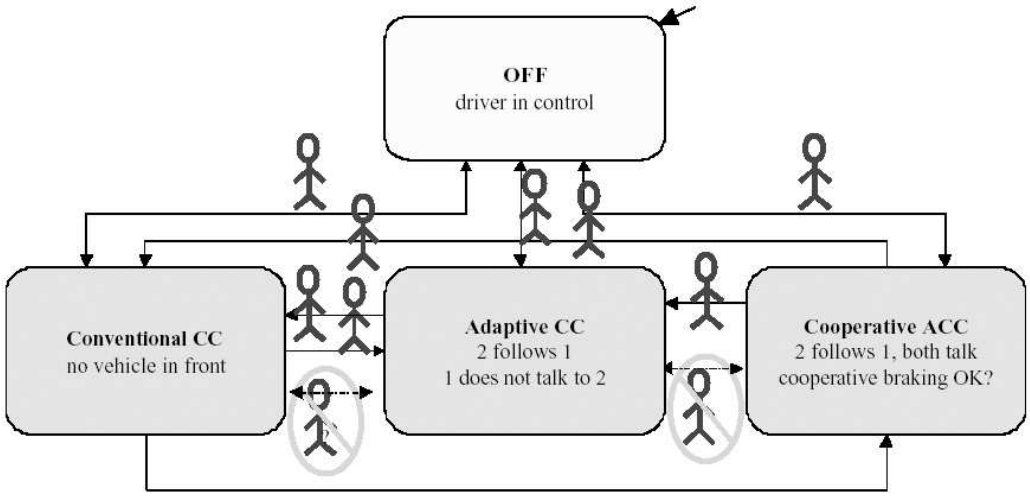

Figure 4.7: A Proposed Cruise Control Mode Switching System: Automation Vs. Human Control [2]

Arrows with stick figures represent transitions that the user can control. Transitions with a crossed-out stick figure represent transitions that should occur automatically. The dashed arrows represent transitions during errors or system faults. The proposed arrangement of Figure 4.7 allows user control between the off state and the individual cruise control modes. This way, the user can turn any of the three levels of cruise control on or off. Transitions between the individual levels can also be made by the user. Alternatively, in the case of a sensor failure, automatic transitions may take precedence without any human interface.

The area of component failure and fault detection has not yet been considered in this project. However, the switching logic of the basic, properly-working controller has similar issues on a less abstract level. For instance, what type of logic should dictate whether our vehicle is in a following mode (ACC or CACC) or in the speed tracking mode (CC)? At first, the answer would seem to be simply the presence of a leading car and it's range to our subject vehicle. i.e., if we detect a lead car in our lane closer than our desired following distance, we should switch to a following mode.

While the above logic may actually work in the most basic sense, it will not produce a very comfortable or safe ride. The transitions between modes may generate braking and acceleration extremes which will be punishing for the driver and unsettling for the vehicle dynamics. We can help alleviate this problem using a transitionary state between the following modes and the CC mode so that for a small period of time (an adjustable parameter) the controller will interpolate between signals from its former state and its future state.

This transitionary state works well for smoothing the performance of our controller, but what if a lead car abruptly cuts in front of our vehicle, causing us to require sudden braking? In this case, if the transitionary state will cost us too much time, we will want leave the transitionary state immediately, or skip it all together and switch directly to the following mode state. The criteria for this decision would be a lower bound of following distance. For this project, we have used a percentage of the desired following distance - bellow which we make a "premature" switch to a following mode.

The above discussion is suggestive of the idea of collision avoidance. Known as Forward Collision Warning (FCW), or Forward Collision Avoidance (FCA), such systems are an extension of C/ACC and are an active area of research. Like C/ACC systems, many complex issues arise. For instance, how much control are we willing to give to an FCA system? Are our sensors trustworthy enough to force the car into a potentially dangerous maximum braking situation? Furthermore, to what degree should we allow the driver to depend on such a system? An FCW system may be more appropriate - providing a warning of a possible collision and allowing the driver to retain control.

In essence, a longitudinal FCA system is simply ACC without any limit to braking ability. For this project, we have not included any FCA or FCW logic, and we limit the maximum controller-requested deceleration to an acceptable 3.5 m/s^2 [4]. However, such systems, especially FCW, would be a natural addition to any existing C/ACC controller.

Another logic issue may arise when the system remains close to a switching limit for extended periods of time. For example, a lead vehicle may be travelling at a velocity very close to our subject vehicle's desired speed. We can add a hysteresis parameter to

avoid rapid switching between states which will reduce the tendency for the controller to rapidly alternate between following and speed tracking modes.

We now see that the design of a comfortable, safe, and robust controller involves more than just a stable control law. Every state or logic that is introduced to the system compounds the difficulty of designing an acceptable C/ACC controller, and the issues discussed above are by no means exhaustive.

4.5.2 Mode Switching Implementation in the Teja Model

Since Teja is a hybrid system design tool, components are defined which contain finite states. Each component can exist in only one state at any time. We can define continuous-time variables in the component, and the dynamics of these variables can be defined independently within each hybrid state (according to $\dot{\mathbf{x}} = f(\mathbf{x}, u, t)$). Teja numerically integrates the continuous time dynamics every time the state is updated. These update transitions are represented in Figure 4.8 by the loops (named **read_n1_2_n2**) above each of the states. Within each transition we can also perform other operations like executing functions, or resetting variables. Each of the transitions shown contain boolean guards which dictate whether the transition is taken. The updates are triggered either by an external time-driven event (for simulation), or a database triggered event (for test car implementation). The five states used in the CC $\rightleftharpoons$ CACC transitions are described as follows:

1. **CC_Control:** This is the default state of the controller and represents the speed tracking mode. During updates, a_{des} is calculated by a function containing the speed

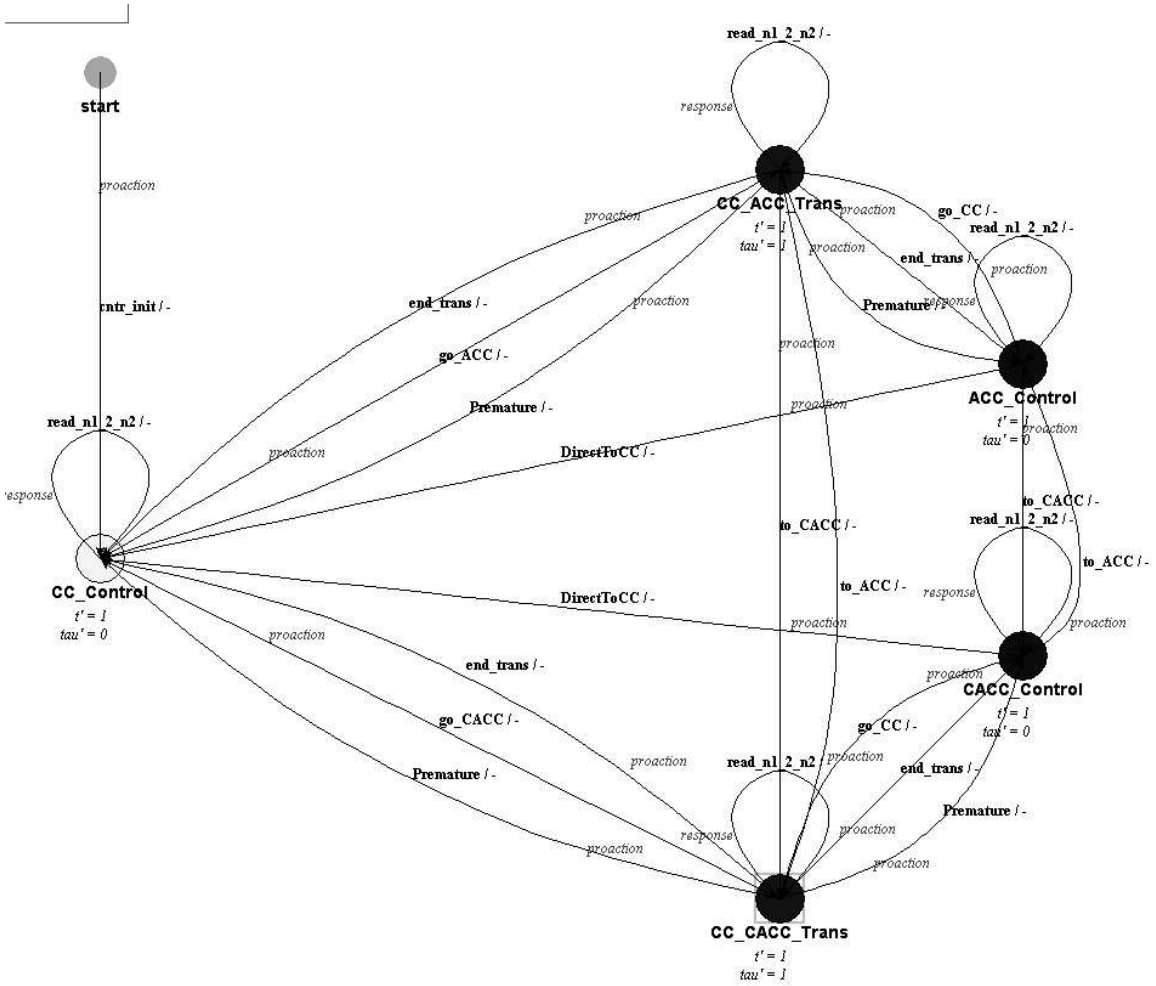

Figure 4.8: Upper Level C/ACC Switching Logic in Teja

tracking mode control law.

2. **CC_ACC_Trans:** This is a transitional state which is occupied only for a preset finite period between the **CC_Control** state and the **ACC_Control** state. During this time a_{des} is calculated by interpolating between a_{des} from the previous state and the a_{des} from the future state.

3. **ACC_Control:** This state represents the ACC following mode control. During up-

dates, a_{des} is calculated according to one of the selected following mode control laws. The signals used for detecting lead car the controller come from the radar only.

4. **CC_CACC_Trans:** This is identical to the **CC_ACC_Trans** state, but is between the **CC_Control** state and the **CACC_Control** state.

5. **CACC_Control:** Identical to the **ACC_Control** state, but range-rate and a_p are derived from the CACC radio instead of the radar.

Currently, a simple switching logic is used between the above states. Switching from the speed tracking mode towards a following mode (via the appropriate transitional state) occurs when a lead vehicle is detected within the desired following distance having a velocity less than the subject vehicle's desired velocity. This logic is based on the assumption that if the lead car's speed is greater than our own, then even if the range is less than desired it will soon increase - thus we avoid an unnecessary transition. This assumption is not entirely correct. It is possible to construct a scenario which would violate this logic and cause the subject vehicle to hit the lead vehicle, but this scenario is only remotely possible and will not be considered.

For the reverse transition, we move from a following mode towards the speed tracking mode via the transitionary state if the lead car leaves our lane, or if its speed is greater than our desired speed and the following distance is greater than a critical distance. This critical distance is set as a percentage of the desired range. The philosophy is that even if the lead car is present and travelling faster than our desired speed, we should remain in a following mode until the lead car has safely moved out of the critical range. Another transition is defined so that we switch directly from a following mode to the CC mode if the

subject car velocity exceeds the desired velocity, and the range is greater than the desired range. We make a direct transition in this case to avoid exceeding the desired velocity.

So called "premature" transitions are also defined from the transitional states to the CC and following mode states. If the controller is moving from the CC state to a following mode, a premature transition is made to the following mode before the end of the defined transition period if the following distance falls below the critical value. Likewise, if the controller is moving towards the CC mode, a premature transition is made to the CC mode if the subject car velocity is exceeded.

The switching logic between CC and CACC modes is nearly identical to that of CC and ACC modes. (Thus, the apparent vertical symmetry of Figure 4.8.) Transitions between the ACC side (top) and the CACC side (bottom) can occur depending on the detection of a leading cooperative vehicle. For example, if a slower-moving vehicle moves in front of our car and is determined to have cooperative capability, we switch toward the CACC following mode. If, while in the **CC_CACC_Trans** (transitionary) state, or in the **CACC_Control** state we loose cooperative communication (or if the CACC signal degrades below an "acceptable" level), the logic will force a transition to the ACC side of the controller. We could develop a more elegant version of this logic so that based on the amount of signal degradation, we transition into a mode which combines both the CACC radio and the radar signals using sensor fusion techniques which optimize the signal quality.

