Title

Approximate Method To Determine The Worst Case Performance Of A Nonlinear Dynamical System

Authors

Tongue, Benson H.
Packard, Andrew

Publication Date

1998

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

Approximate Method to Determine the Worst Case Performance of a Nonlinear Dynamical System

Benson H. Tongue, Andrew Packard

California PATH Research Report
UCB-ITS-PRR-98-3

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

Report for MOU 244

January 1998

ISSN 1055-1425

MOU-244: Final Report

Approximate Method to Determine the Worst Case Performance of a Nonlinear Dynamical System

Benson H. Tongue, Andrew Packard

Department of Mechanical Engineering
University of California at Berkeley

California PATH Program

December 1997

Abstract

In this report we present the theoretical development of a method through which a user can evaluate differing platoon control strategies and determine each strategy's worst case behavior under bounded parametric variations. The usefulness of the approach is that a platoon designer can determine how robust her design strategy is in the face of system uncertainties. The general approach is similar to an optimal control design and has applicability to complex, nonlinear systems. The method allows for an arbitrary number of uncertain parameters, unmodeled system components and system inputs. The end result of the iterative procedure is a lower bound for the worst case platoon performance.

keywords: performance, platooning, safety, AHS, collision

Nomenclature

A	vehicle frontal area (m^2)
a_b	braking deceleration (m/s^2)
a_h	tire hysteresis (N/m)
C_D	aerodynamic drag coefficient
$C_{D,un}$	percent uncertainty in aerodynamic drag coefficient
CR_a	correction factor for aerodynamic drag uncertainty bounds
CR_D	correction factor for aerodynamic drag drafting effects
CR_r	correction factor for rolling resistance uncertainty bounds
e_{ff}	drivetrain effectiveness, percent
e_{un}	percent uncertainty in engine effectiveness
F	total force acting on a car (N)
F_a	aerodynamic drag force (N)
F_b	brake force acting at the tire-road interface (N)
$F_{b,m}$	maximum brake force available at the tire-road interface (N)
F_e	engine force acting at the tire-road interface (N)
$F_{e,m}$	maximum engine force available at the tire-road interface (N)
F_g	gravitational force due to road grade (N)
F_r	rolling resistance force (N)
f_r	coefficient of rolling resistance
$f_{r,un}$	percent uncertainty in coefficient of rolling resistance
G_o	road roughness
GR	gear ratio reduction from engine shaft to wheel axle
g	gravitational acceleration (m/s^2)
h	axle height (tire radius, wheelbase) (m)
L_i	length of car i (m)
M	vehicle mass (kg)
M_{un}	percent uncertainty in vehicle mass
$PC(v)$	power curve function for engine torque (N·m)
T_b	brake torque (N·m)
$T_{b,m}$	maximum brake torque (N·m)
T_e	engine torque (N·m)
$T_{e,m}$	maximum engine torque (N·m)
TR	road traction (available coeff. of friction)
v	velocity of vehicle (m/s)
v_w	velocity of wind (m/s)
α	brake input
Δ_i	space between front of car i and back of car $i-1$ (m)
$\Delta_{i,d}$	desired spacing between car i and car $i-1$ (m)

θ	road grade (rad)
μ	braking coefficient of friction
μ_{un}	percent uncertainty in braking coefficient of friction
ρ	density of air (kg/m^3)
τ_b	brake time lag (sec)
τ_e	engine time lag (sec)
ϕ	engine input
ω	engine speed (RPM)

EXECUTIVE SUMMARY

This report is the final one for project MOU-244. This project has been aimed at the problem of platoon performance in the face of uncertainties. The basic ideas that motivated the research stemmed from the realization that, in order to evaluate a platoon's performance, one needs to have some idea of what "good" means in a platooning scenario. Does "good" mean a platoon in which the possibilities of intra-platoon collisions is minimized or does it mean one in which the deviation of each car from it's desired position is minimized. The two goals are not, in general, independent. This reflection on platooning quality led to the concept of a performance index that takes into account the various factors that might influence platoon "goodness," such as acceleration levels, collision frequency, collision magnitude, station-keeping ability, etc.

Once one has an index to evaluate platoons, one is then led to ask whether a platoon that is "good" for nominal conditions, remains "good" if the system changes. Tires can wear, engines can lose effectiveness and a host of other parameters can shift over time. In addition, every possible system dynamics isn't contained in any simulation model and the question arises as to how un-modeled dynamics might affect the overall system. It is this question that this research report addresses. In the following pages, we present an iterative method that works to determine a lower bound for the performance of a general, nonlinear system, when unmodeled dynamics and parametric variations are present. Applying the method to a platoon should let the user know how "bad" the platoon's performance can get and what the parametric variations are that would lead to this performance state.

1 Introduction

The number of vehicles on roads has been steadily increasing over the last several decades, a situation that looks unlikely to change in the near future. In order to alleviate the congestion associated with this growth, various strategies are being proposed with the aim of increasing overall vehicular throughput. One particular approach, part of the IVHS (Intelligent Vehicle Highway Systems) effort, would place vehicles in platoons that are guided by on-board computers. Platoons are defined as a group of closely-spaced vehicles, an example of which is shown in Figure 1. This platoon consists of three vehicles and a lead car with the origin of the Cartesian coordinate frame associated with each vehicle located at the vehicle's rear. Each vehicle is of length L_i and is spaced Δ_i meters behind the preceding vehicle. The lead car, a fictitious vehicle that is used in the simulation as a guide for the entire platoon, is assumed to follow desired accleration (or velocity) trajectories perfectly while each of the following vehicles attempts to maintain a desired spacing of $\Delta_{i,d}$ meters between itself and the preceding vehicle. A platoon is thus a higher level dynamical construct, composed of smaller dynamical units (the individual vehicles). These platoons are envisioned as traveling along one or more special highway lanes that have been specially designated for platoon activity. When a vehicle enters a platoon lane, control of the car is handed over from the driver to the car's on-board computer.

Equipping vehicles with platooning capabilities would theoretically increase throughput on highways since increased tracking accuracy and reduced reaction time over that of human drivers could permit much smaller vehicle to vehicle spacings than is possible for vehicles under human control. Along with this increase in performance, however, must come a high level of safety. Popular acceptance of the platooning approach will certainly be strongly influenced by the public's perception of the platooning activity as a safe one. Control strategies must therefore be developed and tested for all forseeable circumstances that could arise during platooning operations. What one needs, in addition to knowing the performance during nominal operations, is a knowledge of the system's worst case performance. Only by suitably weighing both these factors can a reasonable decision be made as to what platoon configuration is ultimately deployed.

To determine the worst case performance of a platoon over all conditions would require changing each uncertain parameter in every vehicle in combination with every other uncertain parameter, every vehicle make, and and all possible input trajectories, an approach that is clearly computationally prohibitive. To ameliorate this situation, an algorithm will be utilized in this paper which leads to worst case performance scenarios without requiring such extensive simulations.

The basis of our approach was developed by Tierno et al. [6]. In [6], the authors present an approach for determining the worst case performance of general nonlinear control systems. The approach can be viewed as an extension of classical optimal control procedures. This algorithm finds a lower bound for a given nonlinear, noisy system and a desired trajectory

with a feedback controller and a description of the desired performance for a tracking control problem. The method for determining the lower bound is similar to that used in computing a μ lower bound for linear systems.

In this paper, we will clarify the notation and extend the presentation of [6] for a general nonlinear control system with an arbitrary number of uncertain parameters, unmodeled components, and inputs. We will then implement the algorithm to find the inputs and parametric uncertainties that produce a lower bound for the worst case performance for a platoon simulation in order to demonstrate the method's utility in platoon performance evaluation.

2 Problem Statement

We will consider a nonlinear system for which $\mathbf{x} = \{x_1, \ldots, x_{n_x}\}$ is the vector of states, $\boldsymbol{\delta} = \{\delta_1, \ldots, \delta_{n_d}\}$ is a vector of uncertain parameters, $\mathbf{u} = \{u_1, \ldots, u_{n_u}\}$ represents the vector of time-varying inputs signals which perturb the system, $\mathbf{v} = \{v_1, \ldots, v_{n_v}\}$ is the vector of time-varying unmodeled dynamics components, and y is the system performance measure. The number of states, uncertain parameters, inputs, and unmodeled components are n_x, n_d, n_u, and n_v, respectively. The performance measure, y, will be maximized over its 2-norm, $\|y\|_2$ or simply $\|y\|$. The system equations are given by:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \tag{1}$$

$$y = g(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \tag{2}$$

$$\mathbf{z} = \mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \tag{3}$$

Figure 2 shows a schematic representation of this system.

The following restrictions are imposed are imposed on the problem. The uncertain parameters will be real and have a maximum absolute value of 1. The inputs will be bounded in the 2-norm. The 2-induced norm of the unmodeled dynamics block will likewise be bounded. Our performance measure will be maximized over its 2-norm. All signals are assumed to be real. The constraints are as follows:

$$|\delta_i| \leq 1 \quad \text{for } i = 1, \ldots, n_d \tag{4}$$

$$\|u_j\| \leq 1 \quad \text{for } j = 1, \ldots, n_u \tag{5}$$

$$\|z_k\| \leq \|v_k\| \quad \text{for } k = 1, \ldots, n_v \tag{6}$$

Unfortunately, the constraints as formulated above will manifest themselves (in the next section) as final state inequality constraints and complicate the problem beyond a reasonable level. By restricting the disturbances to have norm *equal* to 1, and restricting the

perturbations to be norm preserving (as opposed to norm reducing) the problem becomes

$$
\begin{aligned}
|\delta_i| &\leq 1 \qquad \text{for } i = 1, \ldots, n_d && (7)\\
\|u_j\| &= 1 \qquad \text{for } j = 1, \ldots, n_u && (8)\\
\|z_k\| &= \|v_k\| \qquad \text{for } k = 1, \ldots, n_v && (9)
\end{aligned}
$$

Certainly, if this restricted problem exhibits poor worst-case performance, then so does the original problem. The optimization problem for the worst case performance is summarized as:

$$
\max_{\substack{\delta_i \leq 1 \\ i=1,\ldots,n_d}} \; \max_{\substack{\|u_j\|=1 \\ j=1,\ldots,n_u}} \; \max_{\substack{\|z_k\|=\|v_k\| \\ k=1,\ldots,n_v}} \|y\| \qquad (10)
$$

i.e. , subject to the constraints (1– 3), find the worst performance of the system (maximum 2 norm of y, $\|y\|$).

