Pre-crash Sensing and Restraint Optimization for Automobiles

A thesis presented to the faculty

of the

School of Engineering and Applied Science

University of Virginia

In partial fulfillment

of the requirements

for the degree

Masters of Science, Mechanical Engineering

by

David C. Meeker

May, 1993

Approval Sheet

This thesis is submitted in partial fulfillment of the requirements for the degree of:

Masters of Science, Mechanical Engineering

Author

This thesis has been read and approved by the Examining Committee:

Accepted for the School of Engineering and Applied Science:

Dean, School of Engineering and Applied Science

Abstract

Since the early 1970s, researchers have attempted to use pre-crash sensors in order either to avoid or to lessen the severity of imminent collisions. Unfortunately, early attempts met with limited success due to both technological shortcomings and, in a more philosophical sense, flaws in their general approach.

This work revisits the notion of pre-crash sensing. The main objective is to develop a collision prediction system that performs non-intrusive actions so that vehicle occupants are protected in the event of a collision. Non-intrusive techniques are desired so that some false alarm predictions would be acceptable.

A paradigm is formulated in which this sensing can be used to reduce the severity of collision injuries. A neural network approach to medium range sensor signal processing is examined and found to be inadequate for the purposes of collision prediction. A deterministic scheme for short-range pre-crash sensing is developed, including several simulation examples of the process. Finally, passenger restraints are optimized with respect to the additional information derived from the pre-collision sensing algorithm. It is found that a significant advantage in reducing peak occupant acceleration is only obtained through the use of intrusive restraint systems, contrary to the initial aim of the study.

Acknowledgements

Thanks goes to:

To Eric Maslen, without whose guidance I never would have done this work.

To Miles Townsend and Walter Pilkey, for being kind enough to sit on my committee.

To A. Peter Allan, who almost never told me to quit talking shop.

To Winfred Meeker, Ann Meeker and Aimee Dalrymple, who provided essential moral support.

To William R. Meeker, Jr., a man of integrity and a tough act to follow.

Contents

List of Figures

Nomenclature

Chapter 1

I Input to a generic discrete time system.

O Output of a generic discrete time system.

S States inside a generic discrete time system.

α Rule mapping S at time $k-1$ onto S at time k.

β Rule mapping states S onto output O.

Chapter 2

a Braking rate.

d Measured value of x.

r Measured value of v.

t Time.

v Relative velocity between auto and target.

v_1 Derivative of x_1 with respect to time.

v_2 Derivative of x_2 with respect to time.

x Relative disance between auto and target.

x_1 Absolute position of rear of target.

x_2 Absolute position of front of sensor-equipped auto.

Chapter 3

c_i The i^{th} coefficient of a polynomial describing an object's desired path.

E Kinetic energy of impact.

$s_i(t_j)$ Range delivered by the i^{th} sensor at time step j.

threshold discrimination point for the determination of an alarm for the neural network system.

v Speed of an object.

v_{rel} Closing rate between auto and target.

x, y Fixed reference frame axes.

Δ Length of time step.

Θ Steering angle, degrees.

ϑ Direction of impact

ϕ Heading of an object measured from the y-axis.

ψ Half-width of radar beam, degrees.

Chapter 4, Appendix C

b Vector whose n^{th} entry is $Y_n - \hat{Y}_n + Y_{n+1} - \hat{Y}_{n+1}$.

e Squared profile matching error.

l Distance between sensors.

M Matrix whose n^{th} row is $\{\, 2(\hat{Y}_{n+1} - \hat{Y}_n), -2l \,\}$.

T Time.

ΔT Time step.

v_x, v_y relative velocity of a target object.

x, y relative position of the nearest point of a target object.

Δx, Δy Change in position of a target object over the span of ΔT.

X, $\hat{X}$ Transverse distance measured by sensors in the case of a non-parallel sensor array at times T and $T + \Delta T$ respectively.

Y, $\hat{Y}$ Sensor output at times T and $T + \Delta T$ respectively.

Chapter 5, Appendices D, E

A Instantaneous cost.

C_1 Auto body damping.

C_2 Restraint damping.

F Force impulse during collision.

f Applied restraint force.

f_k^* Force f at the k^{th} time step that minimizes cost J_k.

G System equations of motion.

J_k Recurrence relation cost function at the k^{th} time step.

J_∞ Maximum acceleration experienced by the occupant over the duration of the collision.

K_1 Auto body stiffness.

K_2 Restraint stiffness.

M A rigid mass representing an auto occupant.

M_{auto} Mass of the auto.

p A set of parameters that characterize a restraint system.

t Time.

U_k Minimum of J_k over f_k.

v Derivative of x with respect to time.

x Displacement of the occupant relative to the auto.

X State vector.

Appendix A

a_i Weighted sum of inputs to the i^{th} network node.

e Output error function.

in Inputs to the neural network.

m Number of network inputs.

n Number of network outputs.

w_i Weighting vector of the i^{th} network node (linear version of a_i).

W_i Weighting matrix of the i^{th} network node (quadratic version of a_i).

z_i Output of the i^{th} network node.

Chapter 1

Introduction

The present work arose from the intuitive notion that pre-crash radar sensing might work better for optimizing passenger restraints inside automobiles rather than actually braking the automobiles. In the past, attempts at automatic braking all yielded an unacceptably high rate of false alarms, making them unsuitable for production cars. The unwarranted braking could actually cause a hazard in a traffic situation. Restraint optimization, it was believed, would be a passive reaction to pre-crash sensors in which false alarms would not affect catastrophic results.

A second motivation for the study was the idea that a lower false alarm rate might be achieved by a better form of signal processing than had been used to detect imminent crashes in earlier studies. Since many different heuristic strategies had all been tried without success, a radically different approach was thought to be needed. Recently, neural networks have received much acclaim for their ability to learn functional relations between two sets on the basis of a sample data set. Such a device seemed like a solution to lower the high false alarm rate and perceive complex relationships between sensor data and collision prediction.

Work began first by investigating a neural network approach to collision predic-

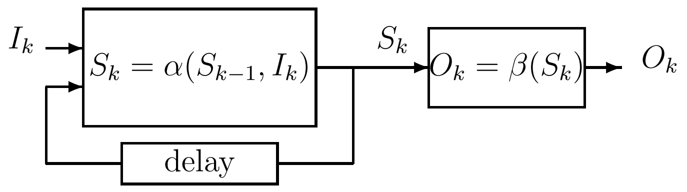


Figure 1.1: Generic Discrete Time Dynamical system

tion. To train a neural network, one has to identify a set of variables as inputs and another set of variables as outputs. The network then finds some kind of best-fit relationship between many sample sets of inputs and outputs.

For predicting crashes, however, it was not obvious what should be the inputs and what should be the outputs. The inputs should probably be some kind of outputs from the pre-crash sensing equipment, but there was no *a priori* notion of which outputs and from which times. Should data from the car itself be included? (speed, acceleration, turning angle, and so on.) And how is it known if any particular set of inputs corresponds to a warning condition? What condition merits warning?

Another problem had arisen that had been neglected in the original statement of the problem. That is, *What is the frame of reference?* What is the model of the system for this problem and is the model valid? What is the mathematical definition of the problem that is being solved, and what assumptions are being made in the solution?

What was needed was a way to mold the whole idea of collision detection and restraint optimization into a form that was amenable for modeling and simulation. The paradigm for the formulation came from control theory.

To make a model for use in a controls problem, one identifies a set of states (S_k), a set of inputs(I_k), and a set of outputs(O_k), and a set of rules for transforming those states and outputs forward in time (α and β), based upon the present values for states and the present inputs. A schematic representation of the system is drawn in Figure 1.1. The above devices are chosen such that the model imitates some facets of the real world deemed important to within some arbitrary degree of accuracy.

Once a frame of reference is defined, there exists a perspective with which to examine the several relevant questions involved in making a working system.

- Are the assumptions made in the model valid ones?
- Is it possible to validate or verify the model?
- Is there enough information available from the sensors to observe all the states of the system?
- Can an adequate prediction of the system state at some point in the future be made, and with what amount of uncertainty?
- Can the control objectives be achieved with the available system inputs?
- What are the performance objectives of the system and are they realistic?

Not all of these questions have precise mathematical answers (especially for nonlinear systems), but the reference frame provides a formal structure through which answers to these questions might be determined.

In Chapter 2, previous work is examined to find out how the system was modeled, what assumptions were made in creating that model, and what can be changed to achieve a better success.

For the remainder of the report, the system model is divided into two parts, each of which are more or less considered separately. The first part is the external model. This part of the model reflects the gross movements of the car in the road environment. The restraint system needs to be able to adequately observe the states of this part of the subsystem, but it cannot affect any control of its states. Then, there is the internal model reflecting the actual movements of the occupant as a result of a collision. The states of this subsystem need to be measured, and there must exist adequate mechanisms to control these states. Both systems are necessary to adequately model the entire system because the external model provides the impact force input into the internal model.

In Chapter 3, a fairly broad frame of reference is taken with regard to the external scenario. A model of the road with reasonably detained models of cars is developed in an attempt to solve the problem by using neural network methods.

In Chapter 4, a new frame of reference with new sensors and new assumptions is introduced to simplify the external system model and reduce the scope of uncertainties. With this new model, a new sensing algorithm is developed.

In Chapter 5, an attempt is made to identify a suitable model for passengers inside an automobile during a collision. Also addressed is the question of a suitable measure of system performance. An investigation of optimal methods to restrain the occupant proceeds on the passenger model.

Chapter 2

Previous Work

Several works have been published in which non-cooperative radar equipment (radar systems that are meant to work without reflectors installed on all possible targets) were used to either automatically brake the auto or warn the driver in dangerous situations. Work began in the early 1970's and has continued sporadically up to the present.

Early works include the 1972 studies of Hopkins, *et. al.*, [7] and Makino and Sato [10]. These studies focused on using a fixed radar sensor to scan a zone in front of the car. These radars estimated target position and velocity, and an effort was made to classify targets on the basis of radar signature. From this information, a collision judgement was made. If a collision was expected, an air bag was deployed.

Further works were published in 1974, this time with the emphasis shifted towards active collision avoidance by automatic braking rather than the triggering of airbags. Braking efforts were attempted at the Bendix Corporation [20] and by the AutoStop Corporation [5] Both attempts used a sensing configuration similar to the previous efforts devoted to airbag deployment.

A radar system was developed by G. Ross at the Sperry Rand Corporation [13] which still concentrated on the use of air bags. However, the suggestion was made that the same system might be useful in automatic braking. This study made the claim that a low false alarm rate might be achieved by the use of baseband radar rather than continuous wave systems. A follow-up to this study was published in 1978 [14]. In this work, the baseband radar was applied to automatic braking. Considerable effort was made to narrow the effective beamwidth of the system, presumably to cut down on false alarms by scanning only objects that could cause a frontal collision.

In an attempt to eliminate problems with false alarms, C. Kaplan and P. Sterzer at the RCA Corporation combined a cooperative system (a system where target vehicles would bear a radar reflective tag) with a non-cooperative system similar to the previously cited examples [9]. To restrict false alarms, the effective range was reduced to only a few meters in the case of non-tagged targets.

In 1976, Wong *et. al.* completed an examination of an automatic braking system using a fixed pulse doppler radar to predict front-end collisions [23]. Although several radar antenna configurations and maximum ranges were tried, false alarms proved to be too numerous for an effective automatic system.

In 1978, E. H. Dull *et. al.* reported improved performance for a fixed-radar collision avoidance system [3]. These results were achieved by making the collision judgement a function of many factors instead of merely approach velocity and distance. These factors included radar target signature measures and a range limitation dependent on speed and steering angle.

For several years, interest waned in anticipatory crash sensing. Through the end of the 1970's, efforts focused only upon a single fixed radar to detect frontal collisions.

By using basically only this one kind of experimental setup, researchers uniformly encountered the same dilemma: if the field of view was widened to detect a large percentage of front end collisions, the false alarm rate was unacceptable. The false alarm rate could be made very small, but then the sensitivity of the system was so reduced as to be of little value.

An example of narrowing the sensitivity to reduce alarms is the Rashid Radar Safety Brake in 1988 [12]. The radar beamwidth was made very narrow, so as to only be able to view an auto directly in front of the test vehicle. Conditions were imposed such that the system would not provide warning if approach velocity was over a certain threshold. Objects like pedestrians, animals, and so on, that do not produce a large radar echo were also ignored. In this case, false alarms were low, but many important accident scenarios (e.g. a car pulling out from a side street) were ignored by the conditions imposed to reduce false warnings. Any road curvature, crest, or sag that resulted in a misalignment between the test car and the target also seriously affected performance, since the target would then have moved out of the narrow sensing beam.

Also in 1988, there was the Nissan Laser Collision Avoidance System. [18] [11] This system used a forward-looking laser beam instead of a radar. However, the information garnered by the laser sensor and the processing scheme were nearly identical to previous attempts using radar. The result was a system that had an unacceptably high rate of erroneous warnings.

To some, it seemed that a greater amount of information needed to be gathered by the radar sensors to provide a good discrimination of a dangerous situation. In about

1984, C. L . Lichtenberg at NASA Johnson used a modified phase monopulse radar that could also give a measure of the angle of the target, even though the radar was fixed on the test vehicle [14]. This study still resulted in limited useful performance. Multipath effects and variable target size contributed to unacceptable accuracy in target angle measurements. This angle measurement was seen as a key attribute for a system which would be able to predict collisions with a low rate of false alarms.

With improvements in radar hardware technology, a scanning radar system has currently become economically feasible. Several candidate scanning radar systems were reviewed by M. Alvis *et. al* in 1991, but these systems have not yet been included in any collision prediction experiments. [1]

Though these studies occurred over a span several years, all faced the similar challenges and many used basically the same experimental configuration. Ultimately, all fell victim to the same difficulties. Since the studies are very similar, only [23] will be examined here in more detail.

The experimental apparatus of the 1976 study consisted of a 1973 Lincoln Continental equipped with a fixed-beam radar mounted on the front grill of the auto. Additional signal processing equipment and actuators were also installed to allow interpretation and implementation of the radar signal and automatic braking.

The objective of the system was to anticipate rear-end collisions and to apply braking so as to avoid the collision. The radar would send pulses ahead of the car. Then, the signal processing setup would interpret the stream of returned pulses to deduce the distance and relative closing rate of the nearest sensed object. Finally, the signal processor would decide if a collision was about to occur. If a crash was anticipated, the brakes would be applied.

In general, the system was a failure. The system did indeed warn of anticipated collisions, but the level of false alarms was unacceptably high. Alarms might be triggered by roadside signs, overpasses, oncoming traffic in the opposite lane, the sides of buildings, and so on. During the course of the investigation, hundreds of responses indicating imminent collision were generated, even though the investigators never had a collision during the in-traffic test runs. In fact, enabling the automatic braking in an actual traffic situation would prove to be a greater hazard than not using the system at all – the radar equipped auto would employ hard braking unpredictably and uncontrollably. Fortunately, such a test was never attempted.

Though the 1976 device was not destined to be a production model, it did expose several obstacles that would have to be surpassed or avoided in any future successful effort. These problems lay both in the technology available in 1976 and in the philosophical approach taken towards the concept of remote sensing.

The first technical problem was what was called the *target recognition* or *target signature* problem. When an object was detected, it was not possible to tell exactly what that target was – automobile, road sign, overpass, etc. This problem arose out of the one-dimensional, limited information nature of the fixed radar. From simply a distance and a closing speed measure, little could be assayed about the physical character of the target. In this study, the only possible remedy was to shorten the range of the sensor and narrow the beam width. This fix did reduce the number of false alarms, but at the price of sensitivity towards targets that might cause an actual collision.

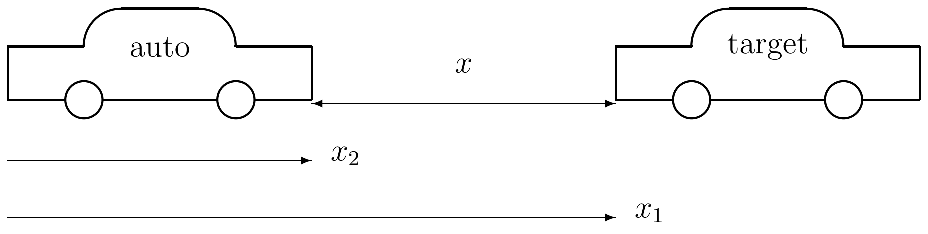


Figure 2.1: 1-D collision scenario.