The above logic is not foolproof, but it is stable within the confines of the simulation. It is still very rudimentary, and there is a lot that can be done to improve the functionality of the system but with the expense of added complexity. One such idea is to

utilize the improved data quality of the CACC system.

We will see in later sections that CACC offers an advantage over ACC because of its signal quality. We intuitively expect that the improved sensing ability of the CACC system would facilitate a more efficient and/or safe controller than standard ACC. For instance, CACC may allow a closer following distance. However, we might also utilize CACC's advantages in the switching logic. Consider how we, the human driver, controls the speed of our vehicle. If another car cuts in front of us, we may temporarily allow for a smaller following distance if we sense that the leading car's velocity and acceleration is large enough that we can anticipate the following distance to momentarily increase to the desirable range. Thus, the improved signal reliability of the CACC system should allow for a more "human", or ergonomic system design.

Chapter 5

Simulation Results

Now that we have a model for the subject vehicle dynamics, a simulated lead car interface, a controller structure, and the modelled communication between all three, we are ready to run simulations which test our controller design. A **Logger** component was created in Teja which writes all the pertinent data from the simulation into an output file. Matlab is then used to plot and analyze the results. Also, a parameter output file is written during execution which records the time, date, gains, performance indices, and other valuable parameters which help us tune and design the controller. To change the simulation parameters without having to recompile each run, a data input function was created which parses a data file containing variable names and their values. Then, changing the simulation conditions is simply a matter of editing a data file. While there are certainly many inaccuracies between the models and the actual systems they represent, these simulations are a valid base-line for testing broad qualitative aspects of the controller.

Figure 5.1 shows the output of a sample simulation run. Some of the most useful

data is plotted here. Plot (a) shows the velocities of both the subject and lead vehicle velocities. The subject vehicle begins accelerating at time zero from 5 m/s towards its desired speed of 24 m/s. At 5 seconds, however, the lead vehicle cuts in front of the subject vehicle travelling at a slower speed and with a range of only 5 m. Plot (b) shows that our desired range is closer to 10 m, so the subject vehicle transitions to CACC mode and slows, but the lead vehicle is accelerating, so we quickly achieve our desired range and follow the lead car until about 23.6 seconds. At this point our controller transitions to CC mode so

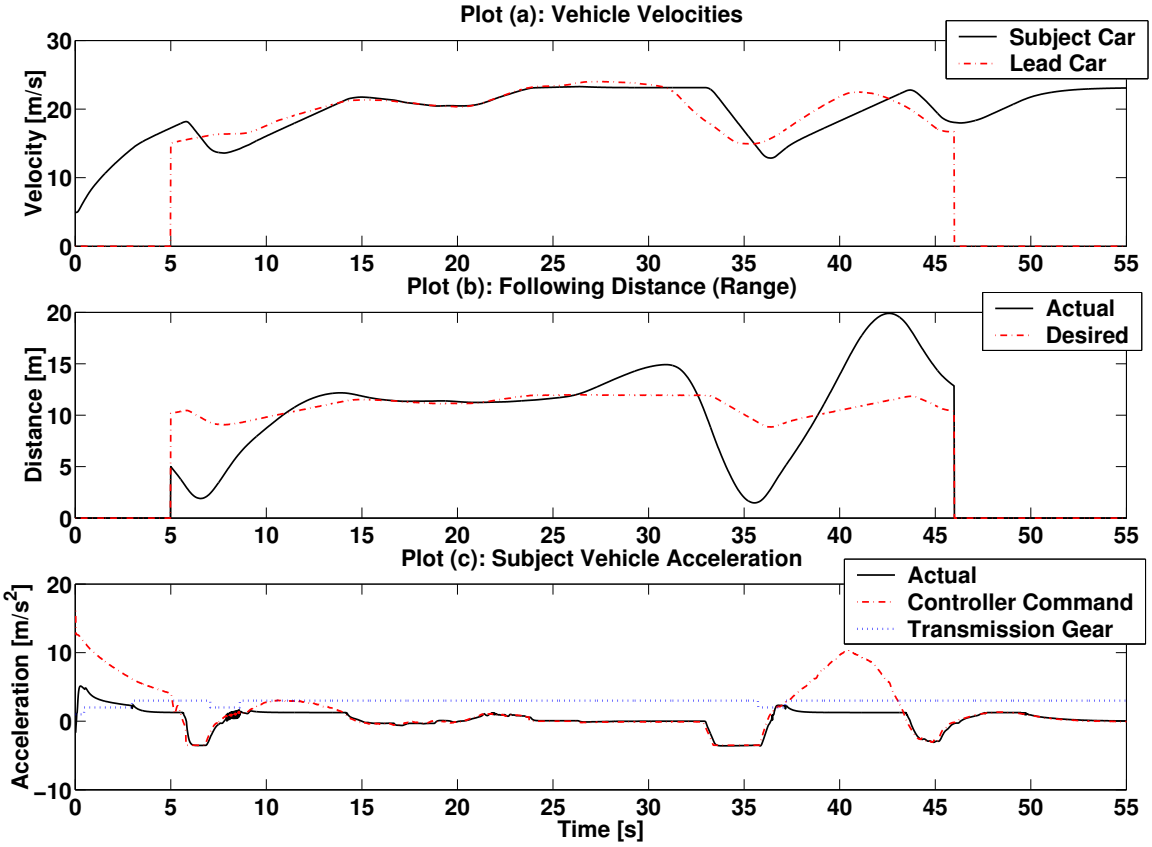

Figure 5.1: Example Simulation Results

that we do not exceed our desired speed. While in CC mode, the actual following distance

greatly increases which is acceptable since we are not in a following mode. Eventually, the lead car decelerates, and at about 33 seconds the controller switches back to CACC mode forcing the subject vehicle to respond to the speed of the lead vehicle. However, the lead vehicle accelerates sharply at around 36 seconds, and the subject vehicle is not able to keep up. We can see in Plot (c) the saturation that occurs in this region where the controller calls for more acceleration than the car can deliver - especially after the $2 \rightarrow 3$ gear shift. This is not a problem, since it only results in a larger following distance. As discussed earlier, when braking is called for we artificially limit the acceleration to -3.5m/s. This is observed between 6 - 7 and 34 - 36 seconds in Plot (c). Finally, at 46 seconds the lead car leaves the subject car's lane, and the controller switches back to the speed tracking CC mode.

We begin our discussion of the simulation results by looking at the signal processing side of the controller. Then we will direct our attention to the control structure itself as we compare various control laws and the two different following modes - ACC and CACC. All the simulations use lead car velocity profiles measured by actual test vehicles from a prior research project. The acceleration profiles were either measured or numerically differentiated and filtered non-causally. The acceleration data is not used in ACC mode, but in CACC mode it is provided to the **CACC_Radio** component (which adds noise and losses) for use by the controller.

5.1 The Performance Index

Before discussing simulation results, we need to develop a way of making quantitative performance judgements about the controller. We can do this by using a cost function

based on the total amount of controller command output and the total error incurred. We would like to minimize both. This performance index is computed by the Teja simulation during ACC and CACC modes:

$$\begin{aligned} J &= \text{Control Norm} + \text{Error Norm} \\ &= \sqrt{\sum a_{des}^2} + \sqrt{\sum (r - r_{des})^2} \end{aligned} \tag{5.1}$$

J is just the sum of the 2-norm of controller effort and controller error. This is not the only valid cost function available for control design. For example, the 1-norm and the infinity-norm are other common system metrics. However this is a suitable and convenient index for our purposes. One advantage of simulation is that specific scenarios can be exactly repeated, and the resulting performance index value allows us to tune gains and filters, compare control laws, etc. while other parameters are held constant.

5.2 Signal Processing Results

In Chapter 3 we discussed the necessity of signal filtering. In Chapter 4 we saw that in a following mode the controller relies on range and range-rate signals from the radar. These are the most important signals. However, lead car acceleration (a_p) is also employed in both the S_1 and S_2 control laws and their variants discussed in Section 4.4.5 (equations 4.47 and 4.48). We expect that using this signal will improve controller performance, but unfortunately it is the most difficult to measure. So while we have filters for all our signals, we concentrate our analysis on the a_p signal filtering since it is the primary motivation for CACC control and the development of alternate control laws.

In the ACC following mode we must synthesize a_p from the radar's range-rate signal and the subject vehicle acceleration. However, numerically differentiating the unfiltered range-rate signal is a noisy process, and adding subject vehicle acceleration (which is also noisy) creates an extremely noisy and inaccurate signal. (Remember, $\ddot{r} = a_p - a$.)

Figure 5.2 compares a noisy synthesized a_p signal (used in ACC mode) on the left to the noisy a_p signal from the CACC radio on the right. The resulting filtered signals are also shown. The accelerometer sensor noise from the lead car (used in CACC) is set equal to the sensor noise from the subject car (used in ACC). Similar filtering is used, with nearly the same lag, yet it is clear that the filtered signal on the right is much more desirable. Of course, we could tune the estimated a_p filter to produce a quieter signal, but this would increase lag. For the a_p signal to be of any use for control, it must be delayed as little as possible.

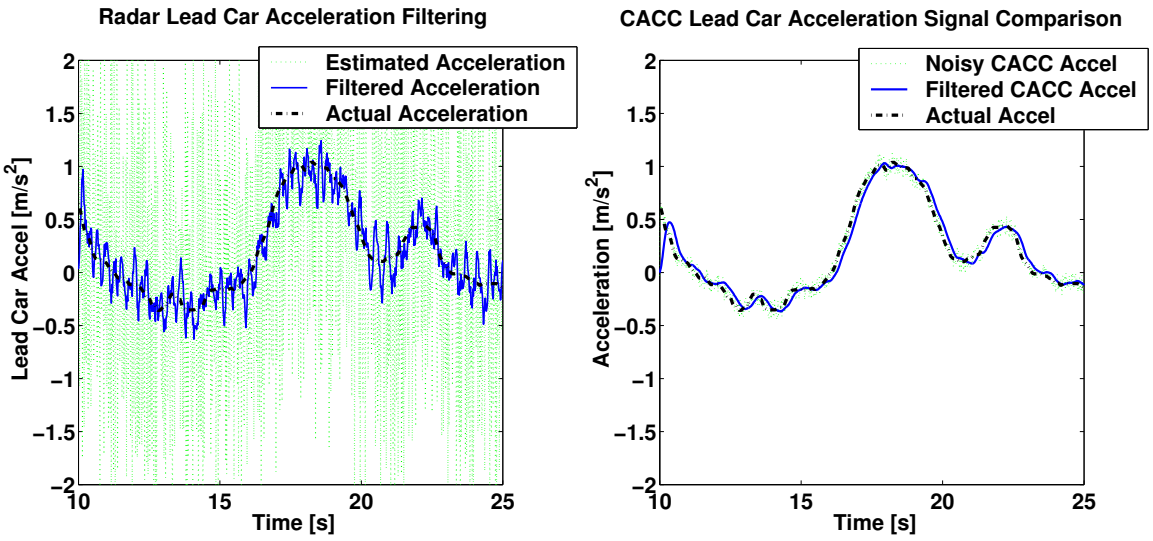

Figure 5.2: Filtered a_p Signals

We can view the tradeoff between filter lag and controller performance by running

simulations using the S_1 control law and holding all other parameters equal while varying the amount of a_p filtering. In Figure 5.3 we have started without any filtering of the estimated a_p signal, then the amount of filtering is gradually increased until the data is smooth but lagging. The legend shows the resulting performance index. These results show the negative effect of signal lag on controller performance since as lag increases, the performance index rises - even beyond the value of the system without filtering. However, we should point out that one shortcoming of our chosen performance index is that it does not capture the undesirable effect of high-frequency control effort chatter. For instance, these results suggest that an estimated a_p filter producing a performance index value close to the non-filtered result would be useless. Consider, however, Figure 5.4 which compares the controller command (a_{des}) of the same following mode scenario with and without estimated a_p filtering. Both simulations produce nearly identical performance indices (the non-filtered run does have a slightly higher control command norm), but the control with filtering is much more desirable. Nevertheless, among the set of controllers using filtering where chatter is greatly reduced, the performance index validates the assertion that excess a_p signal lag hurts performance. Intuitively we would expect this, especially in situations where lead vehicle acceleration changes quickly - as in the case of sudden braking, or "cut-ins". A lag of more than a second could have extremely undesirable results.