3 Worst Case Performance Necessary Conditions

The optimization problem given above is, in general, non-convex in nature. We will therefore find a lower bound for the worst case performance by determining the signals which produce a local extremum for this problem. Our problem will be cast in the form of the following theorem:

Theorem 1 [2] *For a dynamical system described by the equations:*

$$
\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x}(0) \text{ given}, \quad t_o \leq t \leq t_f \qquad (11)
$$

a performance index of the form

$$
J = \int_{t_o}^{t_f} L(\mathbf{x}, \mathbf{u}, t) dt \qquad (12)
$$

and restrictions on the final state

$$
\mathbf{G}(\mathbf{x}(t_f)) = \mathbf{c}, \qquad (13)
$$

if the signal $\mathbf{u}_o$ *achieves an extremum of J, then there exists a vector of constants* $\boldsymbol{\mu}$ *and a solution to the two point boundary value problem:*

$$
\begin{aligned}
\dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}_o, \mathbf{u}_o, t) && (14)\\
\dot{\boldsymbol{\lambda}} &= -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}_o}\right)^T \boldsymbol{\lambda} - \left(\frac{\partial L}{\partial \mathbf{x}_o}\right)^T && (15)\\
0 &= \left(\frac{\partial L}{\partial \mathbf{u}_o}\right)^T + \left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}_o}\right)^T \boldsymbol{\lambda} && (16)
\end{aligned}
$$

with the boundary conditions:

$$
\mathbf{x}_o(0) \quad \text{given} \qquad (17)
$$

$$\boldsymbol{\lambda}(t_f) = \left(\frac{\partial \mathbf{G}}{\partial \mathbf{x}_o(t_f)}\right)^T \boldsymbol{\mu}. \tag{18}$$

Furthermore, if these conditions are met we will have

$$\boldsymbol{\lambda}(t_o) = \frac{\partial J}{\partial \mathbf{x}_o(0)}. \tag{19}$$

The next task is to transform the worst-case performance problem (10) into the standard form as given by **Theorem 1**. This requires several steps, including an augmentation of the state-space. The procedure is described below. Define L to be

$$L = \frac{1}{2} y^T y \tag{20}$$

The performance index then becomes

$$J = \int_{t_o}^{t_f} L dt = \frac{1}{2}\|y\|^2 \tag{21}$$

Optimizing J is thus the same as optimizing $\|y\|$

We will now extend the states of the nonlinear system. First, create a state to track the parameters, $\mathbf{x}_\delta = \{x_{\delta_1}, \ldots, x_{\delta_{n_d}}\}$:

$$\dot{x}_{\delta_i} = 0, \quad x_{\delta_i}(t_0) = \delta_i \qquad \text{for } i = 1, \ldots, n_d \tag{22}$$

Next, add a state for the input signals, $\mathbf{x}_u = \{x_{u_1}, \ldots, x_{u_{n_u}}\}$, described by the following differential equations:

$$\dot{x}_{u_j} = \frac{1}{2} u_j^T u_j, \quad x_{u_j}(t_o) = 0 \qquad \text{for } j = 1, \ldots, n_u \tag{23}$$

From Equation 23, $\|u_j\| = 1$ if and only if $x_{u_j}(t_f) = 1/2$ for all j. (Integrate both sides of the differential in Equation 23 to get: $x_{u_j}(t_f) - x_{u_j}(t_o) = \int_{t_o}^{t_f} \dot{x}_{u_j} dt = \int_{t_o}^{t_f} \frac{1}{2} u_j^T u_j \, dt = \frac{1}{2}\|u_j\|^2$.) Lastly, include a state for the unmodeled dynamics, $\mathbf{x}_\Delta = \{x_{\Delta_1}, \ldots, x_{\Delta_{n_v}}\}$, described by the following differential equations:

$$\dot{x}_{\Delta_k} = \frac{1}{2}(z_k^T z_k - v_k^T v_k), \quad x_{\Delta_k}(t_o) = 0 \qquad \text{for } k = 1, \ldots, n_v \tag{24}$$

From the preceding, $\|v_k\| = \|z_k\|$ if and only if $x_{\Delta_k}(t_f) = 0$ for all k. (Integrate both sides of the differential in Equation 24 to get: $x_{\Delta_k}(t_f) - x_{\Delta_k}(t_o) = \int_{t_o}^{t_f} \dot{x}_{u_k} dt = \int_{t_o}^{t_f} \frac{1}{2}(z_k^T z_k - v_k^T v_k) dt = \frac{1}{2}(\|z_k\|^2 - \|v_k\|^2)$.)

We will denote this new system as

$$\dot{\mathbf{X}} \;=\; \mathbf{F}(\mathbf{X}, \mathbf{U}, t) \tag{25}$$

where

$$\mathbf{X} \;=\; \{\mathbf{x}, \mathbf{x}_\delta, \mathbf{x}_u, \mathbf{x}_\Delta\}, \qquad \mathbf{U} \;=\; \{\mathbf{u}, \mathbf{v}\} \tag{26}$$

and $\mathbf{F}$ is given by the following differential equations:

$$\begin{aligned}
\dot{x}_l &= f_l(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \;=\; f_l(\mathbf{X}, \mathbf{U}, t) \qquad \text{for } l = 1, \ldots, n_x && (27)\\
\dot{x}_{\delta_i} &= 0 \qquad \text{for } i = 1, \ldots, n_d && (28)\\
\dot{x}_{u_j} &= \frac{1}{2} u_j^T u_j \qquad \text{for } j = 1, \ldots, n_u && (29)\\
\dot{x}_{\Delta_k} &= \frac{1}{2}(z_k^T z_k - v_k^T v_k) \qquad \text{for } k = 1, \ldots, n_u && (30)
\end{aligned}$$

with the initial conditions:

$$\mathbf{x}(t_o) = \mathbf{x}_o, \quad \mathbf{x}_\delta(t_o) = \boldsymbol{\delta}, \quad \mathbf{x}_u(t_o) = \mathbf{0}, \quad \mathbf{x}_\Delta(t_o) = \mathbf{0} \tag{31}$$

where

$$\begin{aligned}
y &= g(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \;=\; g(\mathbf{X}, \mathbf{U}, t) && (32)\\
\mathbf{z} &= \mathbf{h}(\mathbf{x}, \mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, t) \;=\; \mathbf{h}(\mathbf{X}, \mathbf{U}, t) && (33)
\end{aligned}$$

This system is now in the form of the dynamical system presented in Theorem 1 with the performance index (12)

$$J \;=\; \frac{1}{2}\|y\|^2 \tag{34}$$

and restrictions on the final states (13)

$$\begin{aligned}
x_{u_j}(t_f) &= \frac{1}{2} \qquad \text{for } j = 1, \ldots, n_u && (35)\\
x_{\Delta_k}(t_f) &= 0 \qquad \text{for } k = 1, \ldots, n_v && (36)
\end{aligned}$$

From Theorem 1, if the signals $\mathbf{u}$ and $\mathbf{v}$ and the parameters $\boldsymbol{\delta}$ achieve an extremum of J, then there exists $\boldsymbol{\Lambda} = \{\boldsymbol{\lambda}_x,\ \boldsymbol{\lambda}_\delta,\ \boldsymbol{\lambda}_u,\ \boldsymbol{\lambda}_\Delta\}$ satisfying the two-point boundary problem. The first set of these equations (15) comprise what we will refer to as the cosystem. Substituting $\mathbf{x}_\delta = \boldsymbol{\delta}$ (22), we obtain

$$\begin{aligned}
\dot{\boldsymbol{\Lambda}} &= -\left(\frac{\partial \mathbf{F}}{\partial \mathbf{X}}\right)^T \boldsymbol{\Lambda} - \left(\frac{\partial L}{\partial \mathbf{X}}\right)^T \\
&= -\begin{bmatrix} \frac{\partial \mathbf{f}}{\partial \mathbf{x}} & \frac{\partial \mathbf{f}}{\partial \boldsymbol{\delta}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \mathbf{x}} & \mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \boldsymbol{\delta}} & \mathbf{0} & \mathbf{0} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{\lambda}_x \\ \boldsymbol{\lambda}_\delta \\ \boldsymbol{\lambda}_u \\ \boldsymbol{\lambda}_\Delta \end{bmatrix} - \begin{bmatrix} y^T \frac{\partial g}{\partial \mathbf{x}} & y^T \frac{\partial g}{\partial \boldsymbol{\delta}} & \mathbf{0} & \mathbf{0} \end{bmatrix}^T
\end{aligned} \tag{37}$$

or,

$$\dot{\boldsymbol{\lambda}}_x = -\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T \boldsymbol{\lambda}_x - \left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right)^T \boldsymbol{\lambda}_\Delta - \left(y^T \frac{\partial g}{\partial \mathbf{x}}\right)^T \tag{38}$$

$$= -\begin{bmatrix} \left(\frac{\partial f_1}{\partial x_1}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial x_1}\right)^T \\ \vdots & & \vdots \\ \left(\frac{\partial f_1}{\partial x_{n_x}}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial x_{n_x}}\right)^T \end{bmatrix} \boldsymbol{\lambda}_x - \begin{bmatrix} \left(\frac{\partial h_1}{\partial x_1}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial x_1}\right)^T z_{n_v} \\ \vdots & & \vdots \\ \left(\frac{\partial h_1}{\partial x_{n_x}}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial x_{n_x}}\right)^T z_{n_v} \end{bmatrix} \boldsymbol{\lambda}_\Delta - \begin{bmatrix} \left(\frac{\partial g}{\partial x_1}\right)^T y \\ \vdots \\ \left(\frac{\partial g}{\partial x_{n_x}}\right)^T y \end{bmatrix}$$