The signal processing scheme also contributed to the incidence of false alarms. The processing concept assumed that

- the only object that could be detected was a valid target (*e.g.* no overpasses, etc.)
- both the target and the radar-equipped vehicle only move in one dimension (along a straight line)
- the radar-equipped vehicle's maximum deceleration rate is known
- the target always moves at a constant speed.

Under these assumptions, one can write exactly the sensor conditions in which a crash is indicated (See Figure 2)

Target:

x_1 = position of rear of target

$$\begin{aligned} \dot{x}_1 &= v_1 \\ \dot{v}_1 &= 0 \end{aligned}$$

Auto:

x_2 = position of front of auto

$a =$ braking rate

$$\begin{aligned} \dot{x}_2 &= v_2 \\ \dot{v}_2 &= a \end{aligned}$$

Absolute positions x_1 and x_2 are not important; just their relative positions and relative velocities. The system equations are then

$$\begin{aligned} \dot{x} &= \dot{x}_1 - \dot{x}_2 &&= \quad v_1 - v_2 = v \\ \dot{v} &= \dot{v}_1 - \dot{v}_2 &&= -a \end{aligned}$$

sensor output

$$\begin{aligned} d &= x + noise \\ r &= v + noise \end{aligned}$$

The system integrates to

$$\begin{aligned} x(t) &= -\tfrac{1}{2}at^2 + v(0)t + x(0) \\ v(t) &= -at + v(0) \end{aligned}$$

A successful stop can be made if $x > 0$ when $v = 0$.

$$\begin{aligned} 0 &= -at + v(0) \\ t &= \tfrac{v(0)}{a} \\ 0 &< -\tfrac{1}{2}at^2 + v(0)t + x(0) \ = \\ & \quad -\tfrac{1}{2}a(\tfrac{v(0)}{a})^2 + v(0)(\tfrac{v(0)}{a}) + x(0) \ = \\ & \quad \tfrac{v(0)^2}{2a} + x(0) \end{aligned}$$

and a collision occurs when

$$x(0) \leq -\frac{v(0)^2}{2a}$$

substituting in observed sensor information for x and v, the warning law is then

$$d \leq -\frac{r^2}{2a}$$

Unfortunately, the above model uses assumptions that are not valid in an everyday driving environment. The roadway is an inherently three-dimensional scenario (see Figure 2.2), and there is large uncertainty in target motions extrapolated beyond a

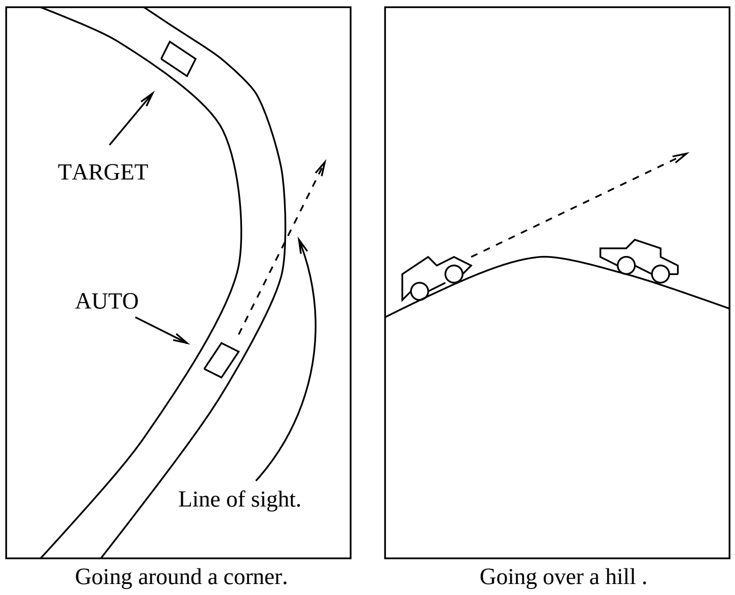


Figure 2.2: Examples of failure in 1-D assumptions.

small fraction of a second. Consequently, the processing scheme signaled many false alarms.

Some of these inadequacies could be remedied by scanning radar, as suggested in [1]. With an ideal scanning radar system, a complete two or even three dimensional representation of the road might be produced. With such information, an intelligent signal processing program might hope to solve the target recognition problem. However, other problems with the braking approach would still lead to inadequate performance.

Problems with the scanning radar approach lie in the necessity of the system to predict the traffic situation up to several seconds in advance. A fairly large amount of lead time is necessary for automatic braking or driver warning so that the car may adequately brake prior to collision.

Currently, scanning radar systems for collision avoidance are in operation for civil

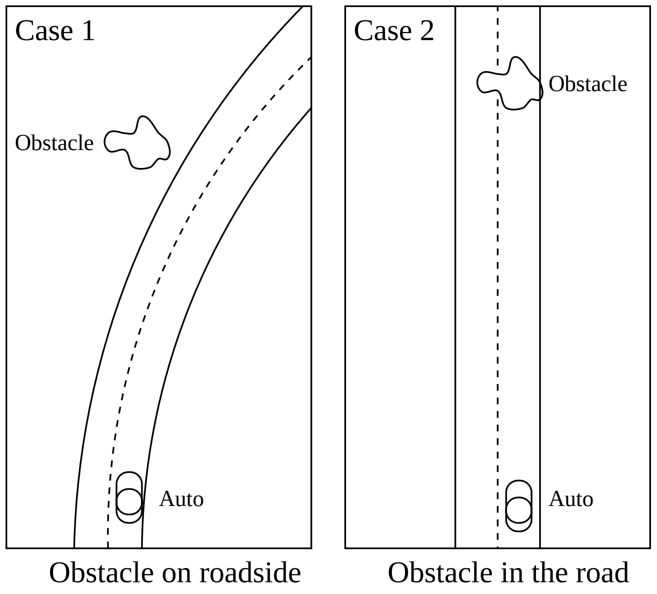


Figure 2.3: Two collision prediction scenarios.

marine and aviation use. In both the naval or aeronautical scenarios, the vehicles move along straight lines or in pre-determined patterns for long periods of time in environments that are for the most part free of obstacles. In these cases, the relevant information in making a collision prediction is only the positions and headings of near-by vehicles whose future positions might be predicted by extrapolation of the present trajectories. This is exactly the information available from a scanning radar. It is the hope of those interested in the scanning radar approach that a similar system may be synthesized for use in cars.

A traffic environment, however, presents a much more complicated problem for a scanning radar. The positions and velocities of all objects in the immediate vicinity might not be enough information to predict a collision up to several seconds in advance. Other cues are necessary to grasp the traffic situation and react accordingly. For example, consider Figure 2.3. Consider that in both cases in the figure, the driver has the same velocity and heading, instantaneously pointing toward the obstacle. In the first case, however, the obstacle is off to the side of the road. By simply following the road, no accident will occur. In the second case, however, the obstacle is truly a hazard – if the driver continues along the road, a collision may occur. However, without the ability to see the markings on the road, it would be difficult to see any difference between these two different situations.

The scanning radar also cannot read a driver's intent. Consider again the previous example. Although the car in case (1) would not have a collision if the driver were to simply stay on the road, there is no guarantee that the driver intends to veer right with the road. Likewise, in case (2), no collision will result (nor any action be necessary on the part of the crash-prediction system) if the driver intends to eventually slow down or swerve around the obstacle. Both cases could potentially result in a collision, and the uncertainty in the driver's actions in the immediate future makes the prediction of a collision uncertain.

As opposed to a brute force technological fix to the automatic braking problem, the goals of the sensing effort should be modified to fit the present constraints on sensing and signal processing. The new objectives would address activities occurring within shorter time spans (where the level of prediction uncertainty is smaller) than are required for automatic braking. Sensing configurations would be identified in which the motions of all significant targets are observable, or all least configurations that have very good statistical properties with regards to predicting collisions.

Chapter 3

Neural Network-based Signal Processing

3.1 Introduction

In an effort to remedy the shortcomings of earlier works, an attempt was made to identify likely processing algorithms and sensor configurations by statistical methods. The approach adopted in this study for examining collision prediction was based on simulation. A simple traffic simulator was constructed which permitted driving a test vehicle equipped with a specified sensor array through various traffic scenarios and collecting the sensor data stream throughout the simulation. By examining the correlation between the sensor data stream and the collision outcome of the simulation in as general a manner as possible, it might be possible to establish more complex correlations than would be determined through more heuristic methods as in past studies. This would also permit correlating a larger collection of data than in the past, where only one sensor could readily be handled.

To form this correlation between sensor data and collision outcome, a very large number of simulations of randomly selected traffic scenarios was run to form a data base of sensor data sets and associated collision outcomes. This data base was then used to train a feed-forward neural network using a backpropagation gradient descent algorithm [15] [21]. At the end of the network training, the correlation sought was contained in the gains and functions of the neural network, representing a functional relationship between the sensor outputs and the collision prediction. For more specifics of the neural network implementation, see Appendix A. Evaluation of this correlation was accomplished by developing a second data base called an evaluation set by simulating a new set of randomly selected traffic scenarios and acquiring new sensor data set / collision outcome pairs. The already trained neural network was then tested against this data set to see how often it would correctly predict a collision, how often it would fail to predict a collision, and how often it would give a false alarm.

3.2 Traffic Simulation

3.2.1 Model Overview

The model is meant to simulate traffic traveling down a curving two lane road. The environment is considered to be flat, and three dimensional effects (*e.g.* overpasses, overhead signs, multipath effects) are neglected.

For the purposes of this model, the environment is divided into a finite number of discrete objects any of which is a potential obstacle for the test vehicle. Each object has the following characteristics: (see Figure 3.1

- (x, y) coordinates of some reference point on the object in a fixed reference frame.

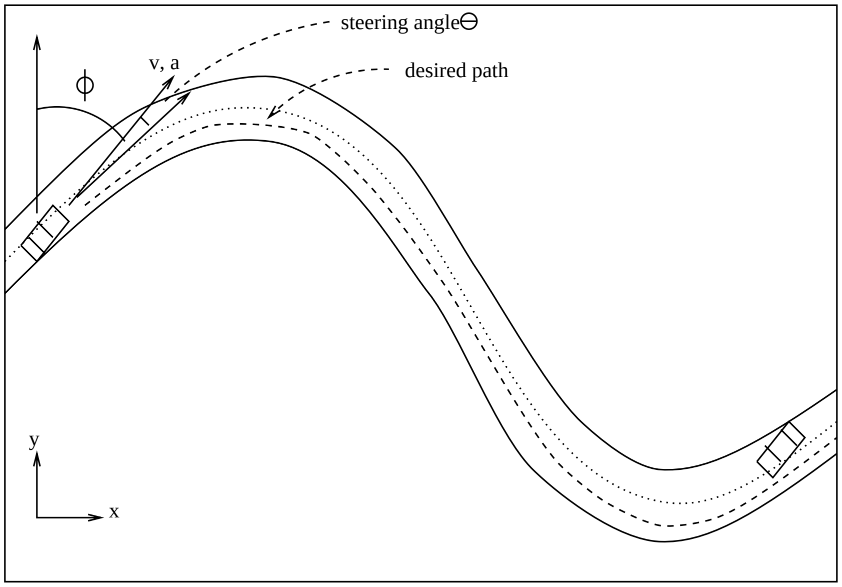


Figure 3.1: Traffic simulation scenario.

- heading ϕ, an angle measured from the positive y axis in the fixed reference frame.
- a velocity v along that heading.
- $c_1, \ldots, c_n$ coefficients of a polynomial that describe the desired path of the object. Usually this polynomial describes a line running down the middle of the right lane of the road relative to the direction that the car is traveling.
- type of object (*e.g.* car, motorcycle, pedestrian).
- A number of test points on the object consisting of the location of the point relative to a the object's local reference frame
- the wheelbase of the object, if applicable.

In an effort to make moving objects follow their desired path, each moving object also has two control inputs, the steering angle Θ and an acceleration a.

3.2.2 Automobile Control Law

The motion of these objects during the simulation is described by a set of four coupled equations of motion. Typically, these equations are

$$x_{t+1} = x_t - \Delta[v \sin(\phi_t + \Theta_t)]$$

$$y_{t+1} = y_t + \Delta[v \cos(\phi_t + \Theta_t)]$$

$$v_{t+1} = v_t + \Delta a_t$$

$$\phi_{t+1} = \phi_t + \Delta[v \sin \Theta_t]/wheelbase$$

where t is the t^{th} time step into the simulation, and Δ is the length of each time step. The object motion depends not only upon past values of x, y, v and ϕ, but also upon the control inputs a and Θ.

The purpose of the control law or driving algorithm which relates the control inputs Θ and a to the motion of the vehicle is to mimic the way that a human driver might respond to the given road situation. The specific nature of this control law strongly dictates the mechanisms that cause collisions and thereby the statistics of those collisions.

For most of the simulations, acceleration was assumed to be zero in order to simplify the simulation math. (This restriction was dropped in one of the last example cases, as will be discussed.) This reduces the problem to finding the best steering angle at each instant in time. The maximum permissible steering angle decreases with increasing vehicle speed. For example, the wheels might be able to turn 30° when maneuvering at 5 mph in a parking lot but at only 2° or 3° at 60 mph while driving down the highway. At an angle greater than the maximum, the car will lose traction. Therefore, these angles are deemed unacceptable by the model.

The steering algorithm is constructed in such a manner as to minimize a "cost of trajectory error" function. For each vehicle, a cost function is selected whose value is roughly proportional to the distance of the vehicle to the nearest point on the desired path. With acceleration always kept at zero, the range of acceptable Θ for the object's velocity is searched to find that value of Θ which results in the lowest cost at the next time step.

Because velocity is constant and the maximum angle of Θ is restricted by a function of velocity, the vehicle may leave the road or cross the centerline in curves, just as if a human driver negotiates a tight curve at too high a speed. A collision can occur if a car or stationary roadside obstacle is in the way when these deviations from the desired path occur.

3.2.3 Sensor Models

As previously discussed, each object in the model has a set of associated test points located along its edges. The first step of the sensor simulation is to transform the coordinates of all of the test points from the each object's local reference frame into the global reference frame. These points are assembled into a table which records the global position of each data point, which object the datapoint is on, and whether or not is it viewable by the test car's sensors.

A region describing the sensor's coverage is bounded by four lines: (see illustration in Figure 3.2)

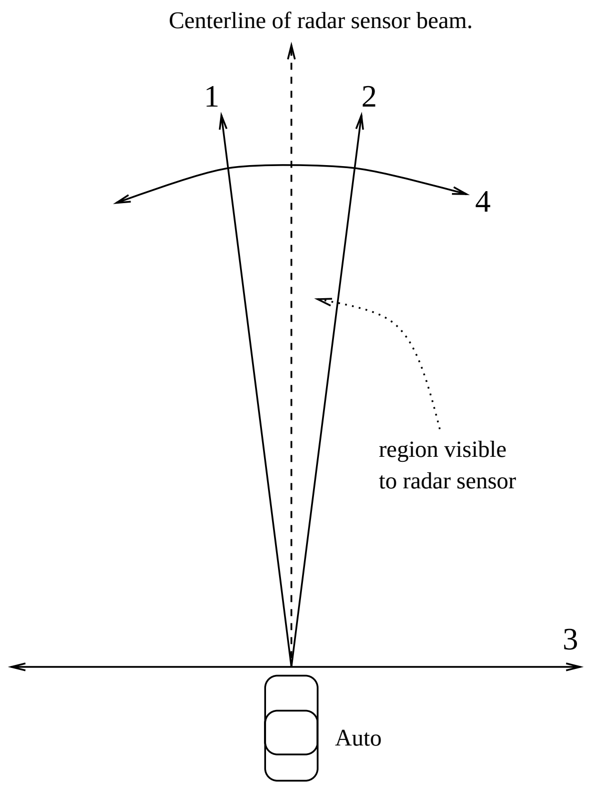


Figure 3.2: Fixed radar field of view.