While CACC mode allows for a much cleaner a_p signal compared to synthesizing it from the radar, it also provides a cleaner range-rate signal as well. This is because range-rate can be computed as the difference between the lead car's transmitted measured velocity (from the CACC radio) and the subject car's velocity measurement. Since velocity

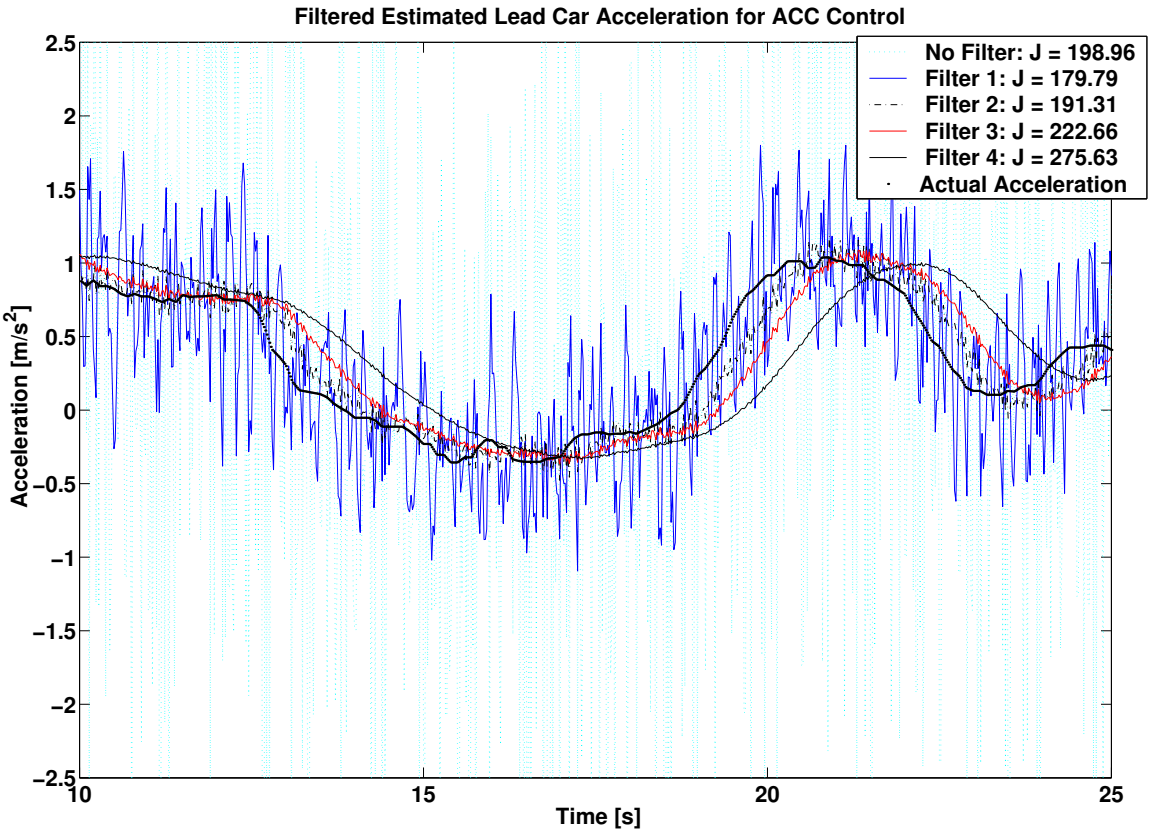

Figure 5.3: A Comparison of Filtered Estimated a_p Signals and the Resulting Controller Performance

measurements are simple and accurate, we expect that barring transmission losses, the CACC range-rate signal is superior to the doppler radar signal.

Figure 5.5 shows a comparison of filtered CACC and ACC range-rate signals. Identical filter settings are used for both. Since we have set the noise of the radar range-rate signal higher than the measurement noise used in CACC (which is a realistic assumption), we expect the filtered CACC signal to be cleaner. However, in this example we have increased the probability of signal loss which explains the occasional spikes in the CACC range-rate plot. During these spikes, the effect of packet loss is to hold the lead car's velocity to its last good value. Subtracting the subject car velocity, produces spikes which

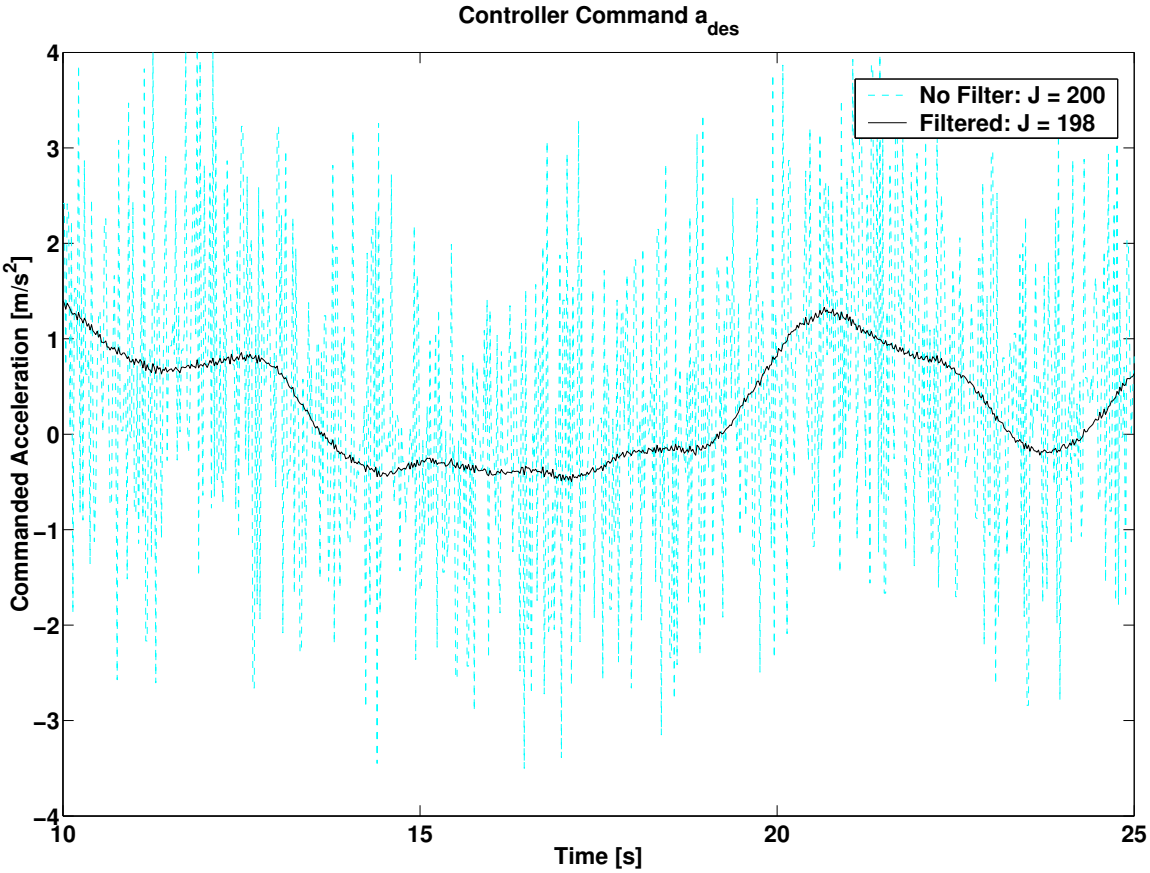

Figure 5.4: A Comparison of Control Command Using Filtered and Non-Filtered Estimated a_p

are smoothed by the filter.

The unfiltered noise of both the ACC and CACC range-rate signals is small compared to the noisy a_p signal. Thus, simulations using control laws which neglect the a_p signal (like the S_3 control law) do not show any substantial improvement due to the CACC range-rate signal's reduced noise. However, real world ACC applications of doppler radar systems involve several complexities other than just constant noise inaccuracies. Since many radar systems acquire multiple targets at various azimuths, care must be taken in identifying what is the lead car, what is a car in an adjacent lane, and what is a stationary object.

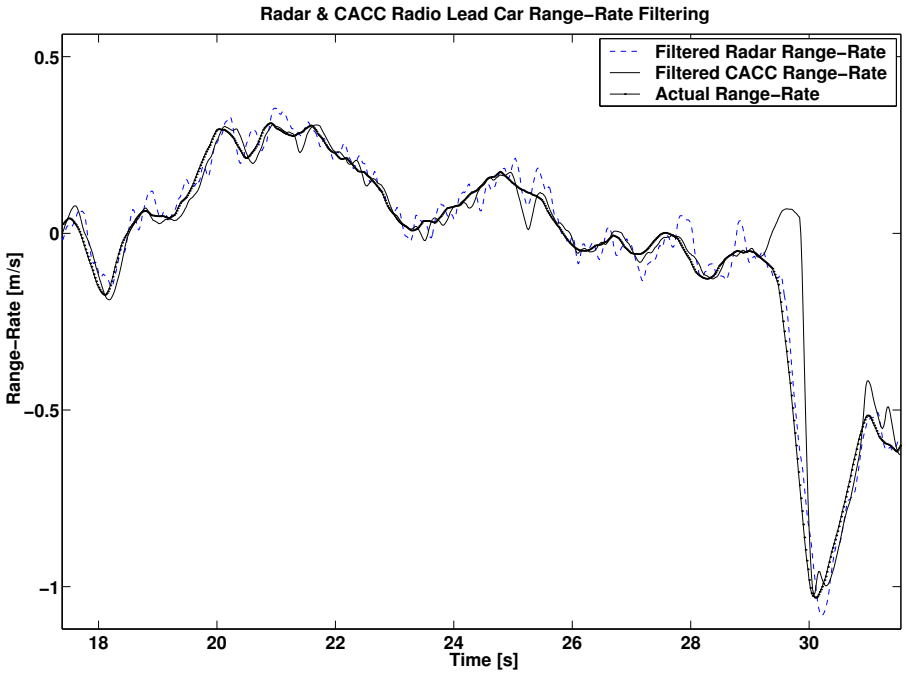

Figure 5.5: A Comparison of Filtered Range-Rate Signals. Velocity Sensor Noise = ±.03 m/s. Radar Range-Rate Noise = ±.15m/s. Packet Loss Probability: 1st = 5%, 2nd = 80%

Although the CACC control structure still depends on the radar for range data, the range-rate signal can be used without having to go through this additional processing. Also,in the case of radar failure or signal ambiguity, the CACC range-rate signal can provide an additional measure of security if sensor fusion techniques are employed.

5.3 Control Law Results

In Section 4.4.4 we noticed the similarity between the S_1 control law and the S_2 control law. In fact, the feedback terms and the a_p feedforward term are identical. When subject vehicle jerk is neglected (as it always is in these simulations) the only difference is the gain of the $\dot{r}_{des}$ term. Thus, we will not consider the S_2 control law in our simulations.

Another interesting result of Chapter 4 was that as a result of our desired following

distance law ($r_{des} = \sigma v + L$), a valid simplified control law can be defined which does not require knowledge of lead vehicle acceleration. This is the S_3 control law discussed in Section 4.4.3. The obvious question is what advantage does the more complex (S_1) control law have which requires a_p? If knowledge of a_p is unnecessary it saves us the trouble of measuring it, and would, in fact, eliminate the major motivation behind CACC control. To answer this question we can look at some simulations which compare the S_1 control law to the S_3 control law.

5.3.1 ACC S_3 Control vs. C/ACC S_1 Control

First, we need to define some common parameters and a scenario that remains constant throughout this simulation set:

- Desired Following Distance: $r_{des} = .3v + 5 \Rightarrow \sigma = .3,\ \ L = 5$
- Subject Car Noise: $v_{noisy} = v \pm .03m/s$, $a_{noisy} = a \pm .1m/s^2$
- Radar Noise: $(r_{noisy})_{radar} = r \pm .03m/s$, $(\dot{r}_{noisy})_{radar} = \dot{r} \pm .15m/s$
- CACC Radio Noise: $(v_p)_{radio} = v_p \pm .03m/s$, $(a_p)_{radio} = a_p \pm .1m/s^2$ Packet loss probabilities: 1st = 5%, 2nd = 70%.
- Cut-In/Out Properties: Lead car cuts in front of subject car with a range of 5 m at time $t = 1$ s. Lead car leaves at $t = 46$ s.