$$\dot{\boldsymbol{\lambda}}_\delta = -\left(\frac{\partial \mathbf{f}}{\partial \boldsymbol{\delta}}\right)^T \boldsymbol{\lambda}_x - \left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \boldsymbol{\delta}}\right)^T \boldsymbol{\lambda}_\Delta - \left(y^T \frac{\partial g}{\partial \boldsymbol{\delta}}\right)^T \tag{39}$$

$$= -\begin{bmatrix} \left(\frac{\partial f_1}{\partial \delta_1}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial \delta_1}\right)^T \\ \vdots & & \vdots \\ \left(\frac{\partial f_1}{\partial \delta_{n_d}}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial \delta_{n_d}}\right)^T \end{bmatrix} \boldsymbol{\lambda}_x - \begin{bmatrix} \left(\frac{\partial h_1}{\partial \delta_1}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial \delta_1}\right)^T z_{n_v} \\ \vdots & & \vdots \\ \left(\frac{\partial h_1}{\partial \delta_{n_d}}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial \delta_{n_d}}\right)^T z_{n_v} \end{bmatrix} \boldsymbol{\lambda}_\Delta - \begin{bmatrix} \left(\frac{\partial g}{\partial \delta_1}\right)^T y \\ \vdots \\ \left(\frac{\partial g}{\partial \delta_{n_d}}\right)^T y \end{bmatrix}$$

$$\dot{\boldsymbol{\lambda}}_u = \mathbf{0} \tag{40}$$

$$\dot{\boldsymbol{\lambda}}_\Delta = \mathbf{0} \tag{41}$$

The second set of equations for the two-point boundary problem (16) will be referred to as the alignment conditions:

$$\mathbf{0} = \left(\frac{\partial L}{\partial \mathbf{U}}\right)^T + \left(\frac{\partial \mathbf{F}}{\partial \mathbf{U}}\right)^T \boldsymbol{\Lambda}$$

$$= \begin{bmatrix} \left(\frac{\partial g}{\partial u_1}\right)^T y \\ \vdots \\ \left(\frac{\partial g}{\partial u_{n_u}}\right)^T y \\ \left(\frac{\partial g}{\partial v_1}\right)^T y \\ \vdots \\ \left(\frac{\partial g}{\partial v_{n_v}}\right)^T y \end{bmatrix} + \begin{bmatrix} \left(\frac{\partial f_1}{\partial u_1}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial u_1}\right)^T & \mathbf{0} & \cdots & \mathbf{0} & u_1 & & \mathbf{0} \\ \vdots & & \vdots & \vdots & & \vdots & & \ddots & \\ \left(\frac{\partial f_1}{\partial u_{n_u}}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial u_{n_u}}\right)^T & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & & u_{n_u} \\ \left(\frac{\partial f_1}{\partial v_1}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial v_1}\right)^T & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \vdots & & \vdots & \vdots & & \vdots & \vdots & & \vdots \\ \left(\frac{\partial f_1}{\partial v_{n_v}}\right)^T & \cdots & \left(\frac{\partial f_{n_x}}{\partial v_{n_v}}\right)^T & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}$$

$$\begin{bmatrix} \left(\frac{\partial h_1}{\partial u_1}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial u_1}\right)^T z_{n_v} \\ \vdots & & \vdots \\ \left(\frac{\partial h_1}{\partial u_{n_u}}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial u_{n_u}}\right)^T z_{n_v} \\ \left(\frac{\partial h_1}{\partial v_1}\right)^T z_1 - v_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial v_1}\right)^T z_{n_v} \\ \vdots & \ddots & \vdots \\ \left(\frac{\partial h_1}{\partial v_{n_v}}\right)^T z_1 & \cdots & \left(\frac{\partial h_{n_v}}{\partial v_{n_v}}\right)^T z_{n_v} - v_{n_v} \end{bmatrix} \begin{bmatrix} \boldsymbol{\lambda}_x \\ \boldsymbol{\lambda}_\delta \\ \boldsymbol{\lambda}_u \\ \boldsymbol{\lambda}_\Delta \end{bmatrix} \tag{42}$$

The top half of the above alignment conditions are then

$$\begin{aligned} 0 &= \left(\frac{\partial g}{\partial u_j}\right)^T y + \left(\frac{\partial f_1}{\partial u_j}\right)^T \lambda_{x_1} + \cdots + \left(\frac{\partial f_{n_x}}{\partial u_j}\right)^T \lambda_{x_{n_x}} + u_j \lambda_{u_j} \\ &\quad + \left(\frac{\partial h_1}{\partial u_j}\right)^T z_1 \lambda_{\Delta_1} + \cdots + \left(\frac{\partial h_{n_v}}{\partial u_j}\right)^T z_{n_v} \lambda_{\Delta_{n_v}} \qquad \text{for } j = 1, \ldots, n_u, \end{aligned} \tag{43}$$

and the bottom half of the alignment conditions are

$$\begin{aligned} 0 &= \left(\frac{\partial g}{\partial v_k}\right)^T y + \left(\frac{\partial f_1}{\partial v_k}\right)^T \lambda_{x_1} + \cdots + \left(\frac{\partial f_{n_x}}{\partial v_k}\right)^T \lambda_{x_{n_x}} + \left(\frac{\partial h_1}{\partial v_k}\right)^T z_1 \lambda_{\Delta_1} + \cdots + \left(\frac{\partial h_{k-1}}{\partial v_k}\right)^T z_{k-1} \lambda_{\Delta_{k-1}} \\ &\quad + \left[\left(\frac{\partial h_k}{\partial v_k}\right)^T z_k - v_k\right] \lambda_{\Delta_k} + \left(\frac{\partial h_{k+1}}{\partial v_k}\right)^T z_{k+1} \lambda_{\Delta_{k+1}} + \cdots + \left(\frac{\partial h_{n_v}}{\partial v_k}\right)^T z_{n_v} \lambda_{\Delta_{n_v}} \\ &\quad \text{for } k = 1, \ldots, n_v \end{aligned} \tag{44}$$

The boundary conditions in Equation 18 become

$$\lambda_{x_l}(t_f) = 0 \qquad \text{for } l = 1, \ldots, n_x \tag{45}$$

$$\lambda_{\delta_i}(t_f) = 0 \qquad \text{for } i = 1, \ldots, n_d. \tag{46}$$

The performance index with respect to the parameters (Equation 19, $\mathbf{X}(0) = \{\mathbf{x}_o, \boldsymbol{\delta}, \mathbf{0}, \mathbf{0}\}$) will be at an extremum if the derivative is 0 or if at an endpoint so that the the initial states of the cosystem satisfy

$$\lambda_{\delta_i}(t_o) = 0, \quad \text{or} \quad \begin{cases} \delta_i = -1 \\ \text{and} \\ \lambda_{\delta_i}(t_o) < 0 \end{cases} \quad \text{or} \quad \begin{cases} \delta_i = 1 \\ \text{and} \\ \lambda_{\delta_i}(t_o) > 0 \end{cases} \qquad \text{for } i = 1, \ldots, n_d. \tag{47}$$

4 Algorithm for Lower Bound of Worst Case Performance

Now that we've laid out the two point boundary value problem we have to solve it, a not inconsiderable task given the nonlinear nature of the equations. One method of determining a solution is to consider the new system and cosystem equations as representing two dynamical systems interconnected in a feedback loop (see Figure 3). The output of the new system will be used to calculate the terms of the cosystem. The cosystem will then be simulated backwards in time. Using the alignment conditions, the output of the cosystem is then related to the input of the new system for use in the next simulation. Also, the parameters for the next simulation of the new system are determined from the initial conditions of the cosystem using a power algorithm for finding the lower bound of μ for linear systems. Iteratively applying this sequence of steps, the inputs and parameters are updated to obtain worst case performance measures. When an extremum is reached, we're located at a lower

bound for the worst case performance. The above steps will be discussed in more detail in what follows.

The first step of the algorithm is to simulate the new system (25– 33) using the given initial conditions (31) and inputs that satisfy the constraints of Section 2. Using the relationships for $\mathbf{f}, g,$ and $\mathbf{h}$ and the trajectories y and $\mathbf{z}$ generated from the new system, calculate the terms $\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)^T$, $\left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right)^T$, $\left(y^T \frac{\partial g}{\partial \mathbf{x}}\right)^T$, $\left(\frac{\partial \mathbf{f}}{\partial \boldsymbol{\delta}}\right)^T$, $\left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \boldsymbol{\delta}}\right)^T$, and $\left(y^T \frac{\partial g}{\partial \boldsymbol{\delta}}\right)^T$ in Equation 37 for the cosystem and then simulate the cosystem backwards in time.