1. left edge of the sensor arc
2. right edge of the sensor arc
3. a line normal to the test vehicle's heading
4. a circle that describes the sensor's maximum range

These lines can be thought of as the loci of points where four 3- dimensional surfaces intersect the 2-dimensional model. These surfaces are defined such that when a point in the model is projected onto each of the 3-D surfaces, it is in the positive part of every surface only when the point is inside the sensor-visible arc. Specifically, the equations for these surfaces are

$$z = \cos(\phi + \psi)x + \sin(\phi + \psi)y - \sin(\phi + \psi)Y - \cos(\phi + \psi)X$$

$$z = -\cos(\phi - \psi)x - \sin(\phi - \psi)y + \sin(\phi - \psi)Y + \cos(\phi - \psi)X$$

$$z = \cos(\phi + \pi/2)x + \sin(\phi + \pi/2)y - \ sin(\phi + \pi/2)Y - \cos(\phi + \pi/2)X$$

$$z = -(x - X)^2 - (y - Y)^2 + r_{max}^2$$

where (x, y) is the global position of the test point, (X, Y) is the global position of the test vehicle, and ψ is the half-width of the sensor's arc.

After each point is tested, the sensor simulation returns the Euclidian distance to the nearest test point and the component of closing velocity of the test point that is visible to the radar sensor. In the case of a forward-looking radar on the centerline of the radar equipped car, that velocity, v_{rel}, is given by

$$v_{rel} = v_{target}(\sin\phi_{target}\sin\phi_{auto} + \cos\phi_{target}\cos\phi_{auto}) - v_{auto}$$

In later versions of the simulation, a sensor model resembling a scanning radar or phased array sensor rather than fixed police-type radars is used. In this variant, the distance to each test point is calculated and both components of the relative velocity computed. The sensor model then returns the distance and both velocity components of the nearest test point moving towards the test vehicle.

3.2.4 Collision Detection

Collisions are detected in the same manner as sensor contacts. Four lines are drawn that define the edges of the test car. Planes are then constructed that intersect the plane of the model in such a fashion that points inside the edges of the car give all positive results when projected onto the planes. If any of the test points are on the positive section of all four planes, a collision has occurred.

3.3 Collision Alarm Synthesis

Ideally, given a data set consisting of some collection of measurements from the sensor array, there would be a mapping or function which would produce the direction and kinetic energy of any impending collision. Negative kinetic energies would indicate that no collision was expected. Such a mapping might look like:

$$\left\{ \begin{array}{c} E \\ \vartheta \end{array} \right\} = f\left(s_1(t_1), s_1(t_2), \ldots, s_1(t_n), s_2(t_1), s_2(t_2), \ldots, s_2(t_n), \ldots, s_m(t_1), s_m(t_2), \ldots, s_m(t_n)\right)$$

The kinetic energy of impact is represented by E while the direction of the impact is represented by ϑ. Note that, for effective restraint optimization, both of these quantities must be known. The s_i are range data from the i^{th} sensor, taken at time t_j. Other information which might contribute to the function include test vehicle position, velocity, and acceleration; road condition; time of day; and so on.

If it were possible to exactly construct a model of the obstacles around the moving car from the sensor data, then such a function could be found. A function of this type would be described as a causal relationship. However, some simple experimentation and consideration of the geometry of the traffic problem in two or three dimensions revealed that such a reconstruction is not possible without a full blown pattern recognition analysis of a fairly detailed image of the car's surroundings. (Methods based on pattern recognition are used in terrain following navigation systems for fighter aircraft, for example.)

Preliminary experiments indicated that, for the intermediate range sensing scheme employed in this simulation, a correlation to collision energy and direction was extremely difficult to identify. Consequently, the simpler problem of simply triggering a collision alarm was examined. While such a simple alarm would be of little value in optimizing the passenger restraints, it was felt that, by examining the properties of this simpler problem, some understanding of the basic detectability problem could be obtained. Thus, rather than seeking a correlation between sensor signals and impact energy/direction, the signals were correlated to a “likelihood of collision” measure or warning rule.

A further conclusion recognized at an early juncture was that, because it is not possible to fully reconstruct the obstacle field from the restricted sensor data available, any correlation between sensor data and collision outcome would be of a statistical nature. That is, since the collision predictor does not have enough information to perfectly extrapolate the motion of the test vehicle and its surrounding obstacles to determine imminence of collision, it must make a “best guess” based on its “experience” with similar data. This “experience” and the corresponding “best guess” takes the form of a complex conditional probability distribution.

In all cases, the synthesis of this warning rule was done using a feed-forward neural network. A feed-forward neural network is a method for determining a least-squares continuous curve fit between a set of input and output samples. To create to sample data record, some arbitrary collection of data for each instant was taken to be the input, and a -1 or 1 was taken to be the output, depending upon whether or not a collision occurred in the simulation within one second of when the data set was recorded. When the neural network is trained, the network reflects a smooth surface varying between -1 and 1. By picking some threshold, one can declare every response above the threshold an alarm and everything below the threshold as safe.

Several different input configurations were tested during the course of exploring the simulation. Since the mapping from inputs to warning level was found to be of a statistical rather than a deterministic nature, the hope was to experimentally identify some input configuration in which all of the collision data would be highly clustered together and well separated from the non–collision data. Such a space would yield a reliable predictor of imminent collisions with a low incidence of false alarms. Additionally, such a mapping would be determined automatically by the statistical characteristics of the data rather than by some arbitrary heuristic, as with previous studies of this sort. Complete results for each test configuration are contained in Appendix B.

3.4 Simulation Results

3.4.1 Case 1

Since there was no clear choice for a specific time history of inputs and since low dimensionality is more conducive to empirical curve fitting, this test included only the range and closing velocity output of three sensors mounted on the test car. Each of the sensors has an effective range of 300 feet and scans an arc of 4°. The sensors are located on the front of the car at −30°, 0°, and 30° from a line running down the center of the vehicle.

In this test run, the only obstacle is a car moving down a curving two-lane road moving in the opposite direction. The speeds of each vehicle were chosen randomly during each trial run between the values of 35 and 45 miles per hour.

The entire record for each test run was stored in memory. Every sensor contact

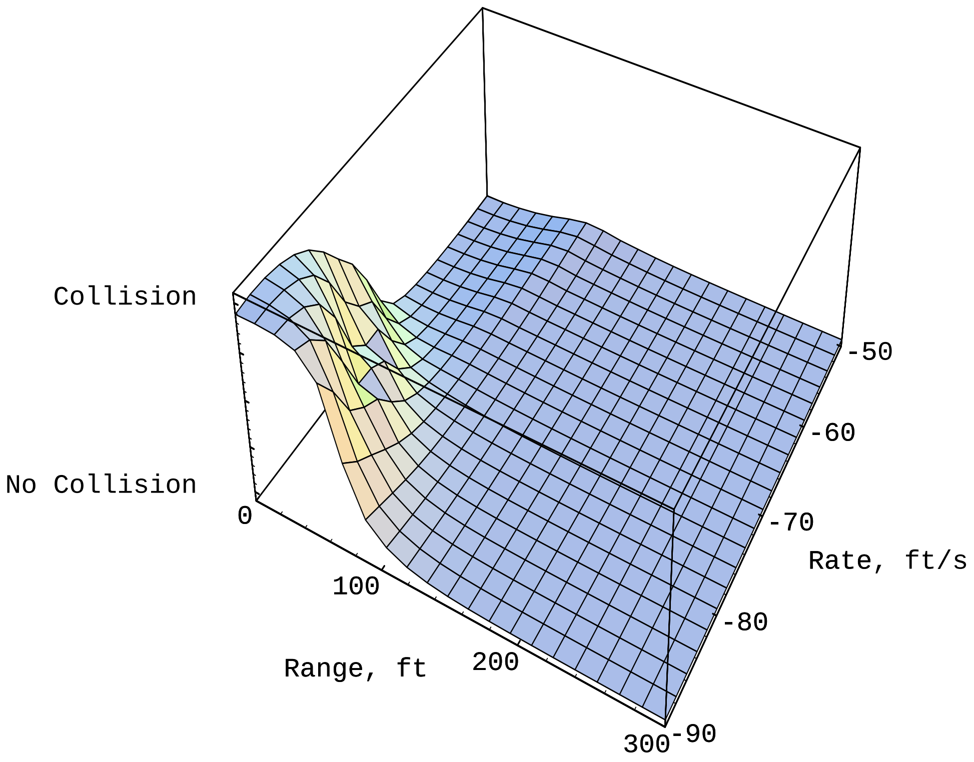


Figure 3.3: Decision Surface for Center Sensor, Case 1.

was then stored to disk with an indication of whether or not a collision occurred within one second of that instant. For the specific test set used to train the network, the log contained data from 500 simulation runs (28 of which resulted in collisions) and 7305 data points. Finally, a six–input, one–output feed-forward neural network was trained to this training set using a gradient descent algorithm.

In this example, there is only one object that can be seen, and the sensors do not overlap. Therefore, the sensors are effectively de-coupled, and the network's actual discrimination function can be shown.

Figure 3.3 shows the mapping that the neural network produced for objects seen by the sensor pointing straight ahead in the condition that the other two sensors see no object. As previously stated, a threshold is chosen, and if the height on the surface corresponding to the rate – closing rate pair is higher than the threshold, an alarm is sounded for one second. Otherwise there is no alarm.

Response from this network was not particularly good. Though the net defined areas of the inputs space that were highly associated with collisions, the small arcwidth of the sensors ($4°$) meant that objects were often not seen at all before a collision occurred. An example of the net's typical response is the $threshold = 0$ case – any input set that provokes a response greater than zero triggers an alarm. For a typical test run of 1000 simulations, 69 collisions occurred. A total of 48 collision alarms were generated, but only 35 of these corresponded to actual collisions; the rest were false alarms. The average warning time was 0.158 seconds. Typically, the false alarm rate was acceptable (as above), but the percentage of collisions detected, only 51%, was unacceptable.

A possible remedy is to widen the arc widths of the sensors considerably beyond their $4°$ widths. By widening the arcs, the problem of completely undetected objects would be addressed.

3.4.2 Case 2

The same configuration was used as in Case 1, except that $30°$ arcwidths were used rather than $4°$ arcwidths. The same obstacle was used – one car in the adjacent lane traveling in the other direction as the test car.

An identical network training procedure was followed in the creation of Case 2 as in Case 1. The resulting decision surface for the forward view is nearly identical in this case as in this case as in the previous example. However, the right-facing sensor was found to have no connection with collisions strong enough to provide anything but a flat surface near the -1 alert level throughout the entire mapping. The left facing sensor still retained a region where alarms could be created, but this region was reduced sharply such that alarms from the left sensor would not be created unless an obstacle was within ten feet and at a high closing velocity.

Under testing conditions identical to the conditions under which the data was generated, Case 2 performed markedly better than Case 1. For example, at one $threshold = 0$ testing, 60 collisions occurred, 114 alarms were sounded, and 55 collisions occurred during an alarm. Average warning time was much better, now at 0.515 seconds and the percentage of collisions detected increased to 92%.

3.4.3 Case 3

The next issue to examine was whether the performance would be maintained under testing conditions differing from those under which the training data was generated. To examine this question, 10 point obstacles (representing small, stationary road-side objects like road signs, trees, and rocks) were placed randomly throughout the course at a distance between 30 and 70 feet from the centerline of the road. In this case, the system performed very poorly. Only a small percentage of collisions were detected. The reason for the drastic degradation in performance can be seen by referring back to Figure 3.3. Since all of the collisions in the training set were the result of essentially head-on impacts with an automobile moving in the other direction, only in a region where closing rate was greater than about 70 mph could an alarm be generated. Collisions just didn't happen at slower closing rates in the training set; therefore, the net did not learn to create alarms for those cases. Since collisions with roadside obstacles occurred at around 35 to 45 mph closing rates, none were detected.

3.4.4 Case 4

Subsequently, a new network was trained using the same configuration as Case 2, but with a dataset derived from exactly those conditions where Case 3 showed poor performance: with roadside obstacles. After training, examination of the network's mapping showed that under these training conditions, the side-view sensors had no correlation whatsoever with the prediction of collisions. Effectively, the side-view sensors were turned off by the network and their output ignored.

For comparison with the mappings from the previous to cases, Figure 3.4 represents the decision surface for the forward facing sensor for this training. As one can see, the areas where an alarm can be generated have expanded down into the slower closing velocities. Another notable change is that the range from which a collision warning can be produced is cut back considerably as compared to Case 1. For example, at $threshold = 0$, alarms can only be produced in the 20 to 40 foot range, depending upon the approach velocity. Typically, the system responded well under testing conditions when an appropriate threshold was chosen.

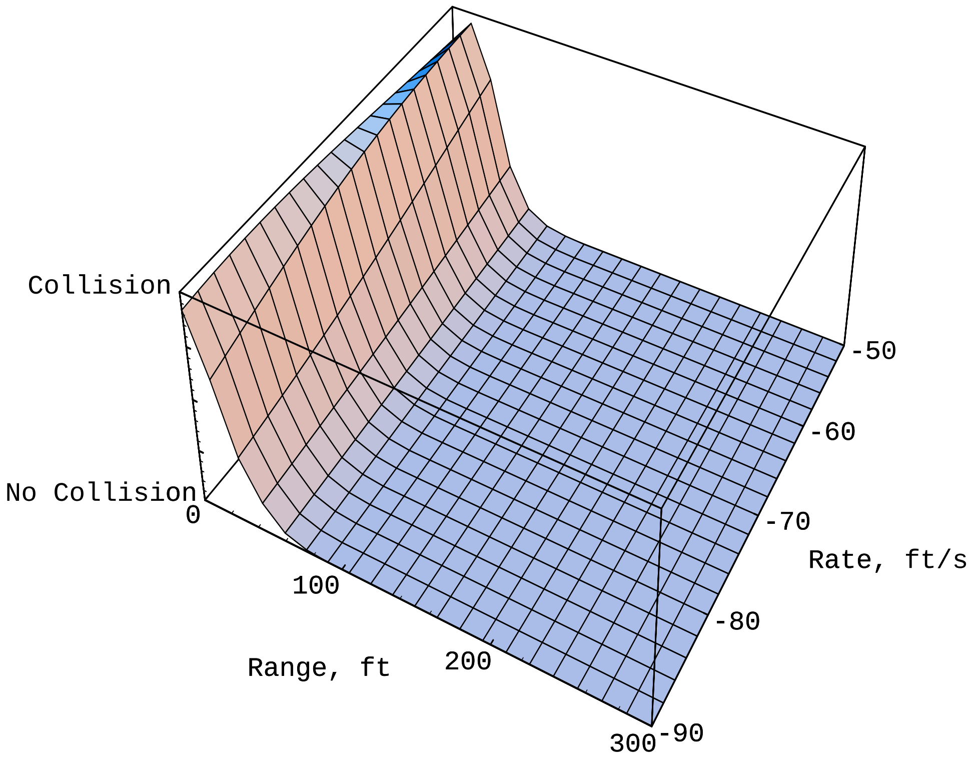


Figure 3.4: Decision Surface for Center Sensor, Case 4.

3.4.5 Cases 5 and 6

An attempt was then made to examine if a sensor scheme similar to a scanning radar or phased array detector would provide better results. As stated previously, the sensor returned the relative position, velocity, and heading of the closest approaching object. These facts were presented to the network as inputs in addition to the test vehicle's own speed and steering angle.

Two cases were trained. Each was trained on a dataset consisting of 40,000 test samples taken in simulation runs in which 80 collisions occurred. In the first case, it was hypothesized that the entire collision area could be bounded by a hyperellipsoid as opposed to a more complex type of neural network. For the data collected, this assumption seemed to work quite well. Paradoxically, the more complex net produced a system that did not perform as well in trials as the simpler system. This effect is the result of the complex network producing much more sharply defined alarm-producing regions that generalized poorly to collisions that did not exactly resemble those in the training set. Since the simpler mapping was somewhat constrained by the form of the network, it created broader alarm-producing regions that performed better during testing.

3.4.6 Cases 7 and 8

To evaluate the effect of the driving algorithm assumptions, the best performing neural net/ sensor array configuration: that developed as Case 5, was reevaluated using evaluation data derived under modified driver algorithms. In Case 7, the driver algorithm was modified to permit braking of the vehicle in the event of excessive divergence from the target trajectory. Evaluation of the neural network with this driver algorithm revealed the extreme sensitivity of the network to driver algorithm assumptions: the system failed miserably, missing over 80 percent of the the collisions and yielding a false alarm rate about 400 times as high as the collision rate. Such a system would clearly not be acceptable.