We also need to decide on a lead car velocity profile. For this simulation set we choose the velocity and acceleration profile shown in Figure 5.6.

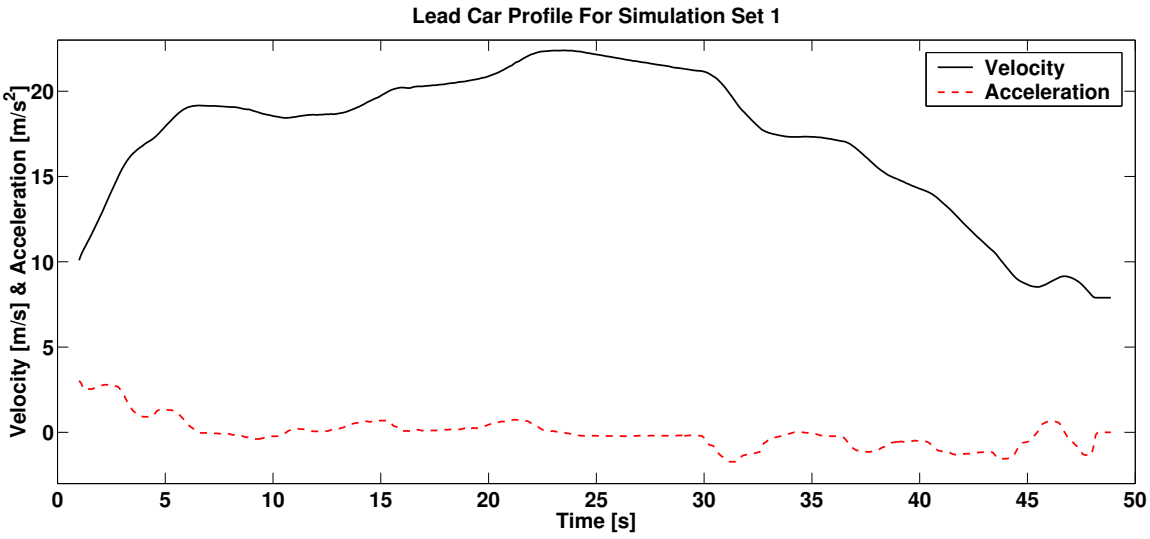

Figure 5.6: Lead Car Profile for Section 5.3.1. Note the small acceleration magnitude and variance.

Figure 5.7 is a comparison of simulations with different control laws. The error in the first 5 seconds is so large that it overwhelms the norm calculations. Therefore, the norms shown are calculated between 5 to 46 seconds where the system is in more of a steady state regime. The first simulation is run using the S_3 control law in ACC mode. After some experimentation, we find that a gain of $K = .4$ produces the best performance. Plot (a) shows the actual and desired following distance results of this simulation during following mode. Next, we tune the S_1 control law in ACC mode and find that the gains $K = .95$ and $\lambda = 1.3$ yield the best performance. However, the results of Plot (b) indicate that although the control effort norm is less, the error norm is greater than when using S_3, and qualitatively we can see that r does not track r_{des} as well as in Plot (a). Finally, in Plot (c) we see that the same control law (S_1 and equivalent gains) in CACC mode performs better than the ACC control, but not significantly better than the S_3 control in ACC mode.

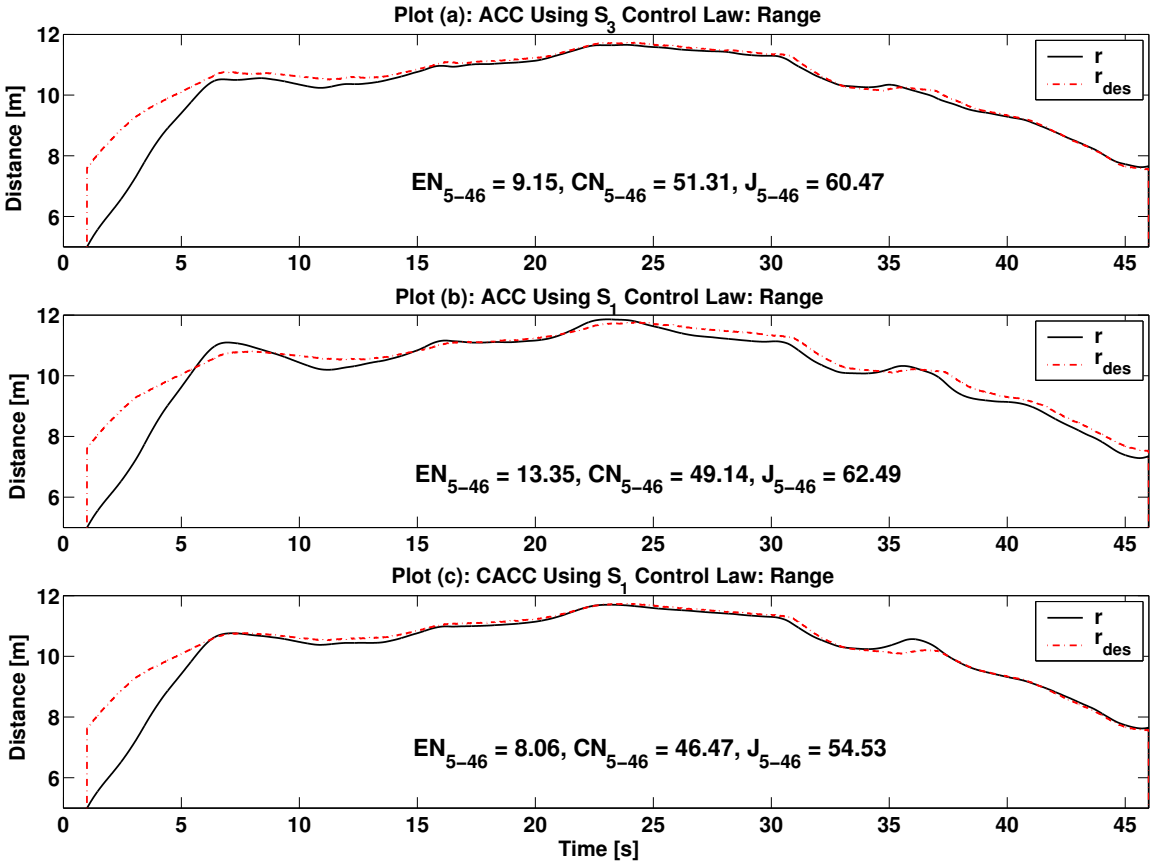

Figure 5.7: Comparison of Control Laws for Section 5.3.1. Note that the addition of a_p does not significantly improve controller performance.

It is easy to explain the poor performance of the ACC S_1 controller exhibited in Plot (b). As we showed in the previous section, the noisy estimated a_p signal is hard to filter without adding excessive lag. In this simulation the erratic a_p signal disturbed the controller and degraded the performance. Figure 5.8 compares the filtered a_p signal used in the ACC controller to the filtered a_p signal used in the CACC controller. Obviously, the smoother and more accurate CACC a_p signal leads to the better performance of Figure 5.7 Plot (c). However, a more interesting result is that the CACC S_1 controller of Part (c) does not provide a major improvement over the S_3 controller of Part (a).

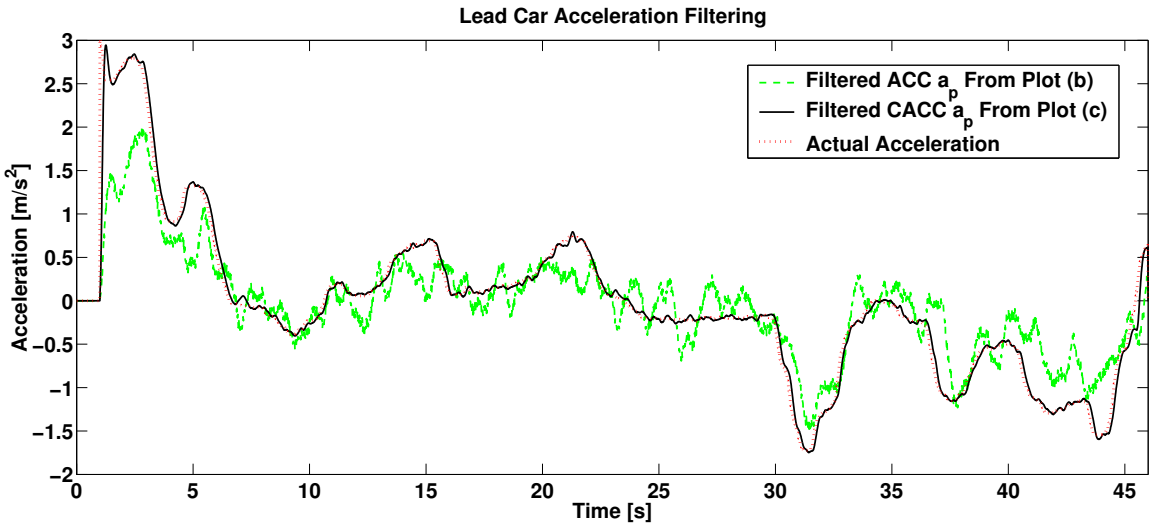

Figure 5.8: Section 5.3.1 Comparison of a_p Signals Used in Figure 5.7 Plots (b) & (c).

The explanation for this behavior lies in the dynamics of the lead car. In Figure 5.6 we notice that the velocity profile is relatively smooth and free from sudden changes in velocity. Accordingly the acceleration profile is small and also smooth. From the controller's perspective, a signal without much magnitude will have a small effect on its performance.

So when the lead car behaves "nicely", the a_p signal is of little benefit to the controller. Furthermore, if the signal is excessively inaccurate, as it was in the simulated ACC mode, the inclusion of a_p cripples the following mode controller performance. Thus, if a_p is of any use at all for a following mode controller, it must be a clean and accurate signal. Nevertheless, the question remains, does using lead car acceleration improve the following mode controller's performance? Perhaps with a more extreme lead car trajectory we can see the advantage of CACC control.

5.3.2 ACC S_3 Control vs. C/ACC S_1 Control: Alternate Lead Car Trajectory

For this simulation set we use the same parameters as in Section 5.3.1, but with the lead car profile shown in Figure 5.9. Between 25 and 45 seconds, this trajectory has a more severe acceleration profile than the previous trajectory.

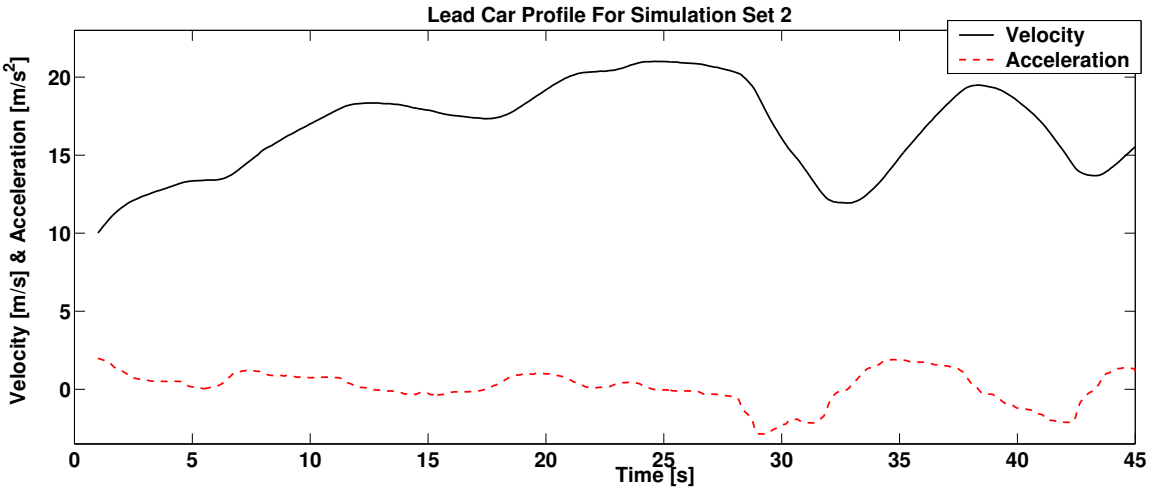

Figure 5.9: Lead Car Profile for Section 5.3.2. Note the larger acceleration magnitude.