Next, compute the terms $\left(\frac{\partial \mathbf{f}}{\partial \mathbf{u}}\right)^T \boldsymbol{\lambda}_x$, $\left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \mathbf{u}}\right)^T$, $\left(y^T \frac{\partial g}{\partial \mathbf{u}}\right)^T$, $\left(\frac{\partial \mathbf{f}}{\partial \mathbf{v}}\right)^T \boldsymbol{\lambda}_x$, $\left(\mathbf{z}^T \frac{\partial \mathbf{h}}{\partial \mathbf{v}}\right)^T$, and $\left(y^T \frac{\partial g}{\partial \mathbf{v}}\right)^T$ in the alignment conditions (42) using the $\boldsymbol{\lambda}_x$ obtained from simulating the cosystem. From the first set of alignment conditions in Equation 43, we can set $u_j \lambda_{u_j}$ equal to a known vector if we use the old values of λ_{Δ_k} and the other terms just calculated. Taking the 2-norm of both sides of this equality, we can solve for u_j and λ_{u_j} using the facts that λ_{u_j} is a scalar and $\|u_j\| = 1$:

$$\begin{aligned}
\lambda_{u_j} &= \pm \Big\| \left(\tfrac{\partial g}{\partial u_j}\right)^T y + \left(\tfrac{\partial f_1}{\partial u_j}\right)^T \lambda_{x_1} + \cdots + \left(\tfrac{\partial f_{n_x}}{\partial u_j}\right)^T \lambda_{x_{n_x}} \\
&\quad + \left(\tfrac{\partial h_1}{\partial u_j}\right)^T z_1 \lambda_{\Delta_1} + \cdots + \left(\tfrac{\partial h_{n_v}}{\partial u_j}\right)^T z_{n_v} \lambda_{\Delta_{n_v}} \Big\| \qquad (48) \\
u_j &= -\Big[\left(\tfrac{\partial g}{\partial u_j}\right)^T y + \left(\tfrac{\partial f_1}{\partial u_j}\right)^T \lambda_{x_1} + \cdots + \left(\tfrac{\partial f_{n_x}}{\partial u_j}\right)^T \lambda_{x_{n_x}} \\
&\quad + \left(\tfrac{\partial h_1}{\partial u_j}\right)^T z_1 \lambda_{\Delta_1} + \cdots + \left(\tfrac{\partial h_{n_v}}{\partial u_j}\right)^T z_{n_v} \lambda_{\Delta_{n_v}} \Big] / \lambda_{u_j} \qquad (49) \\
&\qquad \text{for } j = 1, \ldots, n_u
\end{aligned}$$

From the second set of alignment conditions in Equation 44, we can determine the vector $d = \left[\left(\frac{\partial h_k}{\partial v_k}\right)^T z_k - v_k\right] \lambda_{\Delta_k}$ since the other terms in this equation are known and assuming we use the old values of λ_{Δ_m} for all m except $m = k$. Let this vector d be of infinite magnitude (since λ_{Δ_k} is unknown) passing through the origin with slope $\left[\left(\frac{\partial h_k}{\partial v_k}\right)^T z_k - v_k\right]$. Adding the vector components $\left(\frac{\partial h_k}{\partial v_k}\right)^T z_k$ and $-v_k$ and then multiplying by λ_{Δ_k} will produce a point on d. Since $\|v_k\|$ and $\left(\frac{\partial h_k}{\partial v_k}\right)^T z_k$ are known, we can draw the vector $c = \left(\frac{\partial h_k}{\partial v_k}\right)^T z_k$ and, using this point as a circle center, draw a circle of radius $\|v_k\|$ (the magnitude of v_k). The intersection of this circle with the vector d yields a solution for v_k and λ_{Δ_k}. Defining $c = \{c_1, c_2, \ldots\}$ as the circle center and $d = \{d_1, d_2, \ldots\}$ as the vector:

$$\begin{aligned}
c &= \left(\tfrac{\partial h_k}{\partial v_k}\right)^T z_k \qquad (50) \\
d &= \left(\tfrac{\partial g}{\partial v_k}\right)^T y + \left(\tfrac{\partial f_1}{\partial v_k}\right)^T \lambda_{x_1} + \cdots + \left(\tfrac{\partial f_{n_x}}{\partial v_k}\right)^T \lambda_{x_{n_x}} + \left(\tfrac{\partial h_1}{\partial v_k}\right)^T z_1 \lambda_{\Delta_1} + \cdots + \left(\tfrac{\partial h_{k-1}}{\partial v_k}\right)^T z_{k-1} \lambda_{\Delta_{k-1}} \\
&\quad + 0 + \left(\tfrac{\partial h_{k+1}}{\partial v_k}\right)^T z_{k+1} \lambda_{\Delta_{k+1}} + \cdots + \left(\tfrac{\partial h_{n_v}}{\partial v_k}\right)^T z_{n_v} \lambda_{\Delta_{n_v}}. \qquad (51)
\end{aligned}$$

Figure 4 is a pictorial representation of these equations in two dimensions. The two points of intersection p for the vector and circle are found by solving

$$c'^T c' p^2 - 2 d^T c' p + (d^T d - \|v_k\|^2) = 0 \tag{52}$$

where $c' = c/c_1$. Then, $\lambda_{\Delta_k} = \frac{d_1}{p}$ and $v_k = d/\lambda_{\Delta_k} + c$ (Whether or not one should choose λ_{u_j} positive or negative, or which root to choose for p is not immediately obvious. We have observed through simulation of a simple mass-spring-damper dynamical system that either choice for both variables has approached the expected optimum.)

Finally, update the parameters according to the following rule:

$$\chi_i = \delta_i + a\lambda_{\delta_i}(t_o) \tag{53}$$

$$\delta_i = \begin{cases} -1 & \chi_i < -1 \\ \chi_i & -1 \le \chi_i \le 1 \\ 1 & \chi_i > 1 \end{cases} \qquad \text{for } i = 1, \ldots, n_d \tag{54}$$

where a is the constant one selects to change the rate at which the parameters are updated. (For linear systems, a can be set large to reduce the number of iterations before convergence. For nonlinear systems, however, a large value for a could cause problems when solving for a local extremum.) Repeat the above procedure until the updated values for $\mathbf{u}, \mathbf{v}, \boldsymbol{\delta}, \boldsymbol{\lambda}_u, \boldsymbol{\lambda}_\Delta$ are almost equal to the previous values, *i. e.* convergence has been achieved to within a predetermined error limit.

To summarize, the algorithm steps are:

1. Simulate the new system with initials conditions and current inputs and parameters.
2. Calculate the partial derivative terms for the cosystem using the trajectories generated from the new system.
3. Simulate the cosystem backwards in time with the given final conditions.
4. Update $\mathbf{u}, \mathbf{v}, \boldsymbol{\lambda}_u,$ and $\boldsymbol{\lambda}_\Delta$ using the alignment conditions.
5. Update the parameters using the algorithm rule.
6. Repeat until the updated inputs and parameters are almost equal to their previous values.

5 Algorithm Implementation Issues

To begin, one must choose the initial conditions, inputs, and parameters. We suggest choosing $\mathbf{v} = \mathbf{0}$, $\boldsymbol{\delta} = \mathbf{0}$, and $\boldsymbol{\lambda}_\Delta = \mathbf{0}$. Set the time steps to be used, choose random values for $\mathbf{u}$ at each time step, and normalize $\mathbf{u}$ so that $\|u_j\| = 1$ for $j = 1, \ldots, n_u$.

The new system extended states $\mathbf{x}_u$ and $\mathbf{x}_\Delta$ (Equations 29– 30) do not affect the other states and are not used while implementing the algorithm; they are only included for deriving the cosystem and alignment conditions. Therefore, these states do not need to be simulated. Also, since $\dot{\mathbf{x}}_\delta = \mathbf{0}$ (28), these states do not need to be simulated. As a result, the remaining states are just the states of the original dynamic system (27); the original system can be simulated instead of the entire new system.

Because of the complexity of some nonlinear systems, analytically determining the partial derivatives of the cosystem (37) and alignment conditions (42) may be difficult. To circumvent this problem, one can calculate the linearized model ($\dot{\mathbf{X}} = \mathbf{AX} + \mathbf{BU},\ \mathbf{Y} = \mathbf{CX} + \mathbf{DU}$) of the nonlinear system at each time step. The partials are then easily obtained.

Simulating the cosystem backwards in time, $t = t_f \rightarrow t_o$, is analogous to simulating the following system forward in time, $t = t_o \rightarrow t_f$

$$\dot{\boldsymbol{\lambda}}_x' = \left(\tfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_f - t)\right)^T \boldsymbol{\lambda}_x' + \left(\mathbf{z}(t_f - t)^t \tfrac{\partial \mathbf{h}}{\partial \mathbf{x}}(t_f - t)\right)^T \boldsymbol{\lambda}_\Delta + \left(y(t_f - t)^T \tfrac{\partial q}{\partial \mathbf{x}}(t_f - t)\right)^T \quad (55)$$

$$\dot{\boldsymbol{\lambda}}_\delta' = \left(\tfrac{\partial \mathbf{f}}{\partial \boldsymbol{\delta}}(t_f - t)\right)^T \boldsymbol{\lambda}_x' + \left(\mathbf{z}(t_f - t)^T \tfrac{\partial \mathbf{h}}{\partial \boldsymbol{\delta}}(t_f - t)\right)^T \boldsymbol{\lambda}_\Delta + \left(y(t_f - t)^T \tfrac{\partial q}{\partial \boldsymbol{\delta}}(t_f - t)\right)^T \quad (56)$$

($\dot{\boldsymbol{\lambda}}_u = \mathbf{0}$ and $\dot{\boldsymbol{\lambda}}_\Delta = \mathbf{0}$ do not have to be simulated.) Then, $\boldsymbol{\lambda}_x(t) = \boldsymbol{\lambda}_x'(t_f - t)$ and $\boldsymbol{\lambda}_\delta(t) = \boldsymbol{\lambda}_\delta'(t_f - t)$.