Case 8 represents a milder modification of the driver algorithm where, effectively, the driver looks farther down the road in making steering decisions, but no acceleration is used. In this case, the percentage of missed collisions was more acceptable: typically only about 16% but the false alarm rate was still unacceptable high – about 3:1, depending upon the choice of threshold.

3.5 Conclusions

In general, all procedures showed a high sensitivity to changes in the driving environment. The biggest result of increasing the complexity of the environment was that the system became increasingly more conservative in declaring alarms by waiting until an object was inside a rather small distance and at a reasonably high approach velocity before an alarm would be called.

In one network configuration, the scanning sensor provided good performance, but this configuration appears to be even more sensitive to model parameters than the fixed sensors. For example, as a result of the rule used to pick how the cars would choose to turn in a given situation, collisions always occurred when the test vehicle was turning hard, either to the left or the right – because collisions only occur when the car has swerved out of its lane and is trying hard to get back into it. By looking at the net mapping surface, one sees large regions where a crash can occur for those instances where the wheels are turned sharply, but no zone at all when the wheels are pointed straight on. Intuitively, this does not make sense. One would think that at least some collisions would occur if the driver was going basically straight ahead. However, the network does not reflect that result precisely because of the method used to generate data.

Lacking the information to produce a full reconstruction of the car's surroundings, the best correlation between sensor data and collision that can be derived is a statistical description. That is, the output of the data function is a statistical probability of collision. The drawback to a statistical description as opposed to a causal function based on a full reconstruction is that such a description embeds the statistics of the data used to generate it. While the function may produce fairly accurate predictions of accidents, it will do so only for accidents produced under the specific assumptions used to generate the data from which the correlation was drawn. Such correlations produce very poor results when applied to data generated by traffic scenarios with significantly different underlying assumptions.

After examination of a variety of simulation sets, it was concluded that any correlations derived in the long range will be extremely sensitive to driver algorithm and roadway shape statistics. As an extreme example of the sensitivity to driver/roadway characteristics, consider the following scenarios:

1. A set of driver and roadway characteristics is developed such that in every case where the radar saw an object, a collision would result.

2. A set of driver/roadway characteristics such that there are many close passes that the radar could sense, but no collision occur.

Now, if a network were trained from data from case (1), it would learn that any sighting always implies an impending collision. Now, if the network trained on (1) were tested on case (2), the network would call a false alarm every time an object came into view. Likewise, a network trained on case (2) and applied to case (1) would never anticipate any collisions. The point is that the statistical attributes of the model used to train the network are imbedded into the network, and no "general" result is obtained. Since the reason for using a model to generate learning datasets was originally that no applicable data exists, it is impossible to determine what the "right" driver/roadway characteristics are to imbed to get a "generalized" result.

The above observations suggest that the collision predictions can be made less sensitive to roadway statistics and driver algorithm if the sensing region – both spatial and temporal – around the test car is reduced to less than the driver reaction time. In previous studies, including this one to date, sensor information has been correlated in an effort to predict collisions at least a second in advance. In past studies, this was demanded by the end object of collision avoidance through automatic braking. Here, however, our objective is merely to obtain enough advance warning to be able to optimize the passenger restraints to minimize the injuries due to what is considered an inevitable collision. Indeed, it is unlikely that a warning lead time of more than about 100 msec would have much value in passenger restraint optimization.

Chapter 4

Short Range Sensing

4.1 Derivation of the Sensing Scheme

The sensing goal is to determine the relative position and velocity of any incoming object. In previous long-range sensing efforts, these quantities could be completely determined only under unrealistic assumptions. Instead of radar sensing, the sensor type is changed to a narrow beam sensor (ultrasonic, optical) that will return a distance along that narrow beam to the nearest object within sensor range. By limiting the sensors to a short range (in the range of 3 to 10 feet), it is possible to propose a scenario in which several assumptions might be made that are not valid for long range sensing. By employing these assumptions, all components of relevant target position and velocity are available.

Specifically, the short range assumptions are:

1. There is only one target in view at any time to each sensor array. (Or, at most one target is in view on each side of the car.)

2. Any target moves in a straight line at a constant velocity relative to the sensor equipped auto .

3. Any target has negligible rotational velocity relative to the sensor equipped auto.

Because the sensor range is very short, assumption 1 is well founded. The sensor range can be reduced to less than the distance that drivers normally like to keep between their cars and any other cars or obstacles. Any object that comes close enough to be sensed then has a high probability of being a hazard. Furthermore, collisions do not usually involve two oncoming cars hitting a third car simultaneously on the same side of the auto. Therefore assumption 1 is made.

Assumption 2 can be made because of the short amount of time for which the target is viewed. Even if the path of the target is curved, the duration of sensor contact is so brief that the path can be adequately represented by a linearization of the path for the duration of sensor contact. For example, if a target automobile is approaching at a relative velocity of 30 miles per hour (44 feet/sec), the target is only in view for about 0.07 seconds – a suitably small amount of time in comparison to the time scales at which events happen on the roadway typically about an order of magnitude greater, from previous results. Therefore, assumption 2 is made.

Likewise, the small time window used by the sensors justifies the use of assumption 3. Though the target may be turning or even spinning (e.g. an uncontrolled skid on icy roads), the angle of rotation covered in less than a tenth of a second is negligible for any kind of angular velocity that could be achieved in an automobile.

In short, the the sensing problem has been changed from the point of view used in the neural network portion of the study. Positions and velocities of objects that were

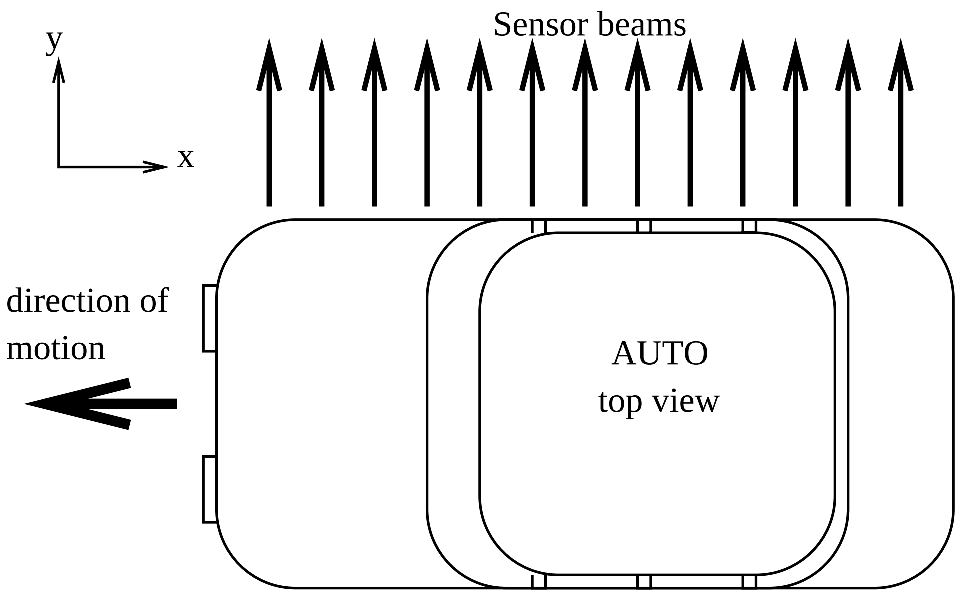


Figure 4.1: Schematic representation of lateral short-range sensors.

relevant to the situation but difficult to measure are made irrelevant by the shortening of sensor distance. Uncertainties in driver action are eliminated by viewing a small enough window so that the driver could have little or no effect. These assumptions reduce the system equations for all relevant targets to simply:

$$\begin{aligned} \dot{x} &= v_x \\ \dot{v}_x &= 0 \\ \dot{y} &= v_y \\ \dot{v}_y &= 0 \end{aligned}$$

where x and y are the relative position of the nearest point on the target and v_x and v_y are the relative velocity of the object. All of these states will be shown to be estimated by the new sensor configuration.

To determine the velocity of the incoming target, the following procedure can be taken. Take an array of parallel sensors mounted on

one side of the automobile, as in Figure 4.1. Call the output of the sensor at time T, Y and the output from one step later at time $T + \Delta T$, $\hat{Y}$. The profile of the incoming target is approximated by lines joining the points measured by each of the sensors. Displacement of $\hat{Y}$ with respect to Y can then be represented by the displacements Δx and Δy. Δx and Δy are determined by a least-squares fit such that the midpoints of each line segment match with the least error when the pattern is shifted.

Take any two adjacent sensors, n and $n+1$. When $\hat{Y}$ is displaced by Δx and Δy, the squared error for the midpoint of the segment is

$$e = \left[\frac{Y_n + Y_{n+1}}{2} - \left(\frac{\hat{Y}_n + \hat{Y}_{n+1}}{2} + \Delta x \frac{\hat{Y}_{n+1} - \hat{Y}_n}{l}\right) + \Delta y\right]^2$$

where l is the distance between sensors. The Δx and Δy that minimize e satisfy are

$$\frac{\partial e}{\partial \Delta x} = -2\frac{\hat{Y}_n + \hat{Y}_{n+1}}{2}\left[\frac{Y_n + Y_{n+1}}{2} - \left(\frac{\hat{Y}_n + \hat{Y}_{n+1}}{2} + \Delta x \frac{\hat{Y}_{n+1} - \hat{Y}_n}{l}\right) + \Delta y\right] = 0$$

$$\frac{\partial e}{\partial \Delta y} = 2\left[\frac{Y_n + Y_{n+1}}{2} - \left(\frac{\hat{Y}_n + \hat{Y}_{n+1}}{2} + \Delta x \frac{\hat{Y}_{n+1} - \hat{Y}_n}{l}\right) + \Delta y\right] = 0$$

However, these two equations are linearly dependent and alone cannot solve for Δx and Δy. Dividing out constants, each can be reduced to

$$2(\hat{Y}_{n+1} - \hat{Y}_n)\Delta x - 2l\Delta y = Y_n - \hat{Y}_n + Y_{n+1} - \hat{Y}_{n+1}$$

However, a second set of equations is also valid. Instead of shifting $\hat{Y}$ relative to Y, Y could be shifted relative to $\hat{Y}$:

$$2(Y_{n+1} - Y_n)\Delta x - 2l\Delta y = Y_n - \hat{Y}_n + Y_{n+1} - \hat{Y}_{n+1}$$

If the target is seen by more than two sensors, the same equation can be written for each set of adjacent sensors, leading to an over-determined system for Δx and Δy:

$$M\left\{\begin{array}{c}\Delta x\\ \Delta y\end{array}\right\} = b$$

which then can be solved by pseudoinverse methods.

$$\left\{ \begin{array}{c} \Delta x \\ \Delta y \end{array} \right\} = [M^T M]^{-1} M^T b$$

Note that the inversion of $[M^T M]$ only involves the inversion of a 2×2 matrix.

Once Δx and Δy are determined, the point of impact can be extrapolated by employing an integration of the system equations. The nearest point detected by the sensors (estimate of x and y) is moved forward in time with velocities $v_x \approx \Delta x/\Delta t$ and $v_y \approx \Delta y/\Delta t$ (where Δt is the time between Y and $\hat{Y}$) until an contact occurs. Averaging predictions from several time steps usually will lead to a good approximation of the time, point, direction, and velocity of impact.

Note that the scheme can be generalized so that the beams do not have to be strictly parallel. In the previous derivation, the x location of each of the data points was included implicitly through l, the distance between sensors. For the case in which the sensor beams are not parallel, the x-coordinate will be included explicitly – there is a vector $\hat{X}$ associated with $\hat{Y}$ and a vector X associated with vector Y determining the 2-dimensional position of each of the sensed points.

The same method of matched section midpoints is used. The midpoint of the segment from Y_n to Y_{n+1} is

$$\left\{ \frac{X_n + X_{n+1}}{2}, \frac{Y_n + Y_{n+1}}{2} \right\}$$

The equation for the line joining $\hat{Y}_n$ to $\hat{Y}_{n+1}n$ is

$$y = \frac{(\hat{Y}_{n+1} - \hat{Y}_n)}{(\hat{X}_{n+1} - \hat{X}_n)}(x - \hat{X}_n) + \hat{Y}_n$$

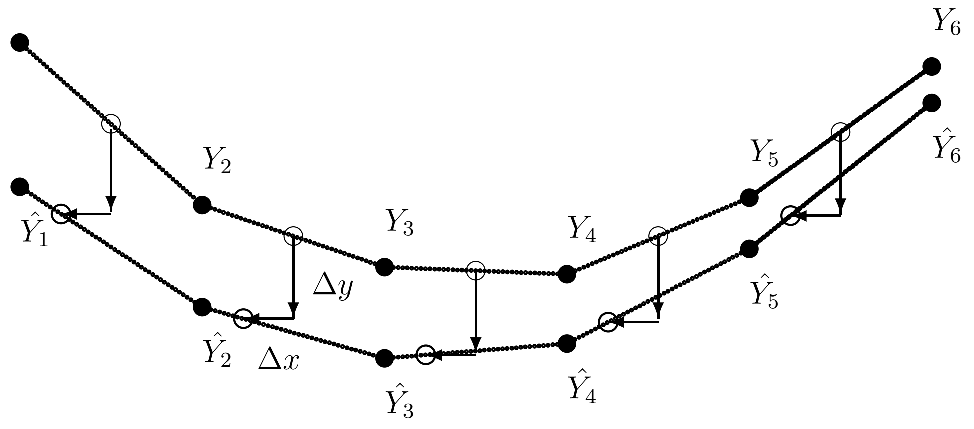


Figure 4.2: Graphic representation of sensor problem.

The shift of the segment from $\hat{Y}_n$ to $\hat{Y}_{n+1}$ corresponding to the midpoint of the section from Y_n to Y_{n+1} is then

$$\left\{\frac{X_n + X_{n+1}}{2}, \frac{(\hat{Y}_{n+1} - \hat{Y}_n)}{(\hat{X}_{n+1} - \hat{X}_n)}(\frac{X_n + X_{n+1}}{2} - \hat{X}_n - \Delta x) + \hat{Y}_n + \Delta y\right\}$$

The squared error between these points is then

$$e = \left[\frac{Y_n + Y_{n+1}}{2} - \frac{(\hat{Y}_{n+1} - \hat{Y}_n)}{(\hat{X}_{n+1} - \hat{X}_n)}(\frac{X_n + X_{n+1}}{2} - \hat{X}_n - \Delta x) - \hat{Y}_n - \Delta y\right]^2$$

Differentiating as before, e leads to the equation

$$\frac{\hat{Y}_{n+1} - \hat{Y}_n}{\hat{X}_{n+1} - \hat{X}_n}\Delta x - \Delta y = \hat{Y}_n - \frac{Y_n + Y_{n+1}}{2} + \frac{(\hat{Y}_{n+1} - \hat{Y}_n)}{(\hat{X}_{n+1} - \hat{X}_n)}(\frac{X_n + X_{n+1}}{2} - \hat{X}_n)$$

This equation degenerates into the form derived for parallel sensors if $X_n = \hat{X}_n$ and $X_{n+1} = \hat{X}_{n+1}$.

As an illustrative example, consider Figure 4.2. Y represents datapoints taken from a parallel sensor array at some instant in time. $\hat{Y}$ represents datapoints taken from the same sensors at an instant ΔT seconds later. The filled circles represent the

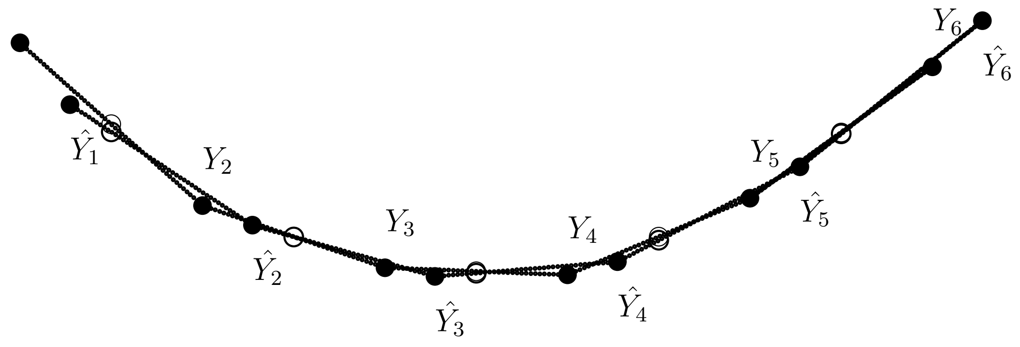


Figure 4.3: Superposition by prescribed displacement

measured data points. An estimation of the surface of the target is formed by joining the measured points by line segments. Since both Y and $\hat{Y}$ are measurements taken from the same surface, the movement of the target object can be found by finding the shifts Δx and Δy such that surface made by joining the points in Y will best match up in the surface defined by $\hat{Y}$.