We perform the same set of simulations as we did in the first set using identical gain combinations. The results are shown in Figure 5.10 with norm calculations between 7 and 45 seconds. Once again, the introduction of the a_p signal under ACC control in Plot (b) produces a larger error norm, but the control norm is significantly less than the S_3 result, and the combined performance index J is reduced. When we switch to CACC control in Plot (c) we see that both the error and control norm is reduced significantly.

Thus, the advantage of CACC control is best realized when the subject vehicle is forced to respond to larger accelerations of the lead car. We recall our earlier intuition that

if we can know lead vehicle acceleration, we can respond quicker to the lead car's behavior. Our simulation results validate this claim with the caveat that the a_p signal must be clean and without excessive lag. If the signal is too inaccurate, the S_1 controller is actually inferior to the simpler S_3 law.

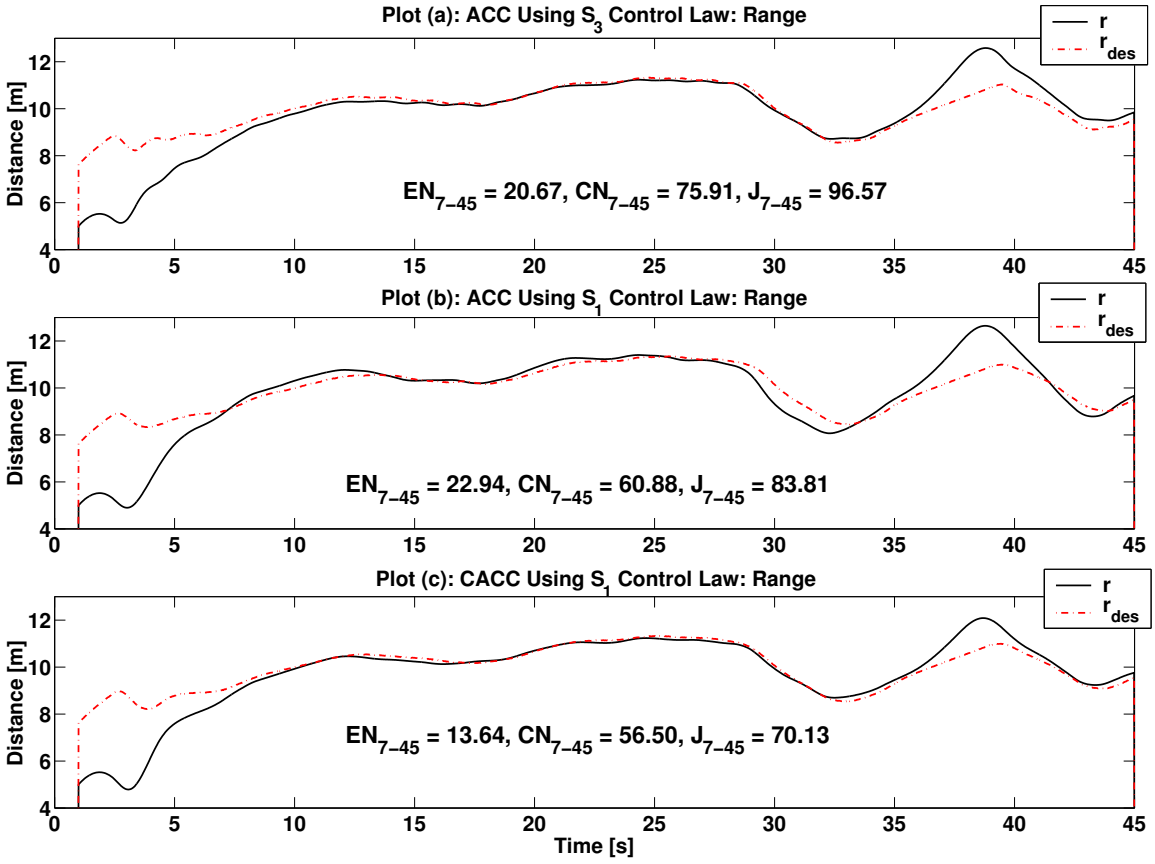

Figure 5.10: Comparison of Control Laws for Section 5.3.2.

5.3.3 The S_1 Variant Control Law

In 4.4.5 we derived equation 4.47 as an variation of the S_1 control law and we showed that it was stable:

$$a_{des} = \frac{1}{1+\lambda\sigma}[KS_1 + \lambda\dot{r} - \ddot{r}_{des}] + \gamma a_p \tag{5.2}$$

In light of the results of the previous two sections, this control law provides an interesting alternative to the standard S_1 controller because it gives us the ability to tune the dependence on the a_p signal directly. We can imagine that depending on the quality or magnitude of the a_p signal we may wish to alter the value of γ to optimize the controller performance.

We investigate this in simulation by using the lead car trajectory of the previous set (Figure 5.9) because of its large acceleration profile. All the gains and parameters are also identical. We begin in ACC mode and compare the S_3 controller (which does not use a_p) to the control law of equation 5.2 with $\gamma = 0$. The results are shown in Figure 5.11. Plot (a) of this figure is identical to Plot (a) of Figure 5.10. Unfortunately, we see that the performance of the S_1 variant control law with $\gamma = 0$ is significantly worse than both the S_3 control law and the standard S_1 control law of Figure 5.10 Plot (b).

This result should not surprise us. We recall the discussion of Section 4.4.3 which demonstrated that if a_p is neglected, the S_1 surface no longer converges to zero, but instead to a boundary we named Δ_1 which is equal to $a_p - \ddot{r}_{des}$. Figure 5.12 demonstrates this behavior. We can see that the S_3 surface converges to zero for the majority of the simulation, while the S_1 surface does not converge when its variant definition is used with $\gamma = 0$. Instead, it approaches a_p. This is because we have artificially removed the a_p signal when, by design, S_1 attempts to cancel the effect of a_p. In fact, a bit of experimentation reveals

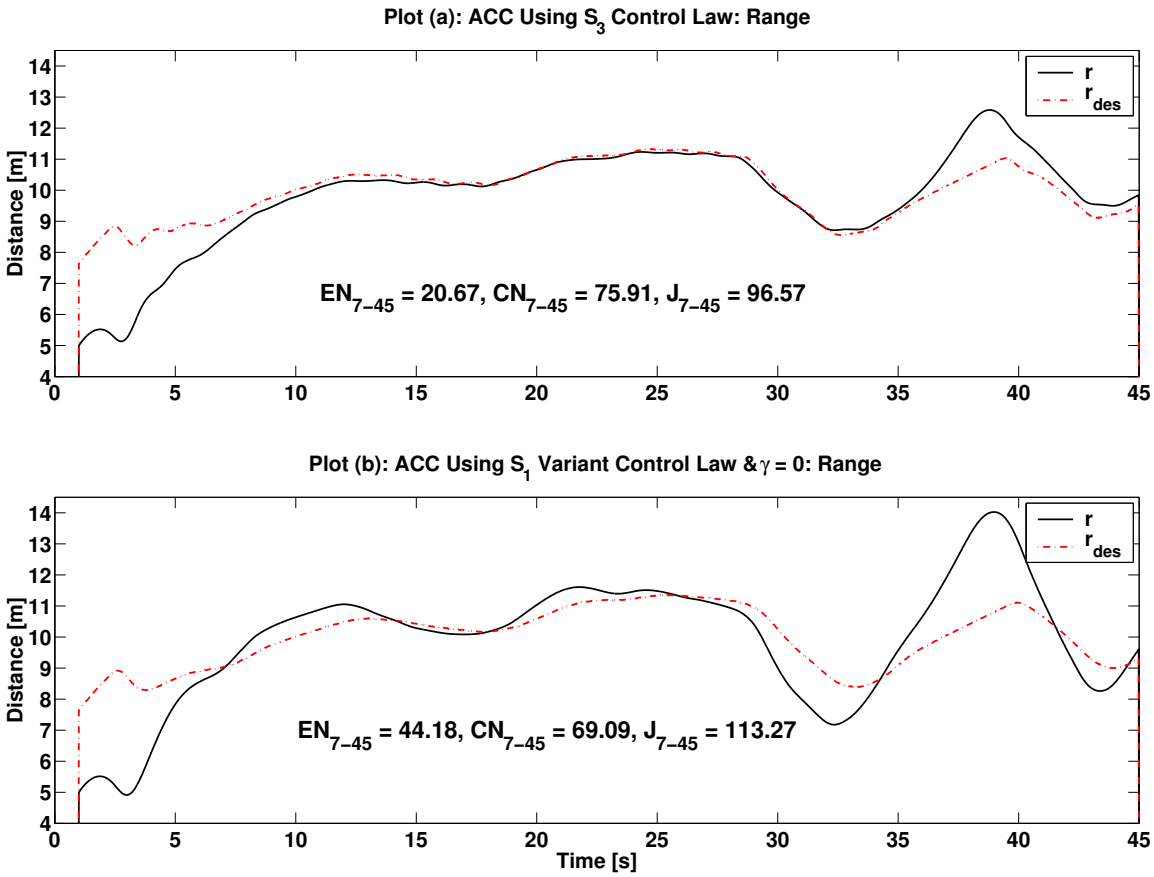

Figure 5.11: Comparison of Control Laws for Section 5.3.3.

that any value of γ less than its equivalent value from the standard S_1 control law ($\gamma_{equ} = 1/(1+\lambda\sigma)$) yields reduced performance. However, what if we wanted to increase the gain of a_p? We pointed out in Section 4.4.3 that unlike the other terms in the standard S_1 control law, the gain of a_p can never be set independently. In fact, it can never be greater than 1. Figure 5.13 Plot (b) shows the results of using the S_1 variant control law with $\gamma = 1.2$ in ACC mode. The same gains are used from the prior simulation sets. The performance indices of the S_1 variant control law is shown to be superior to the performance of the standard S_1 control law in Plot (a) (which is identical to Plot (b) of 5.10). Note the most

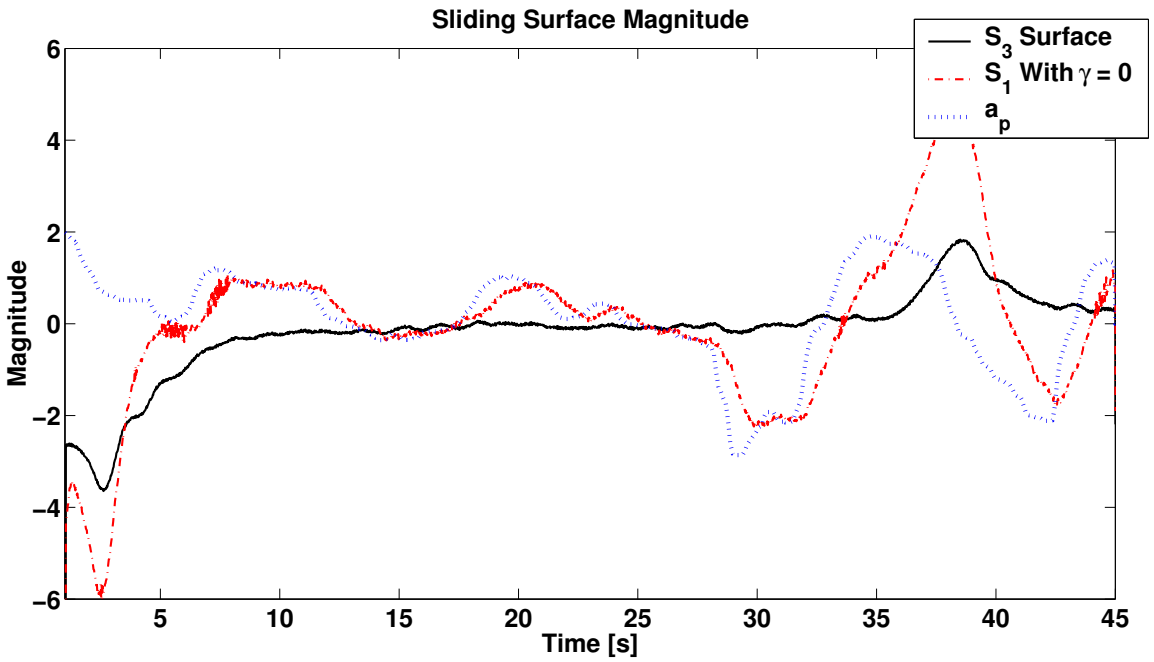

Figure 5.12: Surface Behavior of Section 5.3.3 Simulation Set. Note that S_3 converges to zero, but S_1 converges to a_p when the S_1 variant control law is used with $\gamma = 0$.

drastic improvement is in the final third of the simulation where the lead car acceleration is greatest.