Since the inputs are restricted to the 2-norm with no other requirements, the algorithm may produce seemingly random inputs that might be outside expected frequency ranges, not scaled appropriately, or beyond required limits. By filtering the inputs generated from the alignment conditions before using them in new system, these problems can be avoided. This may involve passing the inputs through saturation blocks and adding a filter prior to the nonlinear system. Adding a filter, however, increases the number of states and complexity of determining the linearized model at every time step. If a linear filter is employed as shown in Figure 5 ($\boldsymbol{\delta}$ will be considered as an input for ease in the derivation) where the linear state equations for the filter are

$$\dot{\mathbf{x}}_f = \mathbf{A}_f \mathbf{x}_f + \mathbf{B}_f \mathbf{u} \quad (57)$$

$$\mathbf{u}_2 = \mathbf{C}_f \mathbf{x}_f + \mathbf{D}_f \mathbf{u}, \quad (58)$$

and the linearized system equations are

$$\dot{\mathbf{x}}_s = \mathbf{A}_s \mathbf{x}_s + \mathbf{B}_s \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u_2} \\ \mathbf{v} \end{bmatrix} = \mathbf{A}_s \mathbf{x}_s + \begin{bmatrix} \mathbf{B}_{sd} & \mathbf{B}_{su} & \mathbf{B}_{sv} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u_2} \\ \mathbf{v} \end{bmatrix} \quad (59)$$

$$\begin{bmatrix} y \\ \mathbf{z} \end{bmatrix} = \mathbf{C}_s \mathbf{x}_s + \mathbf{D}_s \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u_2} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{sy} \\ \mathbf{C}_{sz} \end{bmatrix} \mathbf{x}_s + \begin{bmatrix} \mathbf{D}_{syd} & \mathbf{D}_{syu} & \mathbf{D}_{syv} \\ \mathbf{D}_{szd} & \mathbf{D}_{szu} & \mathbf{D}_{szv} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u_2} \\ \mathbf{v} \end{bmatrix}, \quad (60)$$

then the state equations for the nonlinear system with filter become

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{x}}_s \\ \dot{\mathbf{x}}_f \end{bmatrix} = \begin{bmatrix} \mathbf{A}_s & \mathbf{B}_{su}\mathbf{C}_f \\ \mathbf{0} & \mathbf{A}_f \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_f \end{bmatrix} + \begin{bmatrix} \mathbf{B}_{sd} & \mathbf{B}_{su}\mathbf{D}_f & \mathbf{B}_{sv} \\ \mathbf{0} & \mathbf{B}_f & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u} \\ \mathbf{v} \end{bmatrix} \tag{61}$$

$$\begin{bmatrix} y \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_{sy} & \mathbf{D}_{syu}\mathbf{C}_f \\ \mathbf{C}_{sz} & \mathbf{D}_{szu}\mathbf{C}_f \end{bmatrix} \begin{bmatrix} \mathbf{x}_s \\ \mathbf{x}_f \end{bmatrix} + \begin{bmatrix} \mathbf{D}_{syd} & \mathbf{D}_{syu}\mathbf{D}_f & \mathbf{D}_{syv} \\ \mathbf{D}_{szd} & \mathbf{D}_{szu}\mathbf{D}_f & \mathbf{D}_{szv} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta} \\ \mathbf{u} \\ \mathbf{v} \end{bmatrix}. \tag{62}$$

The partial derivatives can easily obtained, *i. e.*

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{A}_s & \mathbf{B}_{su}\mathbf{C}_f \\ \mathbf{0} & \mathbf{A}_f \end{bmatrix}, \quad \frac{\partial \mathbf{f}}{\partial \boldsymbol{\delta}} = \begin{bmatrix} \mathbf{B}_{sd} \\ \mathbf{0} \end{bmatrix}, \quad \cdots$$

.

6 Results for a Simple Dynamic System

To verify the previous developments, the simple mass-spring-damper dynamic system shown in Figure 6 has been utilized. The equations of motion for the system are given by:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} u \tag{63}$$

$$y = x_1 \tag{64}$$

where x_1 is the mass position, x_2 is the mass velocity, u is a disturbance force input, y is our performance measure – mass position error, and m, k, and b are the mass, spring, and damper constants, respectively. We considered parametric uncertainties in m and k. The parameters chosen for this problem were:

$$\begin{aligned} b &= \text{damping coefficient} &&= 15\ kg/s \\ k &= \text{spring constant} &&= 100\ N/m \\ m &= \text{mass} &&= 10\ kg \\ w_k &= \text{uncertainty weighting for } k &&= 0.30 \\ w_m &= \text{uncertainty weighting for } m &&= 0.30 \\ \omega &= \text{natural frequency of system} &&= 3.16 rad/sec \end{aligned}$$

The high- and low-pass filter specifications were chosen to be:

Filter	Pole (rad/s)	Zero (rad/s)	Gain at 0 rad/s
high-pass speed	$-\omega/100$	-ω/96	0.3125
low-pass grade	$-10 \cdot \omega$	—	1.000

The interconnection of the unmodeled dynamics input and output with the mass-spring damper system is shown in Figure 7. The input signal u passes through a high-pass, first-order linear filter to produce the unmodeled dynamics input z. The unmodeled dynamics output v is filtered through a first-order low-pass filter and adds to the input.

In less than five iterations, $\boldsymbol{\delta} = [\delta_m \; \delta_k]$ reached its final value of $[1 \; -1]$. Figure 8 shows the inputs and output after 25 iterations. These results correspond to a lightly damped system with the inputs acting at the system's natural frequency, inputs which produce large oscillations in the output. The 2-norm of y increased for each iteration, the final value equalling 0.02792. (The 2-norm of y increased after each iteration for the choice of $\boldsymbol{\lambda}_u$ and the root in Equation 52 both negative. For the other choices of $\boldsymbol{\lambda}_u$ and the root in Equation 52, the 2-norm of y approached the same the final value but did not necessarily increase after each iteration.) This result was checked with μ-tools [1], which can be used to determine the lower bound for worst case performance for linear systems; the estimated lower bound for $\|y\|$ was about 0.025.

7 Results for a Vehicular Platoon

In the following example we consider a 4-car platoon with lead car. The performance measure we use for this system is the square root of the sum of the squared spacing errors. The values of the parameters can be found in Appendix B.1 The parametric uncertainties for the system are M_{un}, μ_{un}, e_{un}, $C_{D,un}$, and $f_{r,un}$ for each car, a total of 20 uncertainties. The inputs are v_w, θ, TR, l_a, and, for each car, sensor or communication noise for spacing, velocity, acceleration, previous car velocity, previous car acceleration, lead car velocity, and lead car acceleration, giving 32 total inputs. The inputs were bounded in magnitude and filtered with first-order linear filters according to the specifications listed in Appendix B.3. We did not consider unmodeled dynamics for this implementation. The time step size for all times was 0.01 sec. The weighting for updating parameters was set to 1.

In the simulation each car was represented by a lumped mass that moves in a longitudinal manner under the influence of input forces, both internal (engine/brake) and external road friction, air drag, etc). Each vehicle has a given length and the platoon objective is to maintain a one meter spacing between each vehicle. The controller [4] uses the velocity and acceleration of the lead vehicle as well as position, velocity and acceleration of the preceding vehicle to produce a control input for the controlled car.

After 25 iterations, the maximum value of $\|y\|$ was found to be 9.425. Figure 10 shows the inputs which produced the worst performance, and Figure 11 show the resulting velocity and spacing responses, respectively. The corresponding parametric uncertainties are given in the following table:

Parameter	Car 1	Car 2	Car 3	Car 4
e_{un}	1	-1	1	-1
μ_{un}	-1	1	-1	1
M_{un}	-1	1	-1	1
$C_{D,un}$	-1	1	-1	1
$f_{r,un}$	-1	1	-1	1

As can be seen from the spacing response, the platoon performance deteriorates markedly. The spacing error between Cars 2 and 3 reaches -1 meters near the end of the simulation, indicating a collision. Recall that the nominal spacing was 1 meter for this simulation and thus a spacing error of -1 meters implies zero spacing between the vehicles. The spacing error continues to decrease because this particular simulation did not include a collision module (which would forbid cars to pass through each other) since the objective was simply to validate the worst case algorithm.

The spacings between cars 1 and 2 and between cars 3 and 4 becomes large because the algorithm chose the parametric uncertainties such that cars 1 and 3 would accelerate most easily ($M_{un} = -1,\ e_{un} = 1,\ C_{D,un} = -1,\ f_{r,un} = -1$) and cars 2 and 4 would accelerate most poorly ($M_{un} = 1,\ e_{un} = -1,\ C_{D,un} = 1,\ f_{r,un} = 1$). The final parameters were driven to their limits, a result that makes sense. One would expect the worst case performance to correspond to some cars being as "good" as possible while others in the platoon are as "bad" as possible. An example that illustrates this would be to picture a 2 car platoon faced with an emergency braking maneuver in which the first vehicle has superb braking while the second has very poor brakes. In this case the first vehicle would brake strongly and the second will plow into the first, due to its inability to decelerate as fast as the vehicle in front.

The foregoing observation motivates a topic for further research. It would seem that the outcome of an extreme maneuver will depend strongly on the capabilities of the vehicles in the platoon and on their particular order within the platoon. It would thus be useful to investigate the effect of ordering within a platoon in order to determine a strategy by which platoons can most effectively be constructed. Certainly the vehicles making up the platoon will have a range of capabilities, just as current cars do. Rather than having a new car that wishes to join the platoon simply merge from the rear, it may be desirable to place the vehicle elsewhere in the platoon as a function of that car's capabilities and the capabilities of the pre-existing platoon members.

The simulation results show that the current controller performance is unacceptable under worst case platoon operation. Although the lead car acceleration commands were under $0.2g$, spacing errors reached 3 meters and two cars collided during the simulation. This would suggest the need for further controller design that ensures robustness against the parametric variations produced by the worst case algorithm. It should be noted that this example was chosen to allow the worst case algorithm to easily demonstrate difficulties.

Other simulations would doubtless show less extreme variations. A serious study would involve many simulations over many road conditions. The results of such a study would show the same basic result as that already demonstrated - a calculation of parameters that cause the "worst" response and an indication of the degree of "badness."

8 Conclusions

What is seen from the preceding examples is that the worst case performance algorithm can successfully supply the platoon designer with a useful tool with which to evaluate platoon controllers. By identifying the worst combination of system parameters, it allows the designer to see just how bad the performance of the platoon can be and, when used in conjunction with knowledgable controller design, permits one to create the platoon that best meets given performance specifications.