The fit in each segment is assessed by comparing the open circles in Figure 4.2. On the Y segments, the open circles will simply lie at the segment midpoints. On the $\hat{Y}$ segments, however, the open circles lie on the point corresponding to the midpoint displaced by Δx down the line segment. The objective is to pick Δx and Δy such that the squared distance between corresponding open circles is minimized when the $\hat{Y}$ set of open circles is shifted back by Δx and Δy.

The problem in Figure 4.2 was solved for the best-fit shift using the formulas previously derived. $\hat{Y}$ was then moved by the prescribed shifts and plotted with Y in Figure 4.3. There is a close match between the two curves. One will note that the open circles correspond to the same points on the target surface in both Y and $\hat{Y}$.

4.2 Discussion of the Sensing Scheme

In most cases, this processing scheme will work well. However, there are several conditions under which there will be a greater amount of error in the measured velocities than normally.

The first of these observations has to do with the geometry of the target. The ideal targets are smooth and do not have corners – parabolas, circles, and so on. For these kinds of targets, the error between the target profile as approximated by straight lines and the actual profile is low, making the estimates of Δx and Δy very accurate. However, sharp corners introduce some error because when a corner lies between sensors, a straight line approximation cuts off the corner.

Another special geometry is the case in which the approaching object is completely flat. In this case, matrix $M^T M$ is singular, and there is no unique solution for Δx and Δy. To make sure that the scheme always gives an answer, one can assume $\Delta x = 0$ for this case and get an answer for Δy. This fix will usually still yield reasonable estimate of velocity.

Another point to be addressed concerns the spacing between the sensors and the sampling rate. Both of these parameters must be chosen carefully to minimize error introduced into Δx and Δy. In this processing scheme, if the actual displacement Δx is greater than half of the distance between sensors $(l/2)$, the target profile becomes extrapolated instead of interpolated. The estimates of Δx and Δy will then be unreliable. However, if a very small time step is taken so that the motion between steps is very small, measurement errors in the sensors themselves will dominate the calculations of Δx and Δy. In simulations, one foot sensor spacing and a 0.004 second time step were used successfully.

4.3 Simulation

To examine particular scenarios for the short range sensing scheme, a simulation program was developed. This simulation allows a large range of sensor layouts and collision scenarios. A number of example test runs are investigated in Appendix C.

An example implementation of the sensor processing algorithm is also included in Appendix C. The routine, written in Mathematica [22], takes two lists of sensor inputs from parallel sensors evenly spaced on the auto at a set distance apart and a set sensing range.

The test cases examined in the appendix represent several different accident situations that might occur for an auto equipped with a parallel sensing array down one side of the car. Each situation was chosen so as to test a different angle of approach and relative speed. For a summary of test runs, see Table 4.1.

Table 4.1: Sensing Scheme Example Runs

Case	actual $\frac{dx}{dt}$, ft/s	actual $\frac{dy}{dt}$, ft/s	est. $\frac{dx}{dt}$	est. $\frac{dy}{dt}$	$\frac{dx}{dt}$ error	$\frac{dy}{dt}$ error
0	-0.351	40.348	0.150	40.829	0.501	0.482
1	-40.036	39.751	-53.025	39.3116	-12.989	-0.4394
2	-1.950	38.689	-1.914	38.560	0.036	-0.129
3	12.602	8.585	17.239	9.810	4.637	1.225

Errors in the above simulations are due to the straight line approximation of target profile shape between sensors. For example, consider Figure 4.4. The heavier line

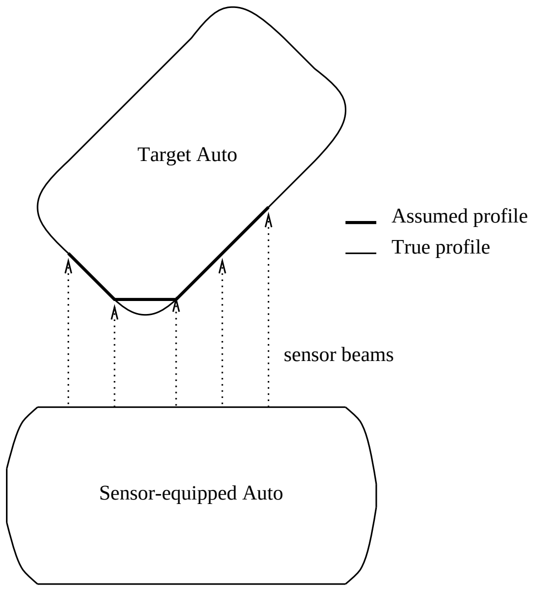


Figure 4.4: Example of linear approximation of target profile.

along the part of the side of the target's surface is the profile assumed by a connection of measured points by lines. In this example, the corner of the target falls between two sensor beams, creating an discrepancy between the true and assumed profiles. When the car moves a small amount during the instant between measurements, the assumed shape will be different, since the sensors are cutting the corner at different pooints. The best-fit displacements between the approximate profiles from two subsequent time steps then do not match exactly the true displacements because of error introduced by attempting to best-fit the cut-corner section of the profile.

4.4 Conclusions

Overall, the the sensing scheme appears to work well. Although the situations considered above are not an exhaustive study of this sensing scheme, they show several important points about its performance. First, an average of perceived velocity over several instants is more accurate than the instantaneous measured velocity. An average over several time steps seems to smooth out errors created by the straight line approximation of the target car's profile between measured points. Second, the velocity measured in the direction parallel to the sensing beams usually gives a more accurate estimation than the velocity measured perpendicular to the beams.

In some situations, however, the system can break down and be unable to estimate both components of relative velocity. The sensing scheme relies on the ability to pick up some characteristic curve in the profile of the target vehicle. The scheme then estimates the movement of the target over a small amount of time by estimating the translation of that feature.

If a target is in view of less than three sensors, the scheme cannot operate normally. Heuristically, the sensing scheme does not have enough information to identify a feature on the target. For just one sensor, the matrix M in Section 4.1 is singular and cannot be inverted to yield approach velocities. For two sensors, the matrix is invertible, but always gives a zero result for the perceived velocity perpendicular to the sensors. More testing is needed to see whether these cases are rare in practice or would prove to be a liability.

Other instances can also result in a poor target velocity estimation. Even if the target is in view of three or more sensors, the scheme will miscalculate velocity if a feature is not visible. Take for example the scenario in Figure 4.5. The target is only within the sensing range of three sensors at times t and $t+1$. Because of the geometry of the the target auto, the three active sensors return the same output at both times and therefore conclude not motion has taken place. The target has, however, moved a substantial directions both parallel and perpendicular to the sensing beams.

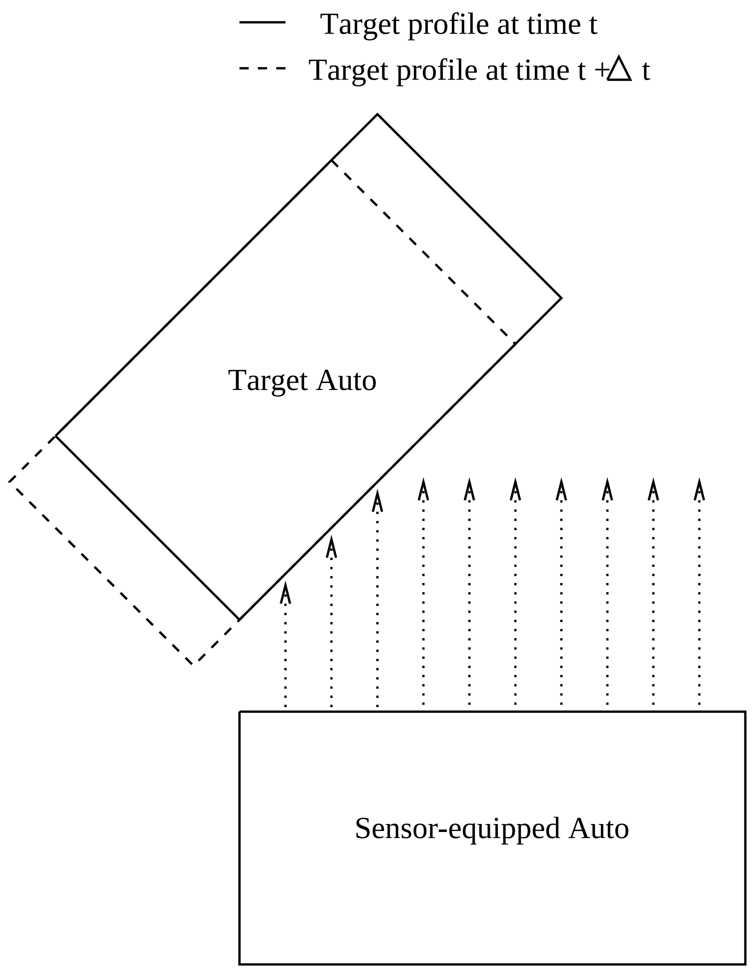


Figure 4.5: Example case where sensing scheme fails.

Also important to the assessment of the value of this scheme is consideration of the use of the information generated by it. Perhaps the device that will use the derived position and velocities will only need to know these quantities roughly – then, this algorithm would suffice. However, if velocities need to be estimated within strict tolerances, another scheme might need to be considered.

The some problems may be remedied to an extent by placing sensors more densely in the sensor array. When distance between measurement points is decreased, the straight-line approximation of the target profile between measured points becomes increasingly better. Likewise, the presence of more sensors decreases the likelihood that the scheme will fail because of contact by less than three sensors.

Other processing schemes might be investigated. The emphasis on the current scheme was to produce an algorithm that will have a closed-form answer and run very rapidly. More accurate schemes may be developed, perhaps, by minimizing the squared area between a nonlinear approximation of the target profile. Another possibility might be a scheme that has preconceived expectations of the shape of the target, or a scheme that builds up an idea of the total target profile by using data from every reading to define the silhouette.

Chapter 5

Restraint Optimization

5.1 Introduction

Creating an adequate pre-crash sensor only solves half the problem. To create a system that effectively reduces injury, the best possible use must be made of the extra lead time and information garnered by the sensors. Since pre-crash information has never before been considered for uses other than automatic braking, the creation of an optimal system of restraints necessarily involves an amount of original and preliminary investigation.

The first, and somewhat insidious, question to be addressed is the question, 'What *is* optimal performance?' Most normal engineering problems have fairly obvious performance goals – often, some kind of H-norm on the system outputs and applied control forces. There are obvious quantities that one would like to contain within an acceptable range. For the problem at hand, however, the goal is the more conceptual task of *reducing injury.*

How does one predict injury? Injuries are highly dependent upon the orientation, point, and magnitude of force application. Injury can also vary drastically between individual subjects that suffer very similar traumas due to individual physiologic differences.

How does one quantify injury? Upon pondering this question, one realizes that any quantification of human injuries is destined to be difficult and highly arbitrary. Is a broken leg worth more or less than a fractured pelvis? How many bruises does it take to equal one healthy cut? Different parts of the body can be injured in a multitude of different fashions and severities, all of which would have to be addressed somehow.

How does one create an adequate dynamic model of a human being? The human body is a very non-homogeneous structure with many articulations. For a general impact model, a simple lumped-mass model may not be adequate.

In an attempt to address some of the above questions, anthropomorphic test devices (crash dummies) have been created from the measurements of a large number of human subjects [6] [16]. Although ATD's are carefully sculpted to reflect the physiological topology and mass of the measured constituency, the dummies only may adequately reflect the shape and not dynamic character of the human body. In practice, ATD crash data has only a rough correlation to injuries in humans.

The goal of a good modeler is to create the simplest model which can be adequately correlated to empirical data. In this case, instrumented injury crashes with live human subjects are unavailable for obvious reasons. In contrast, instrumented dummy crashes are available but are of questionable worth (and still difficult to model). In the restraint optimization problem, adherence to the usual practice of good modeling is nearly impossible.

Perhaps one might decide that ATD collisions are adequate anyhow. In that case,

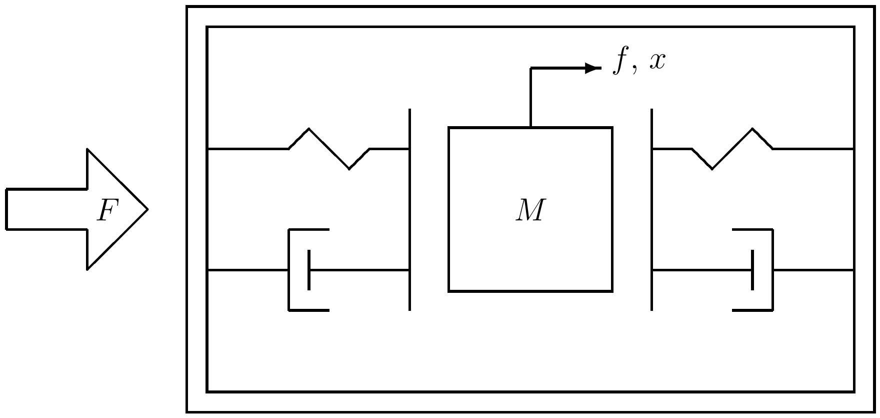


Figure 5.1: Simplified collision model.

a complex finite element model of the auto interior and ATD may be required to make a model that can be validated and used with confidence.

Rather than overpower the problem by producing a difficult and time consuming model, exactly the opposite approach was taken. Accepting the inaccessibility of a truly verified and validated crash model, a very simple model was created. The idea of this model would be to glean an intuitive insight into the problem, rather than to formulate a finalized design tool. From a simple model, the hope was that a paradigm for future development could be discovered before an intricate model might be needed.

The model here considered was made to reflect only a general class of problems and not to correspond directly to any one collision. Instead of a three dimensional model, a one-dimensional approach was used. Rather than measure non-linear dynamic characteristics for both the occupant and auto interior, the inside of the auto was modeled merely by linear springs and dampers, for simplicity's sake. The occupant was then represented by a single rigid mass. The restraints are reduced to a generalized force acting upon the occupant.

Figure 5.1 represents this collision model. M represents the auto's occupant. f represents the applied restraint forces. F represents the force impulse acting upon the car during collision. On either side of the occupant is a spring-damper system separated initially from the occupant by a gap.

By investigating this simple model, some fundamental performance results were indeed discovered.

5.2 Passive Restraints

In the previous section, passenger restraints were modeled by a generalized force, f, as represented in Figure 5.1. One way to optimize this generalized force is to choose f to be a control law of the form

$$f = f(x, \dot{x}, p_1, \ldots p_n)$$

where x is the relative displacement between the occupant and the automobile, $\dot{x}$ is the derivative of x with respect to time, and $p_1, \ldots p_n$ and n parameters that characterize the control law (*e.g.* stiffnesses, dampings, inertias). Then the parameters are chosen such that some measure of performance is optimized for the expected class of system inputs, $F(t)$.

For the approach examined in this section, the form of the generalized force was assumed to be a second set of springs, dampers, and gaps working in parallel with the existing set. Because this control scheme uses only passive components (that is, components that do not add energy to the system), this approach was designated to be the passive restraint system. This form was meant to be a rough model of the standard occupant restraints – seat belts – which assume a similar form in this paradigm.

Before, it was stated that restraint performance is difficult to quantify. Another advantage of employing a very simple model is that relative performance of different restraint strategies can be compared by the peak acceleration experienced by the occupant mass during a give collision. All optimizations in this chapter will seek to minimize the occupant peak acceleration (minimize the L_∞ norm of occupant acceleration).