We repeat this comparison in CACC mode with another simulation shown in Figure 5.14. This time we see less of an overall improvement than we did in the ACC mode. One possible justification for this disparity is that the CACC results are close to optimal because of the more accurate a_p signal, and there is less room for improvement. However, we notice an interesting effect in the region of highest acceleration between 27 and 30 seconds. While there is less tracking error at the peak distance ($t \approx 39$ s) there is an increase in error in the previous valley ($t \approx 32$ s). There is a smoothing effect on the trajectory because by increasing the gain of a_p we are increasing the amount that the controller "anticipates" the velocity of the lead car.

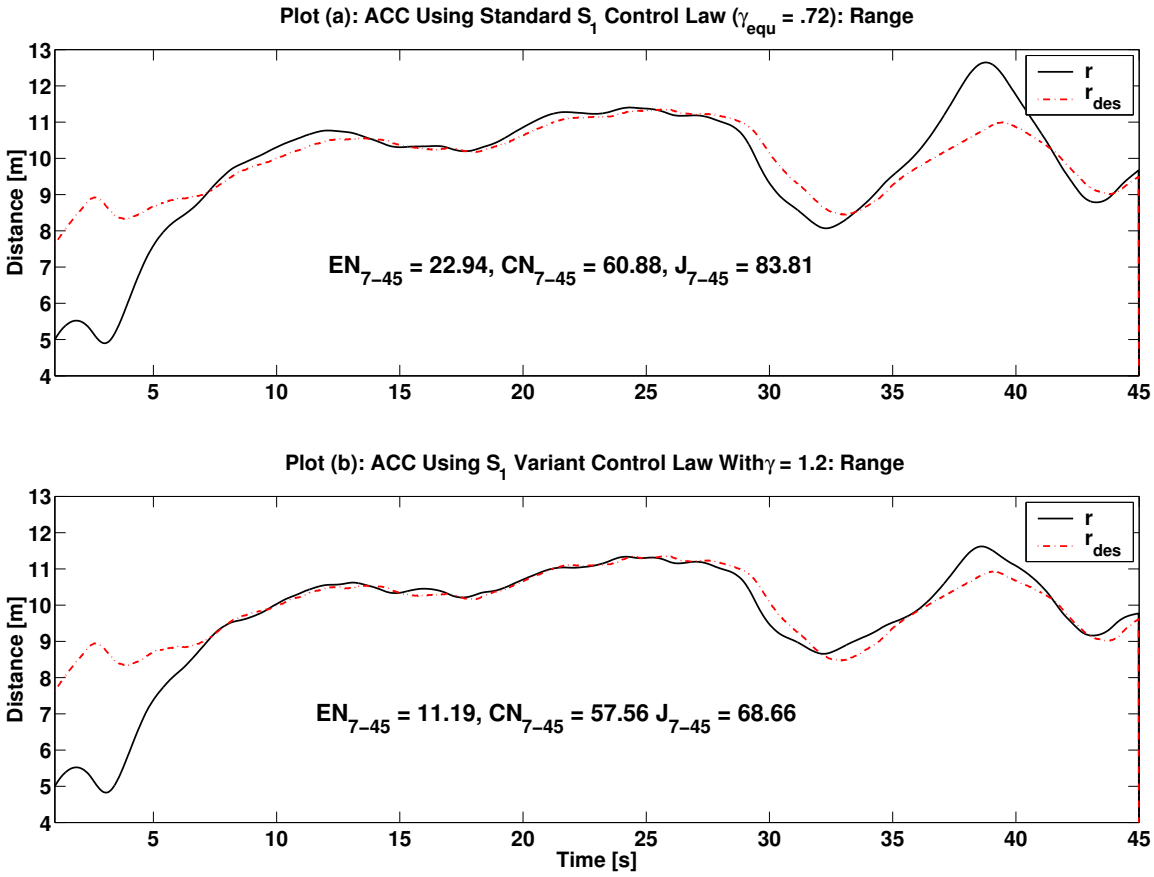

Figure 5.13: Comparison of ACC Control Laws for Section 5.3.3. Note the improved performance of the S_1 variant - especially during the last third of the simulation.

We conclude that the standard S_1 control law provides the best performance when the a_p signal is accurate. However, using the S_1 variant and increasing γ from its equivalent value (γ_{equ}) gives us the freedom to smooth the trajectory when a_p is large and noisy. If we choose to neglect a_p we cannot set $\gamma = 0$, but rather must use the S_3 control law for better performance.

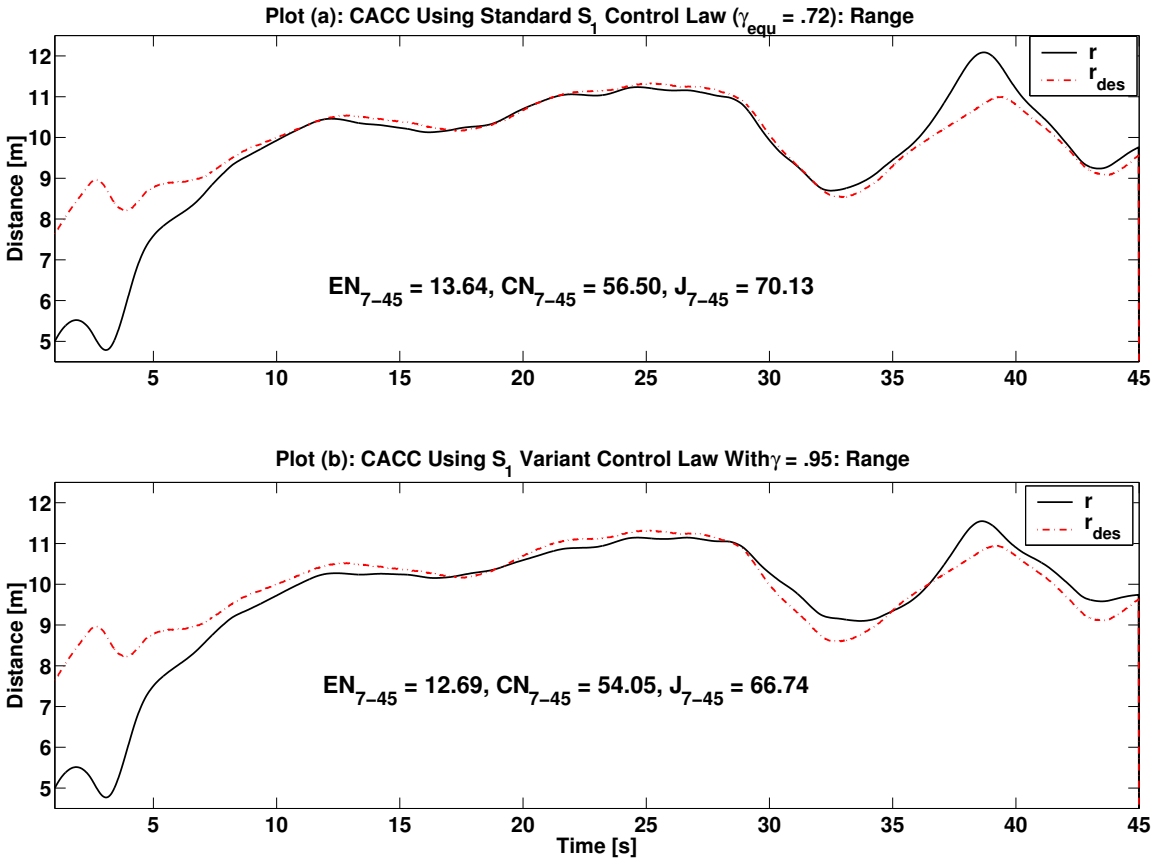

Figure 5.14: Comparison of CACC Control Laws for Section 5.3.3. Note less overall improvement than in ACC mode.

5.4 Mode Switching Results

Finally, we would like to consider the mode switching logic that is designed into the upper level controller. As discussed in Section 4.5 the controller must contain a stable switching logic between the speed tracking CC mode and the 2 following modes (ACC and CACC). Additional transitional states are also added which smooth the operation of the controller. Figure 5.15 is a close-up view of Figure 4.8 and shows only the CACC branch of the controller. Since the logic is primarily the same between the two branches, we will concentrate our discussion on the CC $\leftrightarrow$ CACC transitions shown in Figure 5.15. The

design of this type of state machine logic can become extremely complex, and is not the concentration of this project. Nonetheless, we will view some simulation examples which demonstrate the shortcomings of a basic switching logic structure, and suggest possible improvements.

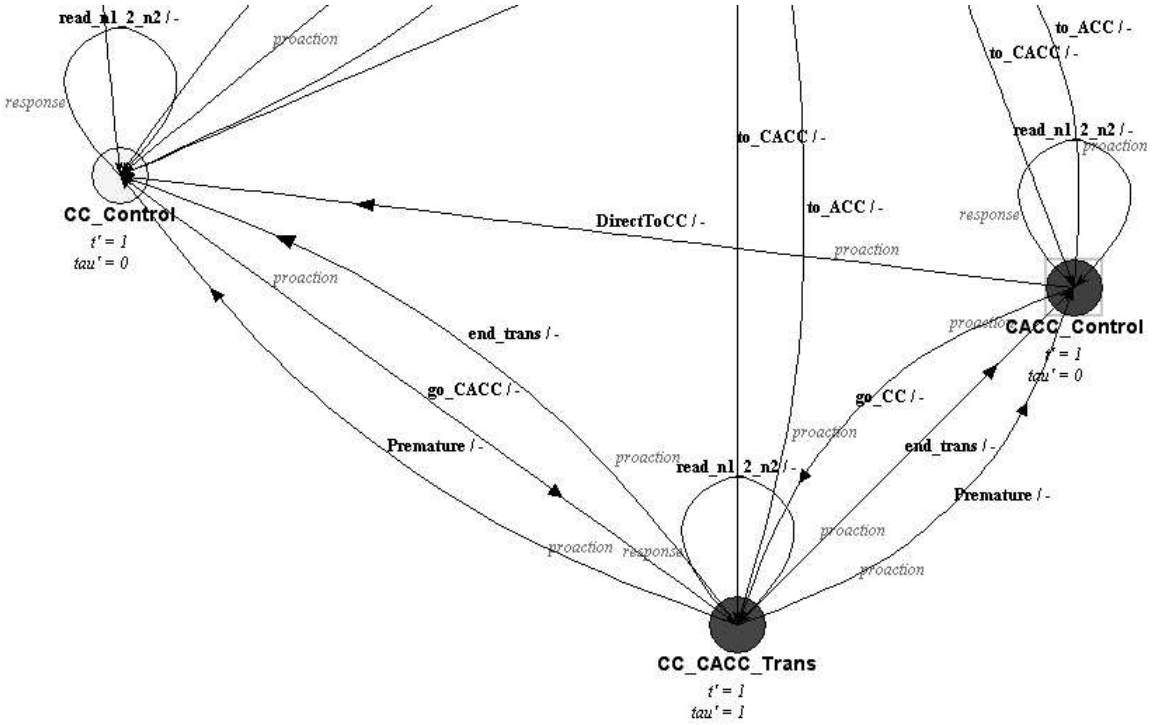

Figure 5.15: The Upper Level Controller Finite State Machine in Teja. (Only CC ↔ CACC branch shown.)