In order to improve performance, one would want to examine which platoon parameters are most influential in determining the overall performance, creating a sensitivity measure for each. Once this was done, one would know which parameters should be most accurately determined and controlled and which could be safely ignored. Once one knew the important parameter, and also determined the parametric variations that each would involve, one could approach the controller design problem from a μ control design perspective, a control approach that specifically targets robustness issues.

It should be stressed that this procedure should be viewed as part of an overall platoon design process. Although it will provide a measure of the "worst" performance, this performance will not be encountered often. Most of the time the platoon will operate within a normal envelope of parametric variations and thus the actual performance will not be far off from the ideal. Thus, in order to logically deal with the information given by the worst case algorithm, one has to put some weighting on how large a performance variation is allowable over a given time interval of operation. These are questions that must be answered by transportation management authorities.

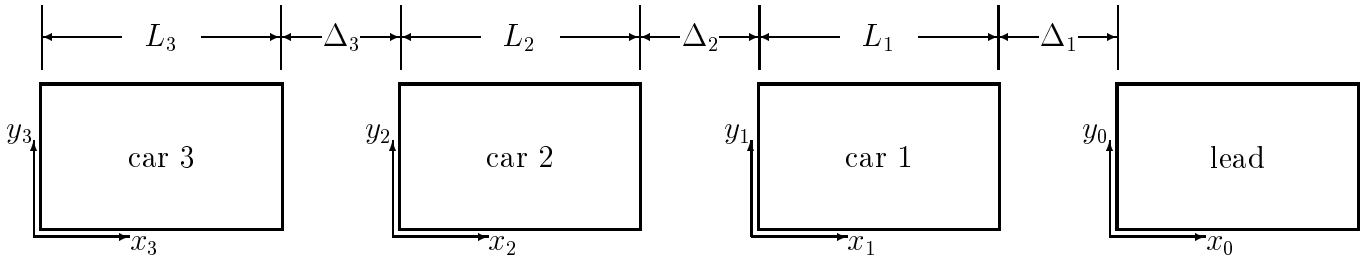

Figure 1: Model of a 3-car Platoon with Lead Car

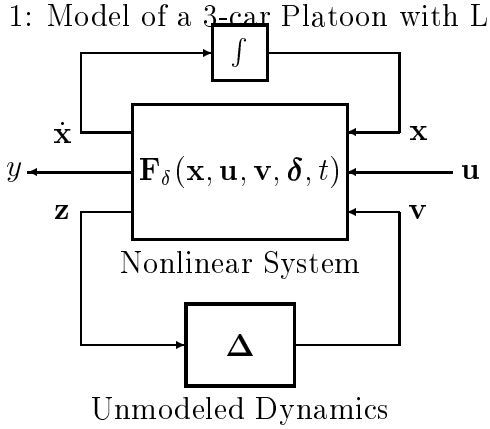

Figure 2: Schematic of Nonlinear System for Optimization

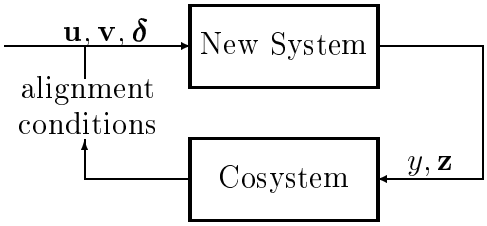

Figure 3: Schematic of New System/Cosystem Feedback Loop

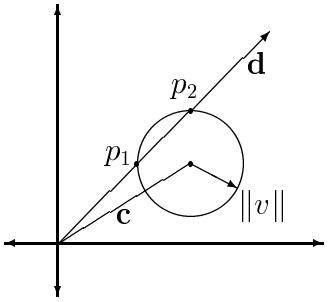

Figure 4: Graphical Solution to Vector – Circle Equation

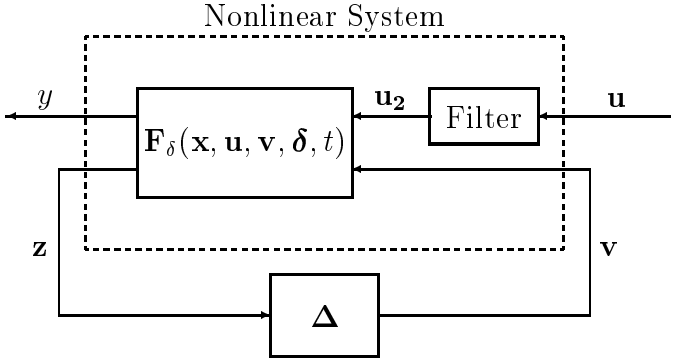

Figure 5: Schematic of a Nonlinear System with a Linear Filter for Disturbance Inputs

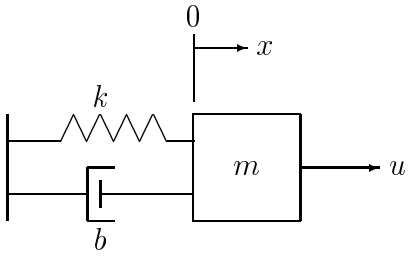

Figure 6: Schematic of Mass-Spring-Damper Dynamic System

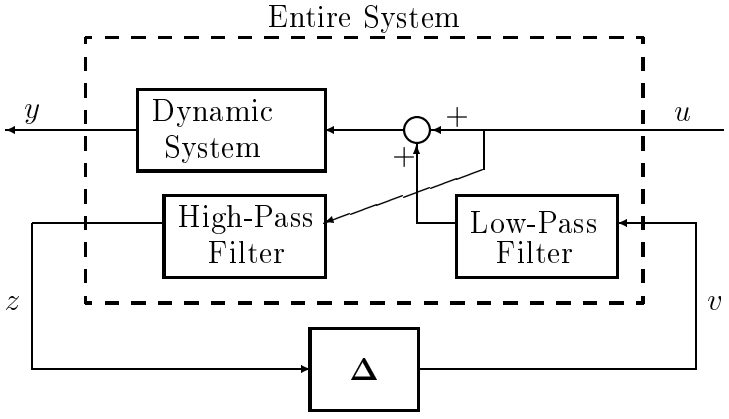

Figure 7: Schematic of Mass-Spring-Damper System with Unmodeled Dynamics

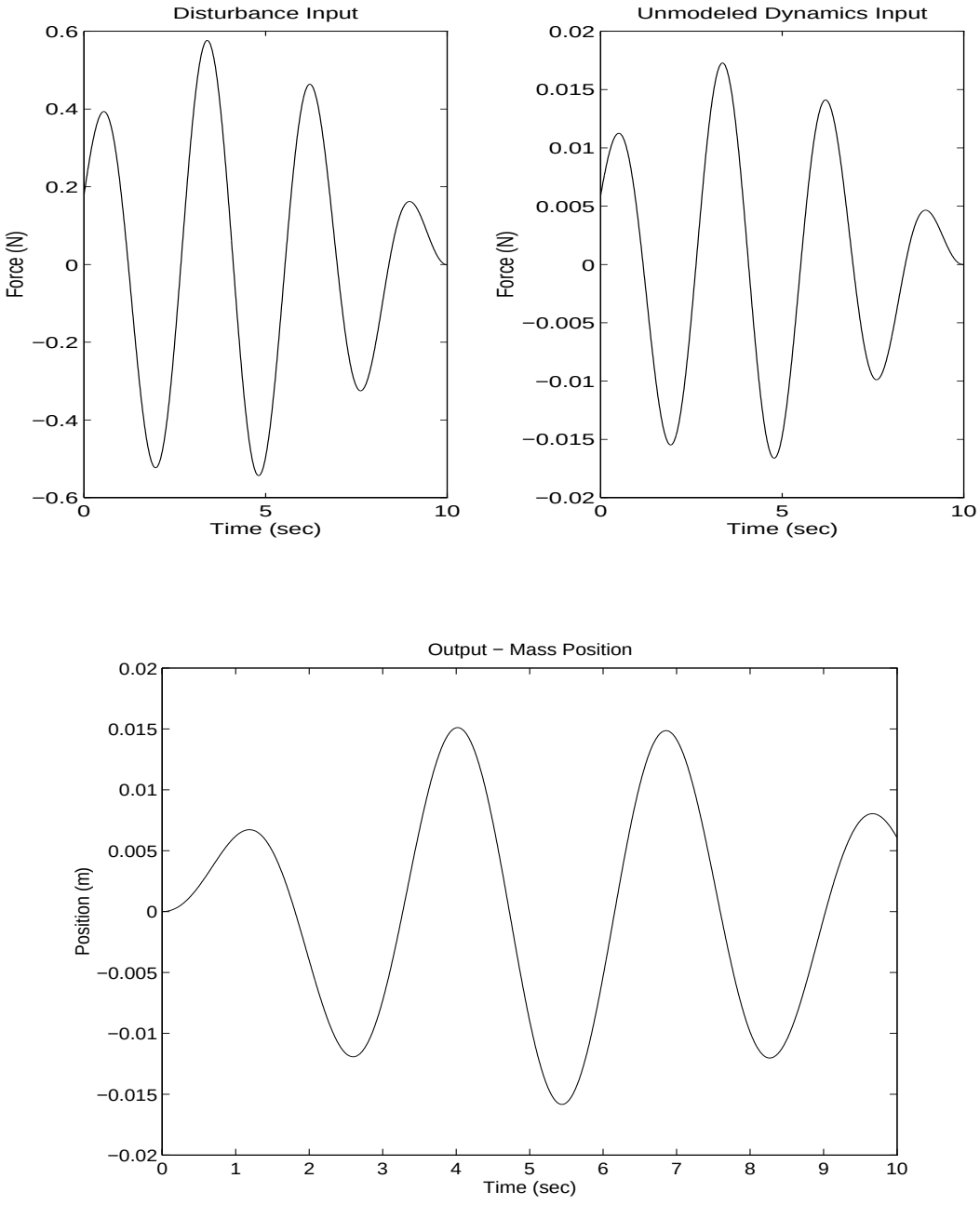

Figure 8: Final Input and Output Results of Mass-Spring-Damper System Using Worst Case Performance Algorithm