The problem of optimizing the passive restraint model is an optimization problem over six parameters – two stiffnesses, two dampings, and two gaps. For any particular $F(t)$, the L_∞ norm is then merely a function of the six parameters. Given an $F(t)$, the parameters can be optimized using a standard nonlinear optimization procedure.

Specifically the equations of motion for the system are

$$\ddot{x} + C_1(x,\dot{x})/M + K_1(x)/M - f = -F(t)/M_{auto}$$

where the spring-and-damper seat belt model of f is

$$f = -(K_2(x)/M + C_2(x,\dot{x})/M)$$

The K's and C's are nonlinear functions of x because of the gaps:

$$\begin{aligned} C_1(x,\dot{x}) &= 0 \quad && g_{1,1} \geq x \geq g_{1,2} \\ &= c_{1,1}\dot{x} && x < g_{1,1} \\ &= c_{1,2}\dot{x} && x > g_{1,2} \end{aligned}$$

$$\begin{aligned} C_2(x,\dot{x}) &= 0 \quad && g_{2,1} \geq x \geq g_{2,2} \\ &= c_{2,1}\dot{x} && x < g_{2,1} \\ &= c_{2,2}\dot{x} && x > g_{2,2} \end{aligned}$$

$$\begin{aligned} K_1(x) &= 0 \quad & g_{1,1} \geq x \geq g_{1,2} \\ &= k_{1,1}x & x < g_{1,1} \\ &= k_{1,2}x & x > g_{1,2} \\ K_2(x) &= 0 & g_{2,1} \geq x \geq g_{2,2} \\ &= k_{2,1}x & x < g_{2,1} \\ &= k_{2,2}x & x > g_{2,2} \end{aligned}$$

where c_1's, k_1's, and g_1's are parameters for the auto body and c_2's, k_2's, and g_2's are parameters for the restraint system.

For the purposes of optimization, the above second-order ordinary differential equation has to be integrated numerically. In this case, the equation was broken down into a system of first-order equations

$$\begin{aligned} \dot{x} &= v \\ \dot{v} &= \tfrac{-1}{M}(C_1(x,v) + C_2(x,v) + K_1(x) + K_2(x) + MF(t)/M_{auto}) \end{aligned}$$

and discretized using the finite difference method

$$\begin{aligned} x_{k+1} &= x_k + \Delta v_k \\ v_{k+1} &= v_k - \tfrac{\Delta}{M}(C_1(x_k,v_k) + C_2(x_k,v_k) + K_1(x_k) + K_2(x_k) + MF(k\Delta)/M_{auto}) \end{aligned}$$

where Δ is the step size and k is the step number.

Rather than a norm on $\ddot{x}$, the performance measure is a measure of the global acceleration of the passenger. This measure is defined as

$$J_\infty(c_{2,1}, c_{2,2}, k_{2,1}, k_{2,2}, g_{2,1}, g_{2,2}) = \max_t (C_1(x,\dot{x}) + C_2(x,\dot{x}) + K_1(x) + K_2(x))/M$$

The decision was made to use a finite difference scheme rather than a Runga-Kutta integration or Runga-Kutta integration with predictor-corrector because J_∞ was found to have an acceptable level of error without the use of a more time-consuming algorithm.

Typically, collision force impulses are of a roughly trapezoidal form. For purposes of optimizing the six parameter system, several trapezoidal profiles were obtained.

One was taken from an actual sled test done at the University of Virginia. [19] Others were created to correspond roughly to those found in various literature sources. [4] [17]

For all acceleration profiles considered there arose several general qualitative results. Firstly, in all cases, the only set of springs and dampers that affected the peak acceleration were the ones first contacted during the collision. All subsequent contacts resulted in accelerations that were less that the maximum. Secondly, the optimum rattle space was zero inches of rattle space in all cases. Lastly, there was a wide range of stiffness-damping parameters that was near-optimal.

As explained above, only two of the original six parameters were responsible for determining the optimal performance (stiffness and damping of the restraints on the side of the car where collision takes place). Examining several test cases, one can see that two peak accelerations occur. A typical collision is illustrated in Figure 5.5. One is due to force transmitted to the occupant by the spring and damper before the collision. The second is the acceleration shock resulting from collision with the car body. The optimal stiffness and damping result when the height of these peaks are exactly equal. However one would have to know the exact energy and shape of the collision impulse before the collision to determine the most optimal values. From the graph of peak acceleration versus stiffness and damping (Figure 5.2), one can see that it is best (and indeed, nearly optimal) to over-estimate the energy of the collision and make the spring and damper harder than may be optimally necessary. Maximum acceleration is very much more sensitive to underestimation of impact energy. Heuristically, one can interpret this result as favoring the pre-impact peak over

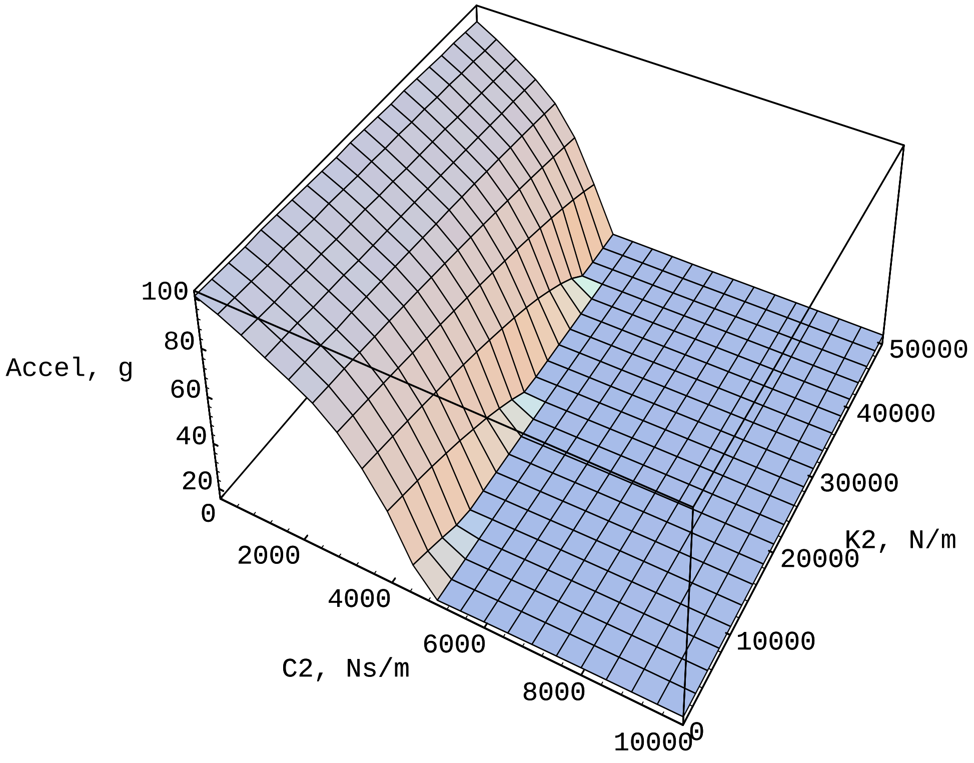


Figure 5.2: Peak acceleration versus stiffness and damping

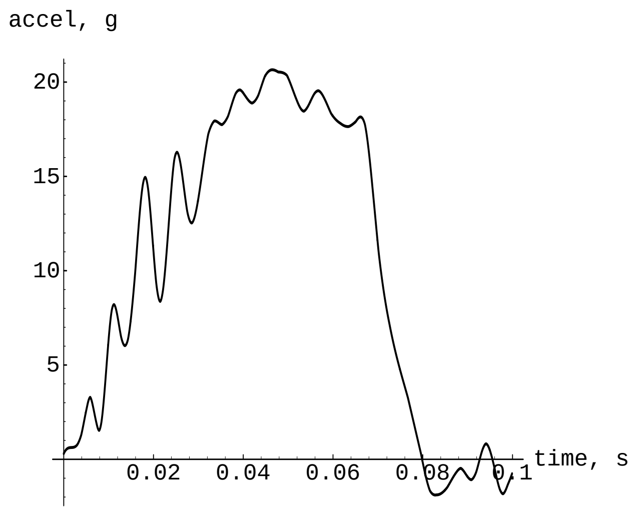


Figure 5.3: Example car body acceleration profile

the car body impact peak because the pre-impact peak is less sensitive to parameter variations. This result seems to be general, regardless of the force impulse. For additional examples, refer to Appendix D.

Included results are from a specific test case, `crsh.dat`, using an acceleration history measured from a 22.5 mph collision. For this collision, the optimal stiffness of 0 and optimal damping of 4771 N s/m resulted in a peak acceleration of 17.57 g's during the collision.

5.3 Best Case Restraints

An optimal solution for the restraints was found fairly easily when the restraints were assumed to follow the passive restraints paradigm. However, the question that then arises is the question of whether better performance can be obtained by starting with a different control form from the beginning. That is, is there some other kind of control law form, other than the above springs and dampers, that will give you

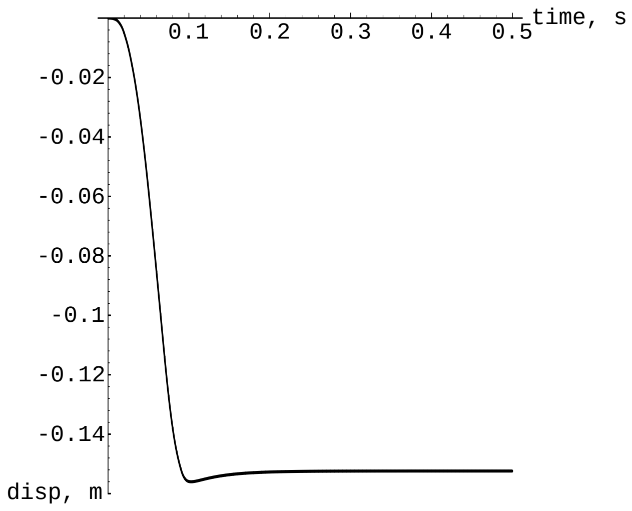


Figure 5.4: Occupant displacement relative to the car body

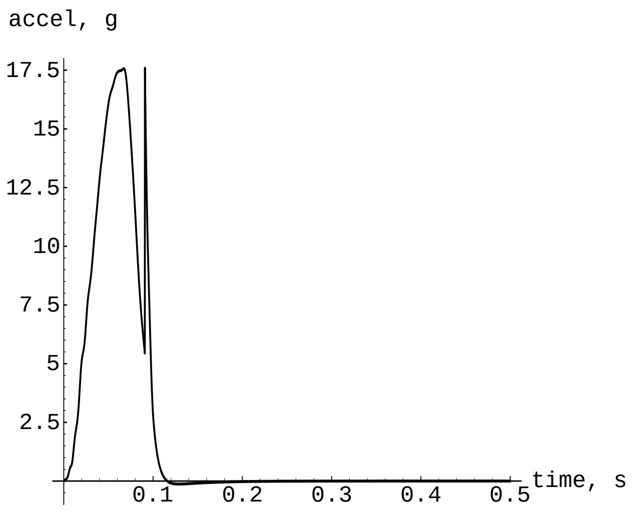


Figure 5.5: Occupant acceleration

much better performance than the optimized from of the springs and dampers? To answer this question, a lower bound on J_∞ was estimated by the use of a dynamic programming procedure [2].

To find this bound, the problem is changed back to the generalized force form:

$$\begin{array}{lcl} x_{k+1} & = & x_k + \Delta v_k \\ v_{k+1} & = & v_k - \frac{\Delta}{M}(C_1(x_k, v_k) + K_1(x_k) - f_k + MF(k\Delta)/M_{auto}) \\ J_\infty(f_1, f_2, \ldots, f_n) & = & \max_k (C_1(x_k, v_k) + K_1(x_k) - f_k)/M \end{array}$$

and the problem to find

$$\min_{f_1,\ldots,f_n} \max_k (C_1(x_k, v_k) + K_1(x_k) - f_k)/M$$

Instead of an optimization on a space of six parameters, the problem is transformed into the much more difficult optimization of control force f at n times (where n is the number of steps in the simulation).

Note that any solution to this problem can only serve as a lower bound – not an actual control law. The solution to the problem requires *a priori* knowledge of the exact shape of the force impulse $F(t)$, something which is never available in actuality. The purpose here is only to have a yardstick by which to measure the performance of other schemes.

To solve this problem, the program was treated as an exercise in dynamic programming. The cost function J_∞ was converted into the recurrence relation

$$J_k(x_k, v_k, f_k) = \max[|(C_1(x_k, v_k) + K_1(x_k) - f_k)/M|, J_{k+1}(x_{k+1}(x_k, v_k, f_k)), v_{k+1}(x_k, v_k, f_k)]$$

with the boundary condition

$$J_n = |(C_1(x_k, v_k) + K_1(x_k))/M|$$

now, denote

$$\begin{aligned}U_k &= \min_f J_k(x_k, v_k, f_k)\\ &= \min_f\{\max[(C_1(x_k, v_k) + K_1(x_k) + f_k)/M, J_{k+1}(x_{k+1}(x_k, v_k, f_k)), v_{k+1}(x_k, v_k, f_k)]\}\end{aligned}$$

and let the f that minimizes U_k be known as $f_k^*(x_k, v_k, k)$.

Briefly, U_k and f_k^* are solved for backwards from the boundary back to the first step of the problem. At the first step of the problem, U_1 fulfills the same conditions as J_∞. f_k^* is the sequence of control forces that realizes J_∞. A solution to the lower bound on the restraint problem is then found on discretized sections of U_k and f_k^* at each time step proceeding back to the beginning of the problem. For specifics of the numerical solution of the dynamic programming version of the restraint problem, see Appendix E.

The main result of these solutions was that the lower bounds to the solution were much lower than optimal passive restraints. For example in case `crsh.dat`, the lower bound on occupant acceleration was only 8.77 g's as opposed to the 17.57 g's pulled with the passive restraints (See comparison in Figure 5.6). This drastic improvement in performance is due to the different method of restraint applied in the optimal case. The solution for f_k^* derived by the dynamic programming problem starts accelerating the occupant at a constant rate as soon as the simulation begins, before there is really even any contact. f_k^* is then adjusted throughout the problem such that it opposes the force applied by the doors just enough to maintain the same acceleration that it prescribed at the beginning of the problem. Then, at the very end of the problem, the acceleration slacks off.

From an intuitive perspective, the optimal solution is makes sense. During the

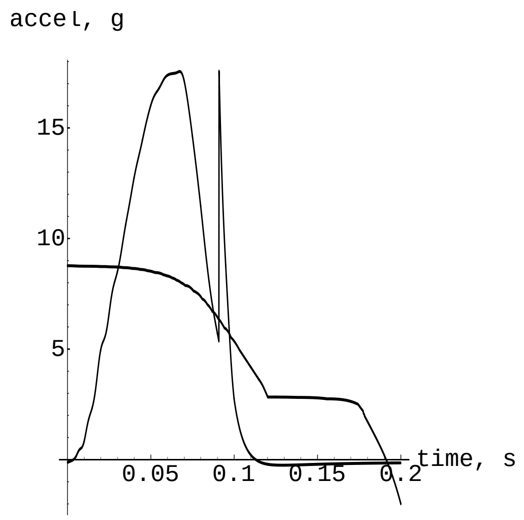


Figure 5.6: Comparison of optimal active and passive acceleration

collision, the occupant has to go through a mandatory amount of acceleration such that he will stay with the car. Since the boundary case knows the acceleration *a priori*, the optimal solution is just a constant acceleration. The occupant ends up at pretty much the same spot as he would in the passive restraint case (give or take an inch), but his peak accelerations have been chopped off and dumped into what would be times of lower acceleration in the passive restraint model.

5.4 Conclusions

From the results of the passive restraint case and the lower bounds case, one can draw several important intuitive conclusions:

- There is no point in using a pre-crash sensor with a system such as the passive restraint law. Near optimal performance for any fairly high speed collision can be achieved by setting the restraint stiffnesses and dampings conservatively high and letting the gap distances equal zero.

- Much better performance can be affected by the use of active components in the restraint system. The optimal solution f_k^* employs forces that could never come from springs and dampers. Although f_k^* was derived by exact knowledge of the $F(t)$, it may be possible to achieve better performance through an approximation derived by the pre-collision sensor outputs.