5.4.1 Mode Switching Simulation 1

The Teja model has been programmed so that each time a simulation is run, a trace of the state transition is printed to the command line. A typical run using the identical scenario from the previous section outputs the following stream of statements to the command line:

```
---------------------------------------

     ->Transition towards CACC-> 1.01
     ->CACC-> 3.01

     End following at 45.00 seconds.
     <-Transition<- towards CC 45.01
     <-CC<- 49.01
     Simulation complete...

---------------------------------------
```

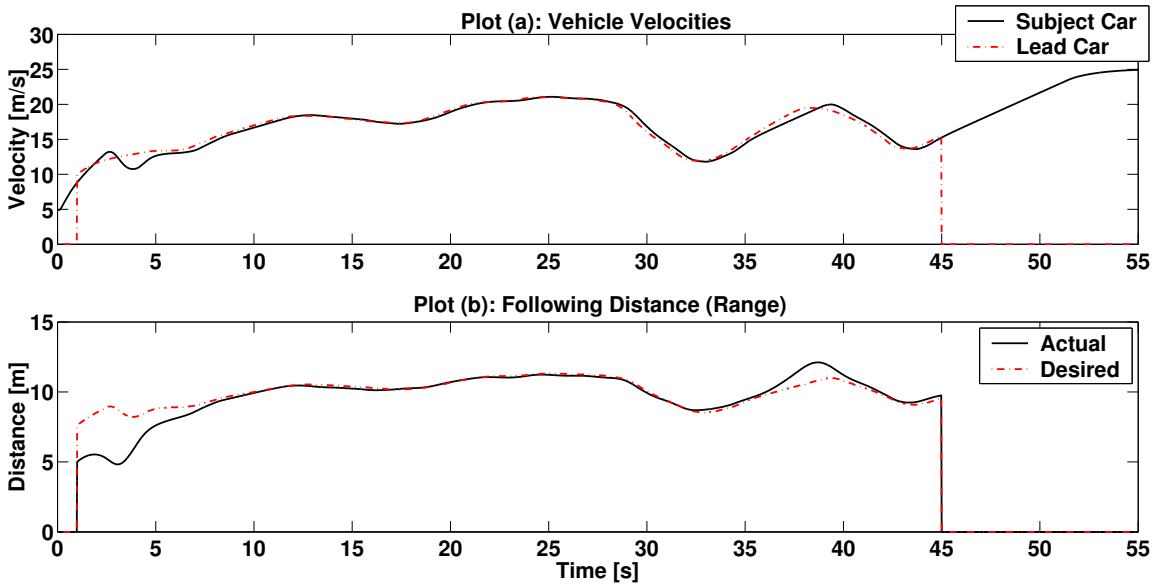

Figure 5.16: Velocity and Range from Section 5.4.1.

The velocity and following distance plots of this simulation are shown in Figure 5.16. This is a simple scenario, and the mode switching is not very complex. At time 1.01, the controller begins a transition toward the CACC mode via the **CC_CACC_Trans** state. The transition period is 2 seconds, so at time 3.01, the entire CC-to-CACC transition is complete, and we are in CACC mode (**CACC_Control** state). For this run we have set the critical following distance to 50% of the desired following distance. Note that during this

transition the following range never falls bellow this critical value - thus the full transition period was completed. At 45 seconds, the lead car leaves the subject car's lane, so we begin the transition back towards CC mode. The reverse transitional period is set to twice the forward transitional time, so at 49 seconds the controller is back in CC mode, and the car accelerates to our desired velocity of 25 m/s.

5.4.2 Mode Switching Simulation 2

Let us look at a more complex scenario. We use the same parameters as before, and the same lead car, but this time we set the desired velocity to 20 m/s. We also decrease the following distance at the moment the lead car cuts in front of the subject car from 5 m to 2 m. The trajectories of this simulation are shown in Figure 5.17, and the screen output is as follows:

```
----------------------------------------

     ->Transition towards CACC-> 1.02
     Premature ->CACC-> 1.02

     <-Direct<- towards CC 21.32

     ->Transition towards CACC-> 30.34
     Premature ->CACC-> 31.26

     ** CRASH OCCURRED AT 32.84 SECONDS!!  **

     <-Direct<- towards CC 40.44

     ->Transition towards CACC-> 42.04
     Premature ->CACC-> 42.90

     End following at 45.00 seconds.
     <-Transition<- towards CC 45.02
     <-CC<- 49.02
     Simulation complete...
```

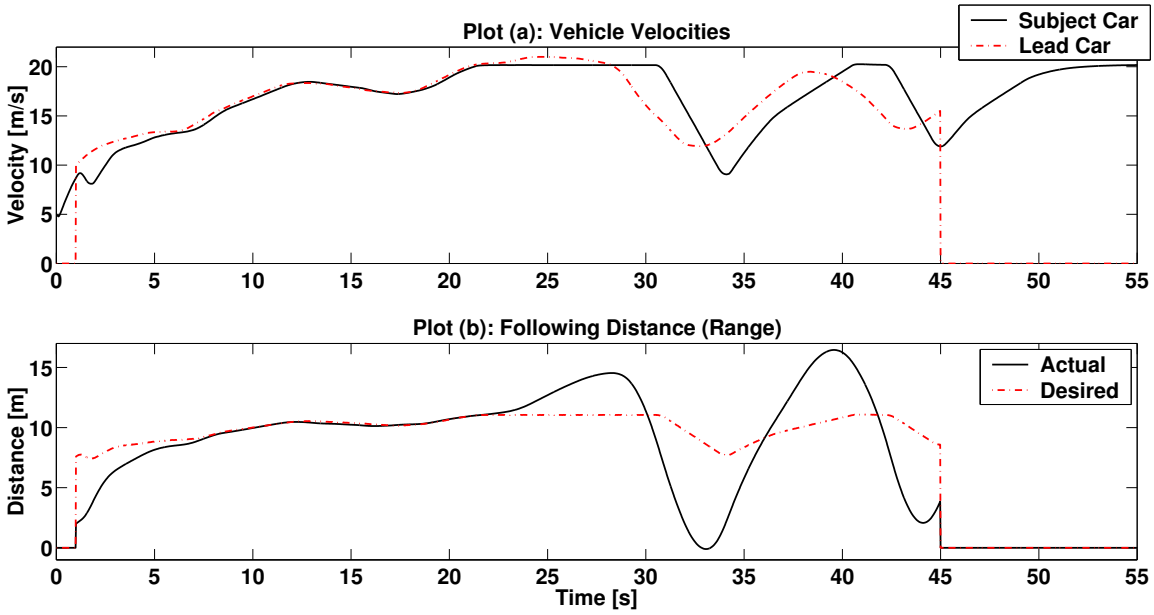

Figure 5.17: Velocity and Range from Section 5.4.2.

Similar to the first simulation, the controller switches towards CACC mode at 1 second, but this time it spends no time in the transitional state, and takes the Premature transition to the CACC mode. This is because the following distance at cut-in time has been set less than the critical range boundary (which is still set at 50% of desired range.) Plot (a) of Figure 5.17 shows that this quick transition results in the sudden decrease of subject vehicle velocity, and prevents the range in Plot (b) from decreasing any further. Thus, we have successfully avoided a crash.

Next, at 21.32 seconds the controller transitions directly to the CC mode because we have begun to exceed our desired speed of 20 m/s. Accordingly, we see the range increase between 21 and 30 seconds because the lead car is travelling faster than our desired rate. So far, this is perfectly acceptable behavior for our controller, but at 30.34 seconds something

interesting happens. The controller recognizes that since the following distance and the lead car's velocity have fallen below the desired levels, it begins a transition to CACC mode. First it goes into the **CC_CACC_Trans** transitional state, but at 32.26 seconds, the range has fallen below the critical level, and a premature transition is taken to CACC mode. But it is too late - the lead car has decelerated very quickly, and the controller has not responded in time to prevent our car from overtaking the lead car. When the actual range falls below 0 m, the simulation indicates that a crash has occurred. The remainder of the simulation is completed without a crash, but notice that when the similar series of transitions is taken between 40 and 45 seconds, the actual range falls dangerously close to zero at 44 seconds.

We have witnessed a major shortcoming of our basic controller switching logic. Since the subject car could track this lead car velocity in the earlier simulations, it should be able to do the same here, even though it switches to the speed tracking mode for some interval. One possible solution is to increase the critical range percentage, but this negates the effect of the transitional state since we would never spend any time in it.

Another possible approach is to design a switching criterion based on the lead vehicle's acceleration. If we switch to a following mode based on a large negative acceleration we could anticipate the lead car's encroachment of our desired following distance and avoid an unsafe following distance. Thus, we can switch to a following mode during a larger range distance if the lead car's deceleration is "too large". We can imagine a profile based on lead car acceleration and current range - the larger the deceleration, the larger the range beneath which we will switch to a following mode. This idea remains to be investigated thoroughly on this project, but we provide the following simulation as evidence of its merits.

5.4.3 Mode Switching Simulation 3

In the previous simulations, switching from the CC mode into a following mode occurs only when a lead vehicle is detected in the subject vehicle's lane travelling faster than the desired speed and with a range less than the desired range. This logic looks like the following:

IF: (Lead Car Detected) AND $(r < r_{des})$ AND $(v_p < v_{des})$,

THEN: GOTO FOLLOWING MODE

We now want to expand this logic to give the controller the ability to "anticipate" the eminent need to react to the lead car. We do so by appending the above logic so it becomes:

IF: [(Lead Car Detected) AND $(r < r_{des})$ AND $(v_p < v_{des})$] OR

[$(a_p < -\alpha$MaxDecel) AND $(r < \beta r_{des})$],

THEN: GOTO FOLLOWING MODE

where $MaxDecel$ is the maximum allowable controller-commanded acceleration (in this case $-3.5m/s^2$), and α and β are tuneable parameters. A proper choice of these parameters may depend on the subject car's capabilities, the desired safety margin, and the quality of the a_p signal. However, it is reasonable to argue that for proper functionality α should be less than 1 and β should be greater than 1. Again, the CACC system's improved a_p signal quality provides a distinct advantage here.

For this simulation, we choose $\alpha = .7$ and $\beta = 1.5$. Running the same scenario as in Section 5.4.2 but with this new switching logic, we get the results shown in Figure 5.18 and the screen output:

```
---------------------------------------

      ->Transition towards CACC-> 1.01
      Premature ->CACC-> 1.01

      <-Direct<- towards CC 21.31

      ->Transition towards CACC-> 28.95
      ->CACC-> 30.95

      End following at 45.00 seconds.
      <-Transition<- towards CC 45.01
      <-CC<- 49.01
      Simulation complete...

---------------------------------------
```

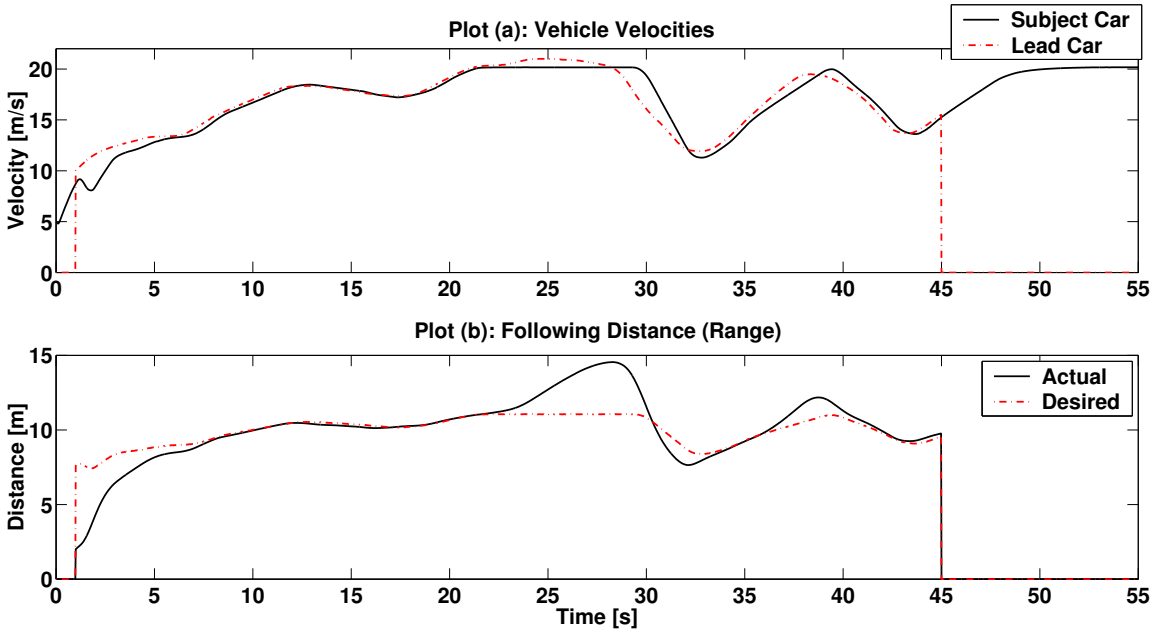

Figure 5.18: Velocity and Range from Section 5.4.3.