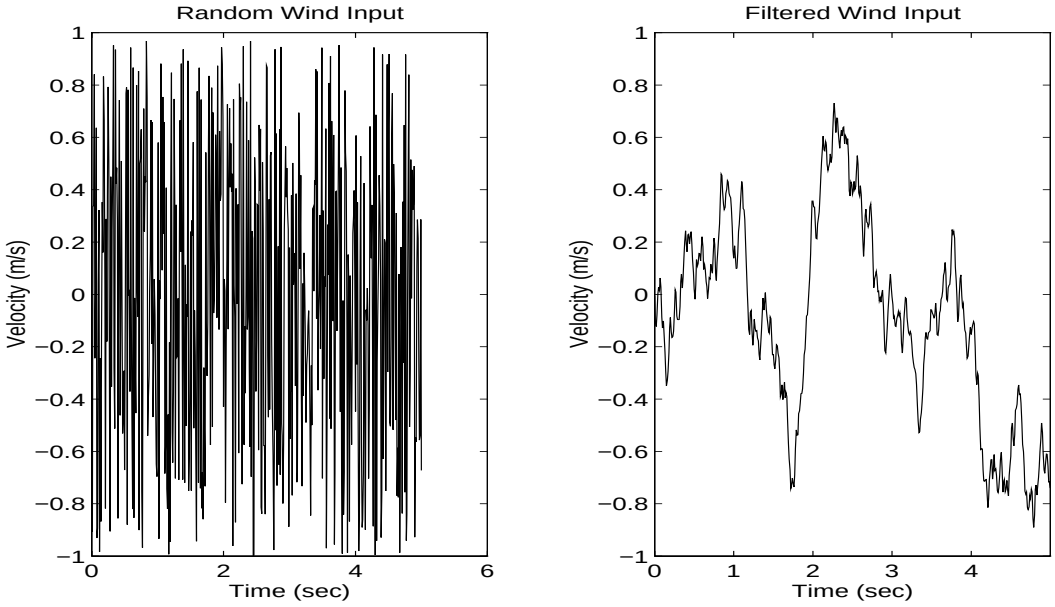

Figure 9: Random Wind Input and Wind Input After Filtering

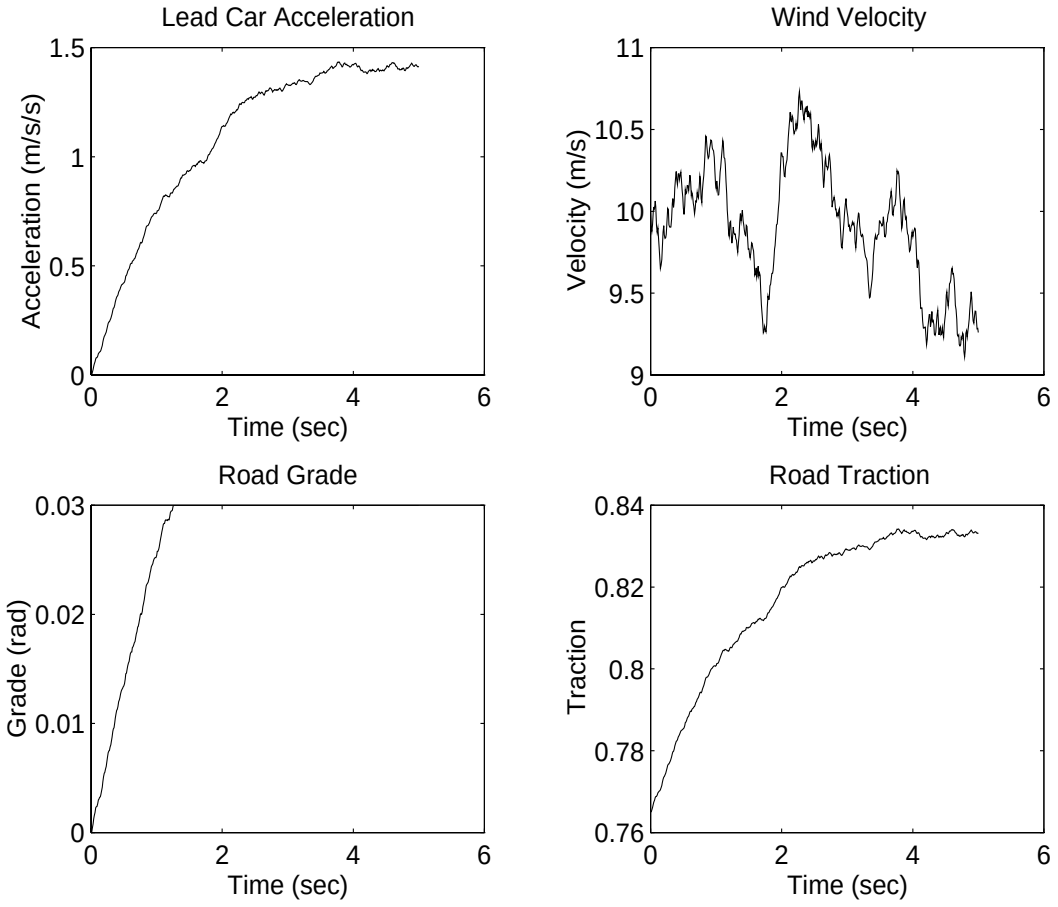

Figure 10: Final Inputs for Platoon Using Worst Case Performance Algorithm

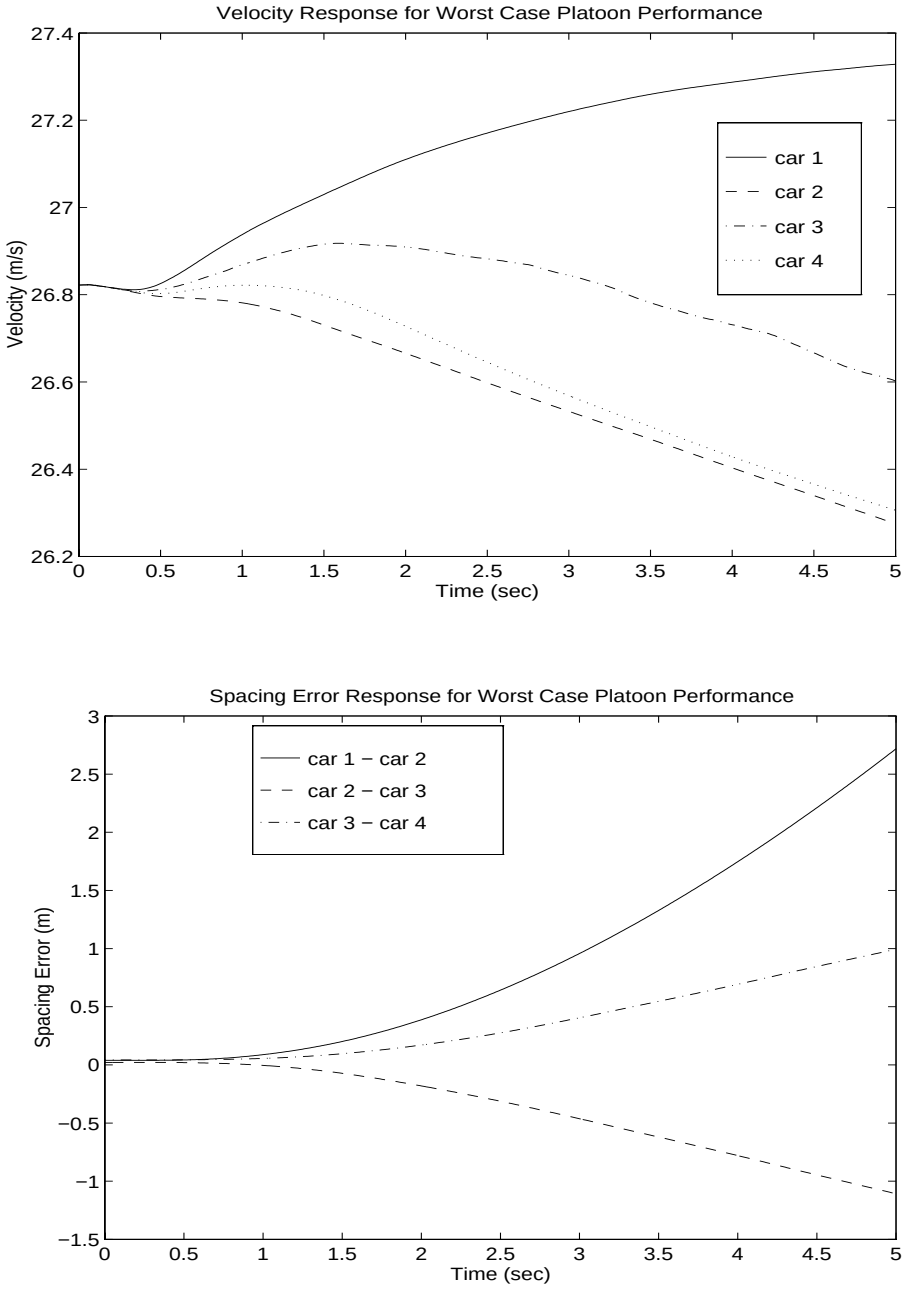

Figure 11: Velocity and Spacing Error Results for Platoon Using Worst Case Performance Algorithm