The drawback of the optimal solution is that it is intrusive, violating the initial tenets of the study. To realize the optimal acceleration, some large forces would have to be applied to the occupant before collision. Carefully controlled forcing as required in the optimal solution could only be obtained by measures such as controlled air-bag firings or anticipatory accelerations of the seat itself. In such instances, the driver would have to be forced to lose control of the automobile, actually causing accidents if a false-alarm situation were to occur. Therefore, this study concludes that the best restraints in the collision model studied are stiff fixed-parameter restraints, since improvements on performance imply unacceptable intrusive devices.

However, these conclusions must still be considered with the following reminder – the entire crash model was a very crude and unvalidated representation of automobile collisions. Any conclusions generated by the model do not necessarily apply to the problem of automobile collisions due to the drastically more complicated nature of actual collisions. Any results from the low-order model should be used for intuitive reasons only.

Chapter 6

Conclusions

The main accomplishment of this work was the development of a formal structure with which to study the concept of pre-collision sensing.

Previous work was done without considering whether the assumptions made in modeling the system were valid. An examination of that work shows that the simplifications used were not necessarily true and led to poor performance.

A sensor configuration similar to that in existing literature was then considered. The major change from previous work was that the objective was changed from automatic braking to restraint optimization, and the collision warning law was to be determined empirically by neural network methods. An examination of this system pointed to the inherent uncertain nature of the system when the objective was to predict collisions at times around one second in advance.

The frame of reference was then altered so that the sensors only scanned over a short range. With this change, many uncertainties of the longer range system were removed. In this environment, a pre-crash sensing system was derived and tested in simulation with some success. Indeed, the short range perspective would seem to be the most useful method with which to solve the pre-crash sensing problem.

The problem of optimal passenger restraint during collisions was then considered. Although an adequate model was difficult to come by due to the inherent nature of the system, a rudimentary model was developed and tested. Work on this rough model implied that the simple optimization of existing seatbelt restraints is not a worthwhile application for pre-crash sensing.

Significant performance improvements might be affected by developing new restraint devices that could apply active forces upon the passengers. However, the basic premise of this research was to study optimal passive restraints because of the hazardous nature of active measures (e.g. automatic braking) in a false alarm situation. Any implementation of the sensing scheme developed herein as a trigger for active restraint measures could only be undertaken if the false alarm rate was established to be zero.

Several questions yet remain to be answered. The first of these questions is implementation of the short range sensing algorithm. Although the processing algorithm is simple and easily implemented by digital signal processing hardware, existing sensor technology might not be adequate to gather the required information. A survey of existing short range ultrasonic, optical, and radar sensors must be undertaken in the future. Possibly a new sensor will have to be developed that will work in conditions of rain, cold, visual obscurity, and so on.

Secondly, the restraint optimization problem must be addressed in better detail. A model that can be validated against experimental data must be developed so that confidence can be put placed in computational studies. Then, mechanisms for the application of active forces on the occupants need to be either identified or invented so that the benefits of pre-crash sensing might be garnered.

Bibliography

[1] M. Alvisi, P. Deloof, W. Linss, G. Preti, and A. Rolland. Anticollision radar: State of the art. In *Advanced Telematics in Road Transport, Volume 2*, Brussels, February 4-6, 1991. Proceedings of the DRIVE Conference.

[2] R. Bellman and R. Kalaba. *Dynamic Programming and Modern Control Theory.* Academic Press, 1965.

[3] E. Dull, R. Bosh, and H. J. Peters. Collision avoidance system for automobiles. Technical Report SAE 780263, Society of Automotive Engineers, 1978.

[4] R. I. Emori. Analytical approach to automobile collisions. In S. H. Backaitis, editor, *Reconstruction of Motor Vehicle Accidents: A Technical Compendum.* Society of Automotive Engineers, 1990.

[5] J. B. Flannery. Automatic braking by radar. Technical Report SAE 740094, Society of Automotive Engineers, 1974.

[6] General Motors Corporation. *Anthropomorphic test dummy, Final Report*, October 1974. DOT-HS-299-3-569.

[7] J. B. Hopkins et al. A microwave anticipatory crash sensor for automobiles. Technical Report SAE 720423, Society of Automotive Engineers, 1972.

[8] H. White K. Hornik, M. Stinchcombe. Multi-layer feed-forward networks are universal approximators. *Neural Networks*, pages 2:359–366, 1989.

[9] G. S. Kaplan and F. Sterzer. Dual-mode automobile collision avoidance radar. Technical Report SAE 750087, Society of Automotive Engineers, 1975.

[10] T. Makino and D. Sato. Development of a radar sensor for inflatable occupant restraint system. Technical Report SAE 720422, Society of Automotive Engineers, 1972.

[11] K. Minami et al. A collision avoidance warning system using laser radar. Technical Report SAE 881859, Society of Automotive Engineers, 1988.

[12] J. K. Pollard. Evaluation of the vehical radar safety system's rashid radar safety brake collison warning system. Technical Report DOT-TSC-HS-802-PM-88-2, National Highway Traffic Safety Administration, February 1988.

[13] G. F. Ross. BARBI, a new radar concept for pre-collision sensing. Technical Report SAE 740574, Society of Automotive Engineers, 1974.

[14] G. F. Ross. A baseband radar system for auto braking applications. Technical Report SAE 780262, Society of Automotive Engineers, 1978.

[15] D. E. Rumelhart and J. L. McClelland. *Parallel distributed processing : explorations in the microstructure of cognition.* MIT Press, 1986.

[16] L. W. Schneider, D. H. Robbins, M. A. Pflug, and R. G. Snyder. Development of anthropometrically based design specifications for an advanced adult anthro-

pomorphic family, volume I. Technical Report DTN22-80-C-07502, National Highway Traffic Safety Administration, 1983.

[17] M. Shaw. Countermeasures for side impact – final report. Technical Report DOT-HS-806 319, National Highway Traffic Safety Administration, August 1982.

[18] A. C. Stein, D. Ziedman, and Z. Parseghian. Field evaluation of a nissan laser collision avoidance system. Technical Report DOT HS 807 375, National Highway Traffic Safety Administration, January 1989.

[19] M. A. Townsend and A. P. Allan. Simulation of a constrained spherical pendulum. University of Virginia, July 1992.

[20] W. C. Troll. Automotive radar brake. Technical Report SAE 740095, Society of Automotive Engineers, 1974.

[21] P. D. Wasserman. *Neural computing: theory and practice.* Van Nostrand Reinhold, 1989.

[22] S. Wolfram. *Mathematica : a system for doing mathematics by computer.* Addison-Wesley Publishing Co., 1991.

[23] R. E. Wong et al. Collision avoidance radar braking systems investigation – phase II study, volume II – technical report. Technical Report DOT-HS-820 020, National Highway Traffic Safety Administration, 1976.

Appendix A

Neural Network Overview

A.1 General Setup

A feed-forward neural network is one empirical method for deriving a functional relationship between two sets of data. A schematic representation of a neural network is contained in Figure A.1. Inputs feed into the input layer of network. At each numbered node, all inputs are combined in a weighted summed. The output of each node is some nonlinear function of the node's weighted sum. The term *feed forward* implies that each node receives input only from nodes on layers lower than itself so that the network's output can be determined by only one evaluation of each of the nodes in the network.

Consider a network with $n + 1$ nodes, $m + 1$ inputs, and one output. Call the output of the i^{th} node z_i. Let a_i denote the weighted sum of all the inputs into the i^{th} node. For the networks used in this study,

$$a_i = x^T W_i x$$

where W_i is an upper triangular matrix of weighting values used in the weighted sum,

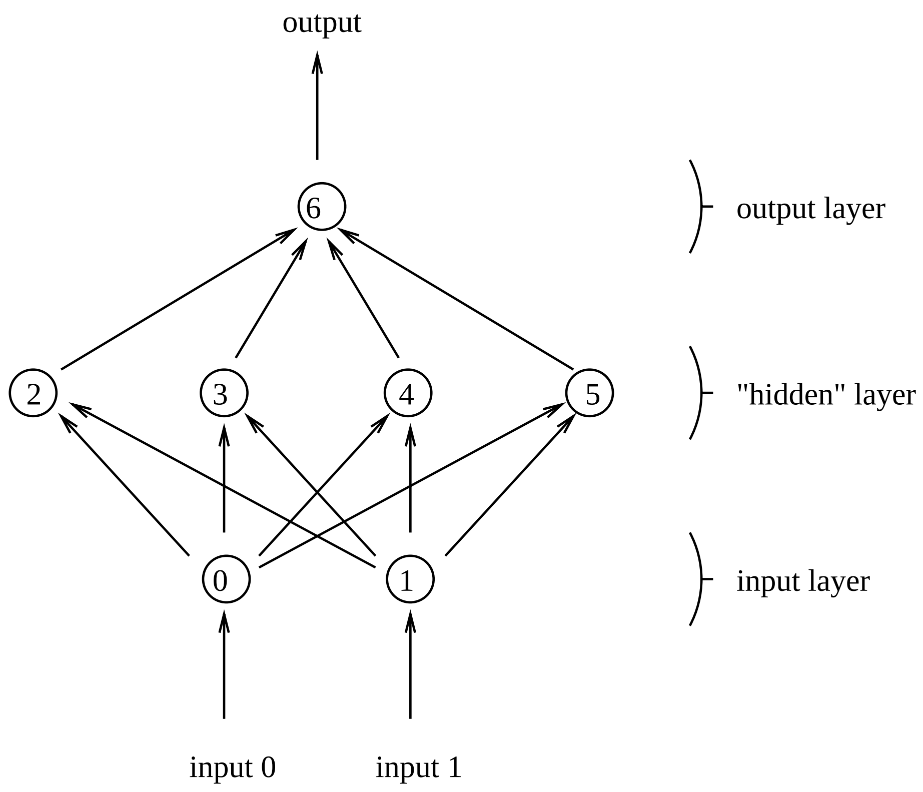


Figure A.1: Schematic representation of a neural net.

and x is the vector

$$x = \{1, z_0, z_1, z_2, z_3, \ldots, z_n\}^T$$

For a feed-forward network, all elements of W_i associated with nodes on the same layer or in higher layers are always zero. If the node in question is on the input layer, x is instead

$$x = \{in_0, in_1, \ldots, in_m\}^T$$

The weighted sum a_i is a complete quadratic in the inputs to the node. The output of the i^{th} node is

$$z_i = \arctan a_i$$

The output of the entire network is evaluated by computing the output of each node sequentially, starting from the nodes in the input layer and working upwards through the hidden and output layers.

This network configuration is a modification on the form typically found in [21] and [15]. In these references, a_i typically has the form

$$a_i = w_i^T x$$

where w_i is a vector instead of a matrix. Activation a_i is a linear function of the inputs rather than a quadratic function. The quadratic function was used in this study because it was found in practice to yield functional relationships with lower error on the training data examples than networks using linear functions of the inputs.

In [8], it is proved that a network with a linear form for a_i and an infinite number of nodes in the hidden layer can identically represent any bounded functional relationship between inputs and outputs. This result is not surprising. Input nodes scale possibly unbounded inputs onto finite range of values. The nodes on the hidden layer then form a basis on the space of functions mapping $\mathbb{R}^m$ onto $\mathbb{R}$, analogous to an infinite series of polynomials forming a basis in a Taylor expansion. Output nodes form a weighted sum of these basis functions and scale the result onto a bounded range of outputs. The proof in [8] is also sufficient to show the same result for the quadratic form of a_i, since the linear form is a special case where the weights associated with all second-order terms are set to zero.

A.2 Training

With a finite number of hidden-layer nodes, the network will not be able to exactly represent any functional relationship. However, the purpose of using the net in this instance is not to represent a previously known functional relationship but to derive an approximation of some unknown functional relationship. By the adjustment of the values in the W weighting matrices of the nodes, a low error approximation relationship can be formed from known examples of the relationship such that the function can be interpolated between known points. This task is exactly a nonlinear regression.

The usual process for training a neural network is known as *backpropagation*. This process is described in detail in references [21] and [15]. This process is a recursive form of gradient descent optimization. Weights leading into the input nodes are set *a priori* so that the inputs from the training dataset are scaled, for the case of the arctan function, on a range of -1 to 1. All other network weights are initially set at some small initial values. In this study, initial weights were chose randomly on $[-1, 1]$. The process proceeds as follows.

1. One example of the input-output relationship is taken at random from the collection of data. The output of the network is evaluated using the current values of weight, and an error, e is computed that is the squared difference between the desired output and the actual output.

2. Using the chain rule of partial derivatives, the partial derivative of e with respect to any element in the weighting matrix of the output node is calculated. Through continued use of the chain rule, the partial derivatives of e with respect to a in the hidden layer nodes is computed.

3. Again using the chain rule, the derivatives of e with respect to the weights

leading into the hidden layer nodes are computed.

4. Each non-zero value of the weighting matrices associated with hidden and output nodes are then modified by

$$W_i(new) = W_i(old) - \delta\, \partial e / \partial W_i(old)$$

where δ is some small arbitrary positive number.

5. Repeat from step (1) until e is acceptably small for some arbitrary number of iterations of the process.

A.3 Considerations for Crash Prediction

To predict a collision, a set of sensor measurements is to be mapped to an output of either a 1, denoting an impending collision, or a -1, denoting no collision. However, since the functional relationship is not known a priori, it is not obvious what set of sensor measurements are necessary such that an adequate functional relationship between sensors and collisions will exist. Many configurations of network inputs might need to be attempted before a suitable relationship reveals itself, and no suitable functional relationship is guaranteed to exist for a given physical implementation of sensors.

Compounding the problem of not knowing what inputs to give the neural network is the problem of what area of space to take those inputs over. Since the neural network is basically a high-dimensional curve fit, the network could can reflect a functional relationship interpolated between known datapoints. However, the performance of the network in regions of the input space where datapoints have not been taken is unknown and of no value for predicting collisions.

Another consideration is the problem of obtaining data with which to train the network. For a collision prediction problem, the physical process of data collection would be to equip a large number of autos with a given sensor configuration. These cars would then be driven over a large range of driving conditions, and some would eventually have collisions. This data could then be used to train the network. Such an approach would be costly in both time and money, and since no functional relationship can be guaranteed for any physical array of sensors using this empirical approach, no network valuable for prediciting collisions could be assured. A dataset might be derived by simulation, but without knowledge of a functional relationship or a large amount of crash data, no confidence can be had that the relationship derived from model training sets will have any correlation to actual driving situations.

Appendix B

Neural Net Simulation Data

B.1 Case 1

training conditions: 1 obstacle vehicle and no roadside obstacles. Vehicle speeds between 34 and 45 MPH.

evaluation conditions: 1 obstacle vehicle and no roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensors: 3 at -30°, 0°, and 30°. subtended angle = 4° for each and range = 300 feet.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	71	24	17	0.132 sec
1000	0.40	68	27	22	0.168 sec
1000	0.20	62	19	15	0.153 sec
1000	0.00	52	42	29	0.190 sec
1000	-0.20	58	50	34	0.180 sec
1000	-0.40	67	92	53	0.259 sec
1000	-0.60	72	106	59	0.290 sec
1000	-0.80	53	175	46	0.377 sec

B.2 Case 2

training conditions: 1 obstacle vehicle and no roadside obstacles. Vehicle speeds between 34 and 45 MPH.

evaluation conditions: 1 obstacle vehicle and no roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensors: 3 at -30°, 0°, and 30°. subtended angle = 30° for each and range = 300 feet.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	55	73	42	0.375 sec
1000	0.40	60	75	56	0.437 sec
1000	0.20	20	104	62	0.467 sec
1000	0.00	60	114	55	0.515 sec
1000	-0.20	60	142	55	0.578 sec
1000	-0.40	58	149	54	0.551 sec
1000	-0.60	67	283	64	0.592 sec
1000	-0.80	57	589	56	0.720 sec
1000	-0.90	75	930	75	0.819 sec

B.3 Case 3

training conditions: 1 obstacle vehicle and no roadside obstacles. Vehicle speeds between 34 and 45 MPH.