Clearly, the added logic prevents a crash by switching towards CACC mode at 28.95 seconds - approximately 1.5 seconds sooner than the previous simulation. Also, compared to Figure 5.17, the actual range remains much closer to the desired range throughout

the remaining following mode interval. We should note, however, that this type of logic is actually a form of Forward Collision Avoidance (FCA). Exactly where we draw the line between FCA logic and standard C/ACC functionality is not yet clear. Perhaps this more advanced switching logic should be placed in a separate FCA mode.

Whenever the set for mode switching is expanded as above, we must make sure that the conditions for remaining in the new mode are satisfied with the new set in mind. Otherwise the state machine will "chatter" between states. Often, a proper logical inverse must be added to the return loops to prevent instability, but this can cause undesirable results in other logic structures. This trickle-down effect increases the overall system complexity, and can quickly render the system intractable. The structure above provides good functionality and addresses some realistic situations that would occur on the road. It appears stable by trial and error simulation, but ultimately, a rigorous hybrid system analysis must be performed on the system.

Chapter 6

Conclusions

ACC and CACC systems continue to be an active area of research, for commercial present-day applications, and for the future goal of highway automation. The complexities of designing these systems lies in developing robust control laws and stable hybrid system logic. For these systems to function optimally, however, proper methods of lead vehicle sensing, and signal processing is essential. Thus, by its design we would expect that CACC provides an advantage over ACC because of its superior signal quality - especially with respect to the lead vehicle acceleration signal a_p.

However, we saw that the choice of a stable control law is not unique, and that it is possible to use a controller which does not require the a_p signal. We also saw that the method of sliding mode control as used in this project is actually a subset of linear feedback and feedforward control. Thus, we suggested an alternate control law which allowed us to alter the gain of the a_p signal independently.

Using models of the vehicle and controller created in the Teja hybrid system mod-

elling environment, we showed the critical role proper signal processing plays in controller performance. However, we also saw that knowledge of lead vehicle acceleration is not required for adequate performance so long as the lead vehicle trajectory is relatively smooth, lacking large acceleration peaks. Nevertheless, when the lead vehicle trajectory is more extreme, the CACC controller with an accurate a_p signal is noticeably superior to the ACC controller. In fact, in some cases ACC control with noisy and inaccurate a_p measurement actually performs worse than the control law which does not require a_p. Interestingly, though, the performance of the ACC controller can be improved by independently tuning the a_p gain using the alternate control law developed in Chapter 4.

Finally, the implementation of a stable, functional, and safe upper level C/ACC controller requires a coherent hybrid system state machine switching logic. This logic may also contain the additional functionality of forward collision warning/avoidance, and contingencies for fault detection. Yet, even in its most basic form, the complexities of the resulting structure require additional efforts to prove stability and to ensure total functionality.

Continuing and Future Work

Work is continuing in an effort to install the upper level controller onto the test vehicle at PATH's Richmond Field Station. The C++ code generated from the Teja models is compiled on the QNX real-time operating system and installed onto the test vehicle's computers. As of writing this report, the code has compiled and executed on the car - albeit with some occasional odd behavior. The next step involves testing and tuning the controller in its "real-world" environment. Also, a more detailed analysis of the hybrid switching logic must be performed to guarantee the desired functionality and stability.

Bibliography

[1] D. Cho and J.K. Hedrick. Automotive powertrain modeling for control. *ASME Journal Dyn. Syst., Meas., Control*, (111):568–576, 1989.

[2] Anouck Girard. Mobies integration of software tools for the simulation and implementation of real-time systems: Control architecture design for the vehicle to vehicle open experimental platform. Technical report, University of California, Berkeley, California, February 2001.

[3] J.K. Hedrick and P.P. Yip. Multiple sliding surface control: Theory and application. *Transactions of the ASME*, (122):586–593, 2000.

[4] Willie D. Jones. Keeping cars from crashing. *IEEE Spectrum*, pages 40–45, September 2001.

[5] Xiao-Yun Lu, J.K. Hedrick, and Michael Drew. ACC/CACC - control design, stability and robust performance. Anchorage, Alaska, May 2002. Proceedings of the American Controls Conference.

Appendix A

Teja Model ACC and CACC Signal Schematics

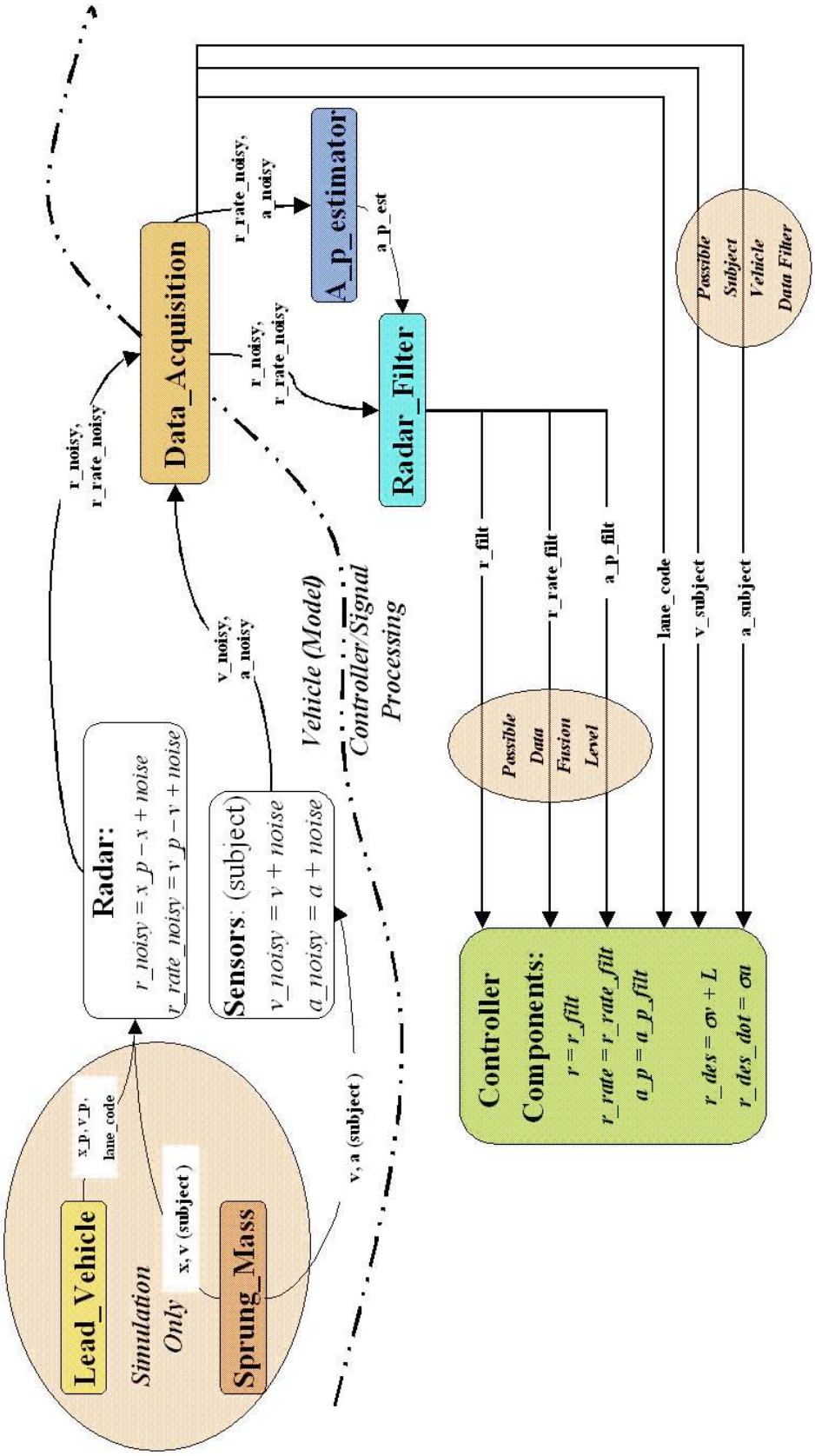

Figure A.1: **ACC** Signal Flow Paths in the Teja Model.

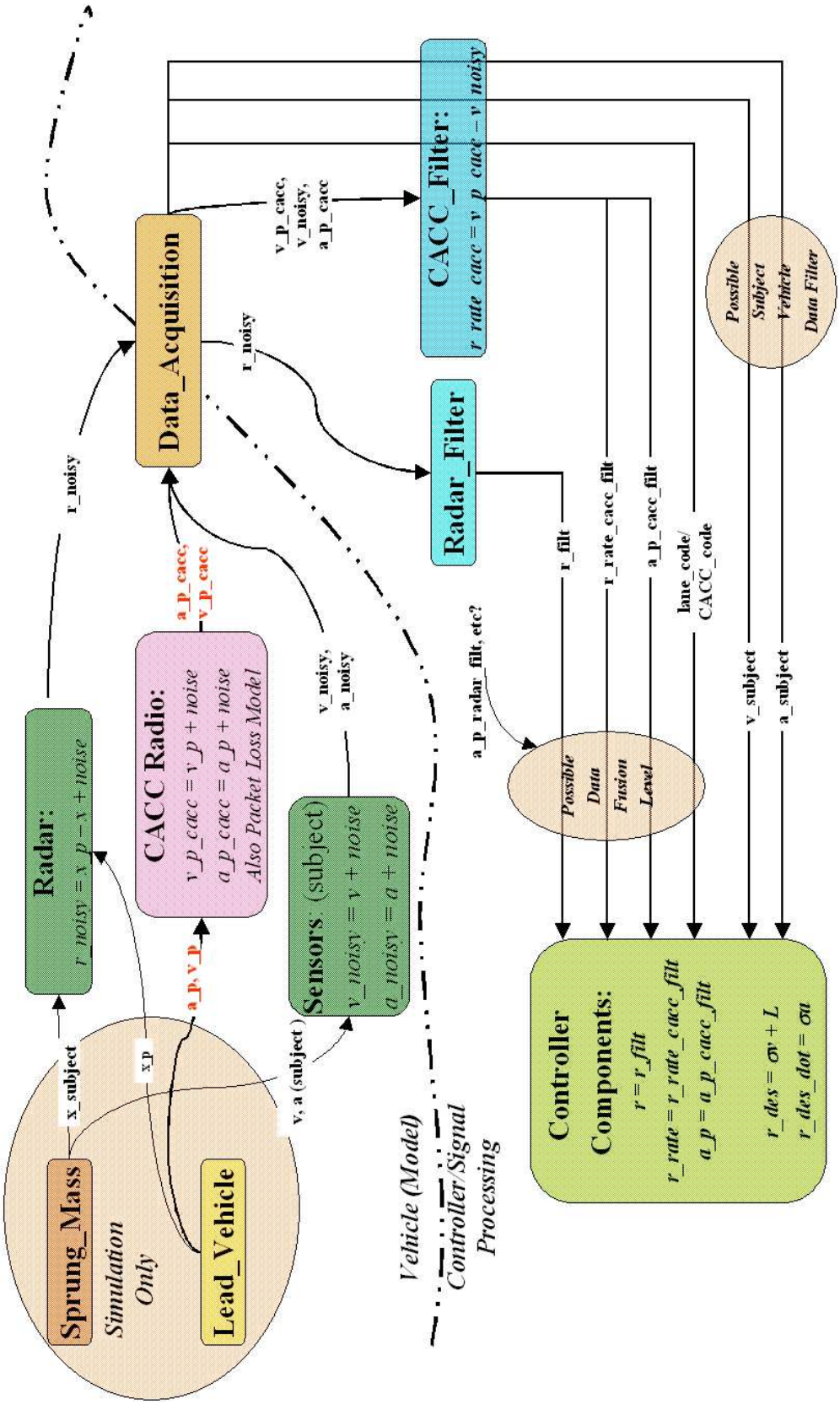

Figure A.2: **CACC** Signal Flow Paths in the Teja Model.