A Platoon Modeling Equations

A.1 Powertrain model

The equations for the reduced order engine dynamics [9] are:

$$F_{e,m}(\phi, v) = \alpha \cdot (1 - e^{-\beta \cdot \phi})^{\gamma} \tag{65}$$

$$\tau_e(\dot{\phi}) = 2.0905 \cdot \dot{\phi}^{-0.7033} \tag{66}$$

where

$$\begin{aligned} \alpha &= 1.0 \times 10^3 \cdot (-0.0053 \cdot v + 2.7404) \\ \beta &= 1.0 \times 10^{-3} \cdot (0.0613 \cdot v + 101.9315) \\ \gamma &= 1.0 \times 10^{-3} \cdot (18.8640 \cdot v + 855.0600) \end{aligned}$$

Further work to implement the equations for simulation [10] has reduced the dynamics to a first-order lag, where $\tau_e = 0.2$ sec, and limiting the engine thrust by the following saturation function:

$$F_{e,m}(v) = \alpha \cdot (1 - e^{-\beta \cdot 75.0})^{\gamma} \cdot (-0.0910 \cdot v + 3.5424) \tag{67}$$

where

$$\begin{aligned} \alpha &= 1.0 \times 10^3 \cdot (-0.01908 \cdot v + 2.7404) \\ \beta &= 1.0 \times 10^{-3} \cdot (0.22068 \cdot v + 101.9315) \\ \gamma &= 1.0 \times 10^{-3} \cdot (67.9104 \cdot v + 855.0600) \end{aligned}$$

Replacing the above saturation function with power curve relationships has shown good results for various cars. Multiplying the maximum torque from the power curves by the gear ratios and drivetrain efficiency and dividing by the wheel base produces the maximum engine force transmitted through the tires:

$$F_{e,m} = PC(v) \cdot e_{ff} \cdot GR/h \tag{68}$$

where

$$\begin{aligned} &T_{e,m} = PC(\omega) \qquad \text{(given by car manufacturers)} \\ &\omega = v \cdot GR/h \end{aligned}$$

Adding a term for uncertainty in engine effectiveness yields:

$$F_{e,m} = PC(v) \cdot e_{ff} \cdot GR/h \cdot (1 + e_{un}) \tag{69}$$

A.2 Brake model

The equations for the brake torque [3] are:

$$\tau_b \cdot \dot{T}_b + T_b = \alpha \cdot T_{b,m} \tag{70}$$

where the simplified equation for the maximum brake torque is given by:

$$T_{b,m} = \mu \cdot M \cdot g \cdot h \cdot TR \tag{71}$$

Including the parametric variations gives:

$$T_{b,m} = \mu \cdot (1 + \mu_{un}) \cdot M \cdot (1 + M_{un}) \cdot g \cdot h \cdot TR \tag{72}$$

A.3 Aerodynamic drag

The force due to aerodynamic drag is [7]:

$$F_a = \frac{\rho}{2} \cdot CR_D \cdot C_D \cdot A \cdot (v + v_w)^2 \cdot sgn(v + v_w) \tag{73}$$

Including terms in the above equation to account for parametric variations gives:

$$F_a = \frac{\rho}{2} \cdot CR_D \cdot C_D \cdot CR_a \cdot (1 + C_{D,un}) \cdot A \cdot (v + v_w)^2 \cdot sgn(v + v_w) \tag{74}$$

A.4 Rolling Resistance

The force due to rolling resistance is [9]:

$$F_r = M \cdot g \cdot f_r \cdot \cos(\theta) \tag{75}$$

The simplified relationship for the coefficient of rolling resistance is [9]:

$$\begin{aligned} f_r &= (4.864 \times 10^{-4} \cdot G_o - 1.03 \times 10^{-8}) \cdot v^3 \\ &\quad + (-0.0952 \cdot G_o + 1.1425 \times 10^{-6}) \cdot v^2 \\ &\quad + (7.0982 \cdot G_o - 3.1010 \times 10^{-5}) \cdot v + 0.01 \end{aligned} \tag{76}$$

where

$$4.050 \times 10^{-7} \text{ (good)} \leq G_o \leq 6.400 \times 10^{-6} \text{ (poor) for highways}$$

Rewriting the above equation in terms of tire hysteresis, axle height, and uncertainty in the coefficient of rolling resistance (which includes changes in G_o), the relationship for the coefficient of rolling resistance becomes:

$$\begin{aligned} f_r &= \frac{a_h}{h} \cdot (-7.209 \times 10^{-14} \cdot v^3 + 7.877 \times 10^{-12} \cdot v^2 \\ &\quad -2.007 \times 10^{-10} \cdot v + 7.136 \times 10^{-8}) \cdot CF_r \cdot (1 + f_{r,un}) \end{aligned} \tag{77}$$

A.5 Gravitational force

The force due to road grade is [4, 9]:

$$F_g = M \cdot g \cdot \sin(\theta) \tag{78}$$

Including parametric variations in M gives:

$$F_g = M \cdot (1 + M_{un}) \cdot g \cdot \sin(\theta) \tag{79}$$

B Simulation Specifications

The following is a list of variables and other specifications used in platoon simulations and the worst case performance algorithm.

B.1 Parameters

These values were chosen for a typical mid-sized passenger vehicle employing the V6 engine model described by Equation 68:

$$\begin{array}{rclrcl} A & = & 1.75m^2 & a_h & = & 49,050\ Nm \\ C_D & = & 0.40 & g & = & 9.81\ m/s^2 \\ h & = & 0.35\ m & M & = & 1800\ kg \\ \mu & = & 0.78 & \rho & = & 1.23\ kg/m^3 \\ \tau_b & = & 0.2\ sec & \tau_e & = & 0.2\ sec \end{array}$$

B.2 Parametric Uncertainties

The following are the weighting bounds chosen for the parametric uncertainties as described in [8]:

$$\begin{array}{rcl} e_{un} & = & \pm 15\% \\ \mu_{un} & = & \pm 25\% \\ M_{un} & = & \pm 12\% \\ C_{D,un} & = & \pm 15\% \\ f_{r,un} & = & \pm 16\% \end{array}$$

with the weighting correction factors:

$$\begin{array}{rcl} CR_a & = & 1.175 \\ CR_r & = & 1.100 \end{array}$$

Note that when employing the parametric uncertainty in the worst case performance algorithm, δ_i is bounded by ± 1. An uncertainty in a car model equation, *e. g.* $(1 + M_{un})$, will be changed to a weighting multiplied by δ_i, *e. g.* $(1 + M_{un,w} \cdot \delta_M)$ where $M_{un,w} = 0.12$.

B.3 Input Specifications

The bounds chosen for the inputs are [8]:

$$\begin{array}{rcccl} -10g & \leq & l_a & \leq & 7g \\ -20\ m/s & \leq & v_w & \leq & 20m/s \\ -0.06\ rad & \leq & \theta & \leq & 0.06\ rad \\ 0.52 & \leq & TR & \leq & 1.01 \\ -0.05 & \leq & SNR & \leq & 0.05 \end{array}$$

where, during one simulation, v_w will not vary more than $\pm$ 10 m/s and θ not more than $\pm$ 0.03 rad.

When implementing in the worst case performance algorithm, the inputs are filtered using a low pass, first-order filter. The filter specifications were chosen as follows:

Input	Pole (rad/s)	Zero (rad/s)	Gain at 0 rad/s
lead acceleration	-0.628	-628	3.924
wind speed	-1.256	-1256	10.0
road grade	-0.314	-314	0.03
road traction	-0.628	-628	0.245
all sensor noises	-314	-0.314	1.0

Since the average value of road traction is 0.765, we will implement TR as an input bounded by $\pm$0.245 with an offset of 0.765. The wind speed has an offset of 10 m/s for a variance from 0 to 20 m/s. The sensor noise inputs are allowed to vary between $\pm$1 and are then weighted by 5%.

References

[1] Balas, Gary J.; Doyle, John C.; Glover, Keith; Packard, Andy; and Smith, Roy, *μ-Analysis and Synthesis Toolbox*, MUSYN Inc. and The MathWorks, Inc., January 1994.

[2] Bryson, A. E. and Ho, Yu-Chi, *Applied Optimal Control: Optimization, Estimation, and Control*, Halsted Press, 1975.

[3] McMahon, Donn H. and Hedrick, J. Karl, *Longitudinal Model Development for Automated Roadway Vehicles*, PATH Research Report UCB-ITS-PRR-89-5, University of California at Berkeley, October 1989.

[4] Sheikholeslam, Shahab and Desoer, Charles A., *Longitudinal Control of a Platoon of Vehicles I: Linear Model*, PATH Research Report UCB-ITS-PRR-89-3, University of California at Berkeley, August 18, 1989.

[5] *SIMULINK User's Guide*, The MathWorks, Inc., Massachusetts, March 1992.

[6] Tierno, Jorge E.; Murray, Richard M.; and Doyle, John C., *An Efficient Algorithm for Performance Analysis of Nonlinear Control Systems*, Proceedings of the American Control Conference, Technical Paper 11-4:10, June 1995.

[7] Tongue, Benson H.; Moon, Ahrie; and Harriman, Douglas, *Low Speed Collision Dynamics: Second Year Report*, PATH Research Report, University of California at Berkeley, submitted December 1995.

[8] Tongue, Benson H.; Packard, Andy; and Sachi, Paul, *Qualitative Analysis on the Performance of Non-uniform Platoons: Report I, Non-uniformities and Performance Issues*, PATH Research Report, University of California at Berkeley, submitted May 1996.

[9] Tongue, Benson H.; Yang, Yean-Tzong; and White, Matthew T., *Platoon Collision Dynamics and Emergency Maneuvering I: Reduced Order Modeling of a Platoon for Dynamical Analysis*, PATH Research Report UCB-ITS-PRR-91-15, University of California at Berkeley, August 1991.

[10] Tongue, Benson H. and Yang, Yean-Tzong, *Platoon Collision Dynamics and Emergency Maneuvering II: Platoon Simulations for Small Disturbances*, PATH Research Report UCB-ITS-PRR-94-4, University of California at Berkeley, February 1994.