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensors: 3 at -30°, 0°, and 30°. subtended angle = 30° for each and range = 300 feet.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.40	123	2045	85	0.615 sec
1000	0.20	105	2162	73	0.762 sec
1000	0.00	123	2214	94	0.745 sec
1000	-0.20	104	2500	79	0.722 sec
1000	-0.40	120	2626	93	0.990 sec
1000	-0.60	118	2767	94	0.978 sec
1000	-0.90	125	3970	121	1.35 sec

B.4 Case 4

training conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensors: 3 at -30°, 0°, and 30°. subtended angle = 30° for each and range = 300 feet.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	137	211	96	0.309 sec
1000	0.40	129	292	103	0.338 sec
1000	0.20	113	324	94	0.411 sec
1000	0.00	116	403	95	0.436 sec
1000	-0.20	124	483	109	0.516 sec
1000	-0.40	116	577	106	0.543 sec
1000	-0.60	108	778	101	0.621 sec
1000	-0.80	104	1133	93	0.739 sec

B.5 Case 5

training conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet. Small neural net configuration.

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensor: Scanning radar with 300 foot sensitivity range.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	73	157	60	0.683 sec
1000	0.40	73	172	56	0.740 sec
1000	0.20	63	192	54	0.744 sec
1000	0.00	72	215	68	0.774 sec
1000	-0.20	75	241	69	0.757 sec
1000	-0.40	79	266	77	0.742 sec
1000	-0.60	65	296	65	0.731 sec
1000	-0.80	70	364	70	0.901 sec

B.6 Case 6

training conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet. Large neural net configuration.

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet.

sensor: Scanning radar with 300 foot sensitivity range.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	71	85	16	0.444 sec
1000	0.40	63	140	25	0.454 sec
1000	0.20	87	176	39	0.541 sec
1000	0.00	72	200	38	0.603 sec
1000	-0.20	72	217	39	0.615 sec
1000	-0.40	82	268	51	0.741 sec
1000	-0.60	71	316	45	0.742 sec
1000	-0.80	70	630	55	0.914 sec

B.7 Case 7

training conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet. Small neural net configuration.

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet. **Driving algorithm modified to include acceleration and deceleration.**

sensor: Scanning radar with 300 foot sensitivity range.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	6	82	1	0.85 sec
1000	0.40	4	120	1	0.60 sec
1000	0.20	5	157	0	–
1000	0.00	5	203	1	0.35 sec
1000	-0.20	8	235	0	–
1000	-0.40	10	305	3	0.21 sec
1000	-0.60	4	427	2	0.10 sec
1000	-0.80	4	595	3	0.43 sec

B.8 Case 8

evaluation conditions: 1 obstacle vehicle and 10 roadside obstacles. Vehicle speeds between 34 and 45 MPH. Range of roadside obstacles from roadside = 30 to 70 feet. **Driving algorithm modified to increase driver anticipation.**

sensor: Scanning radar with 300 foot sensitivity range.

runs	threshold	collisions occurred	alarms generated	collisions predicted	warning interval
1000	0.60	30	84	25	0.706 sec
1000	0.40	32	77	24	0.581 sec
1000	0.20	33	93	28	0.707 sec
1000	0.00	33	126	25	0.790 sec
1000	-0.20	38	132	33	0.708 sec
1000	-0.40	29	153	25	0.718 sec
1000	-0.60	40	181	34	0.790 sec
1000	-0.80	34	284	32	0.859 sec

Appendix C

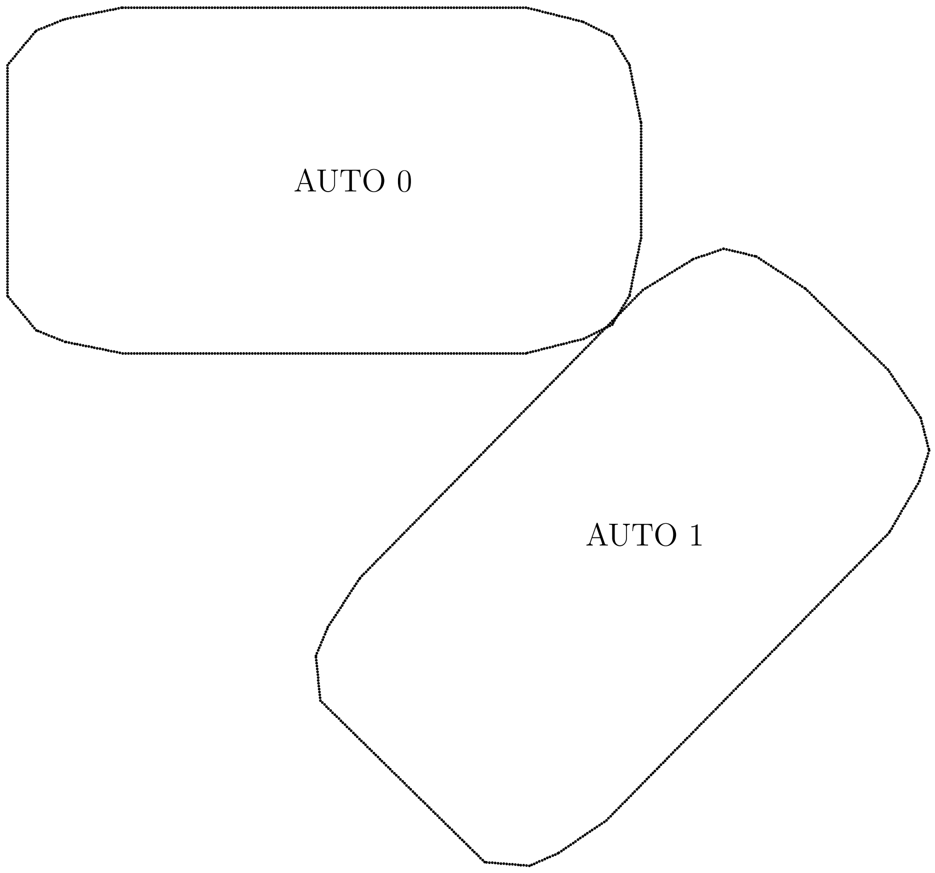
Short-Range Simulation

C.1 Sample Sensor Processor

Code written in Mathematica [22].

```
BeginPackage["Bump`"]

BumpIt::usage = "BumpIt[list1,list2,delta,limit]"

Begin["`Private`"]

BumpIt[l1_,l2_,d_,limit_]:=
        Module[{a,b,m,k,n,flag,ans,hook},
        a={}; b={};
        n=Length[l1];
        For[k=1,k<n,k++,
                flag=1;
                If[ (l2[[k+1]]>=limit) || (l2[[k]]>=limit) , flag=0];
                If[ (l1[[k+1]]>=limit) || (l1[[k]]>=limit) , flag=0];
                If[ flag==1,
                        a=Append[a,{2(l2[[k+1]]-l2[[k]]),-2 d}];
                        b=Append[b,{l1[[k]]-l2[[k]]+l1[[k+1]]-l2[[k+1]]}];
                        a=Append[a,{2(l1[[k+1]]-l1[[k]]),-2 d}];
                        b=Append[b,{l1[[k]]-l2[[k]]+l1[[k+1]]-l2[[k+1]]}];
                        hook=l1[k]-l2[k]; ]

        ];
        If[Length[b]==0,ans=Null,
                m=Transpose[a] . a;
                If[Length[SingularValues[m][[2]]]<2,
                        ans={0,hook},
                        ans=Inverse[m] . Transpose[a] . b ]; ];
        ans
]

End[ ]
EndPackage[ ]
```

C.2 Case 0



target	*relative* $\partial x/\partial t$, fps	*relative* $\partial y/\partial t$, fps
AUTO 1	-0.351	40.348

Sensor Array Performance

- Measured Mean Vx = 0.149876 fps
- Measured Mean Vy = 40.8296 fps

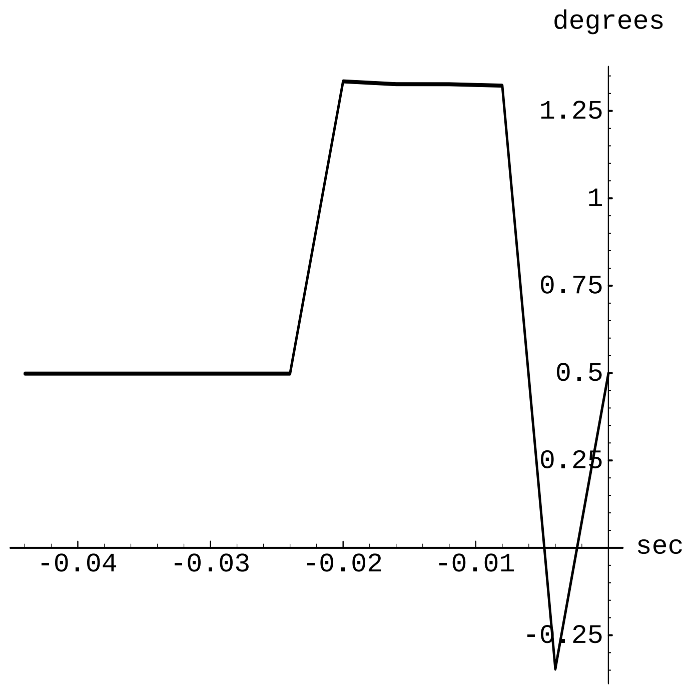


Figure C.1: Case 0 - Error in predicted approach angle

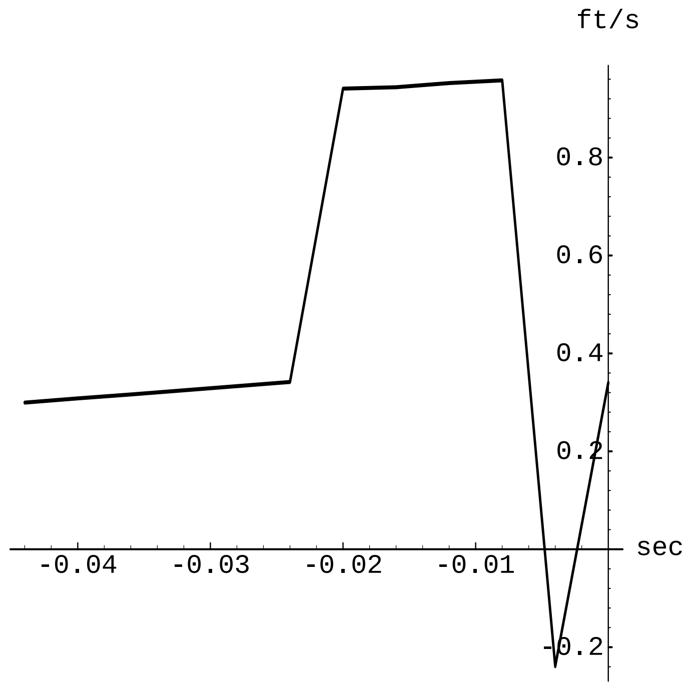
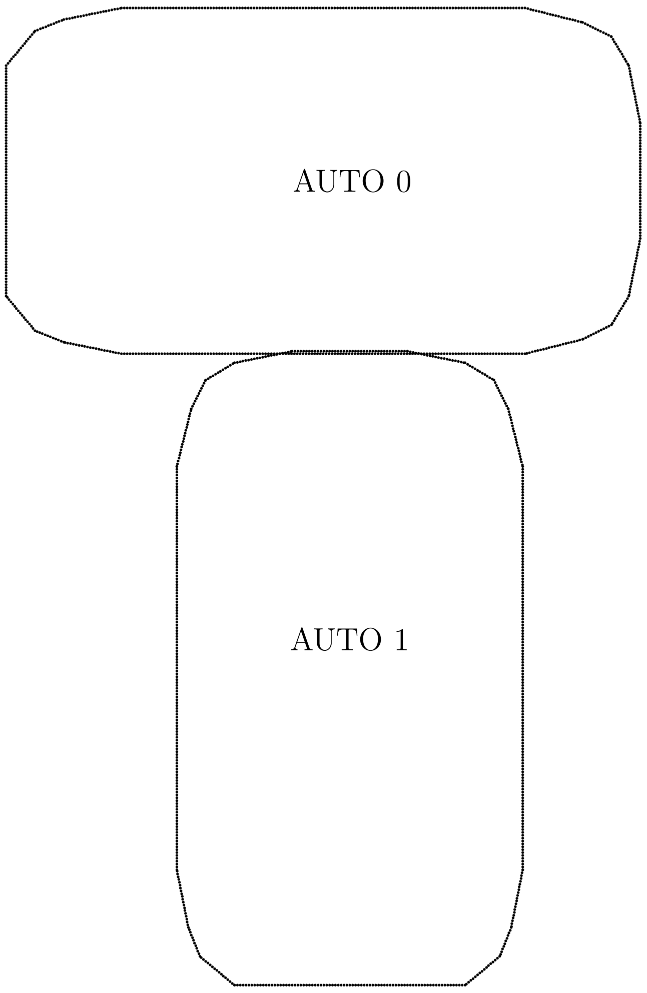


Figure C.2: Case 0 - Error in predicted approach velocity magnitude

C.3 Case 1



target	*relative* $\partial x/\partial t$, fps	*relative* $\partial y/\partial t$, fps
AUTO 1	-40.036	39.751

Sensor Array Performance

- Measured Mean Vx = -53.025 fps
- Measured Mean Vy = 39.3116 fps

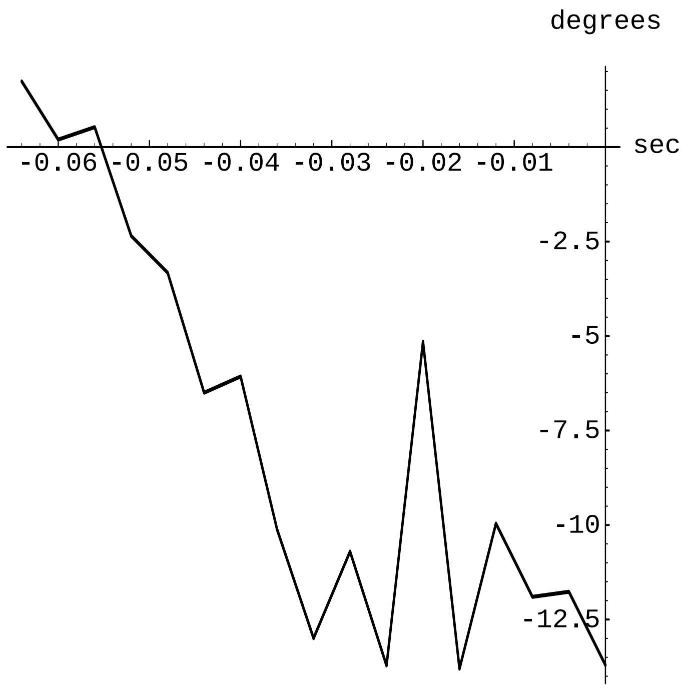


Figure C.3: Case 1 - Error in predicted approach angle

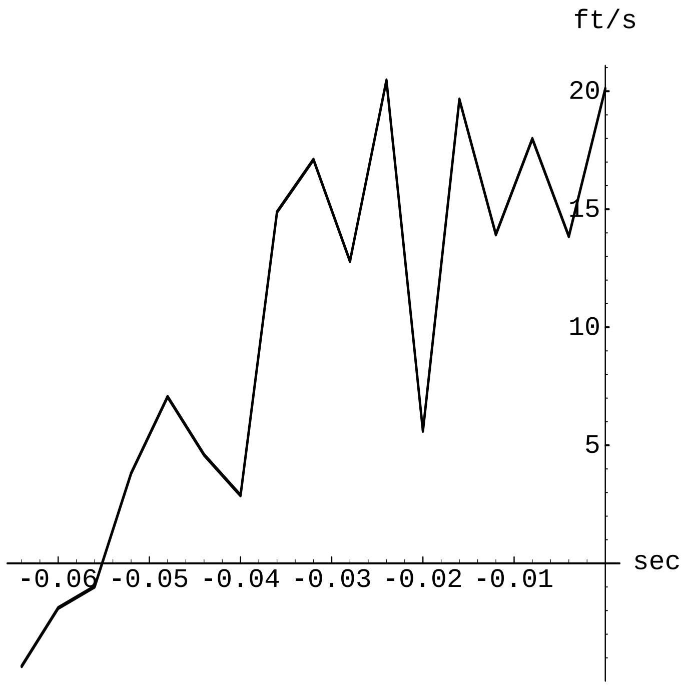
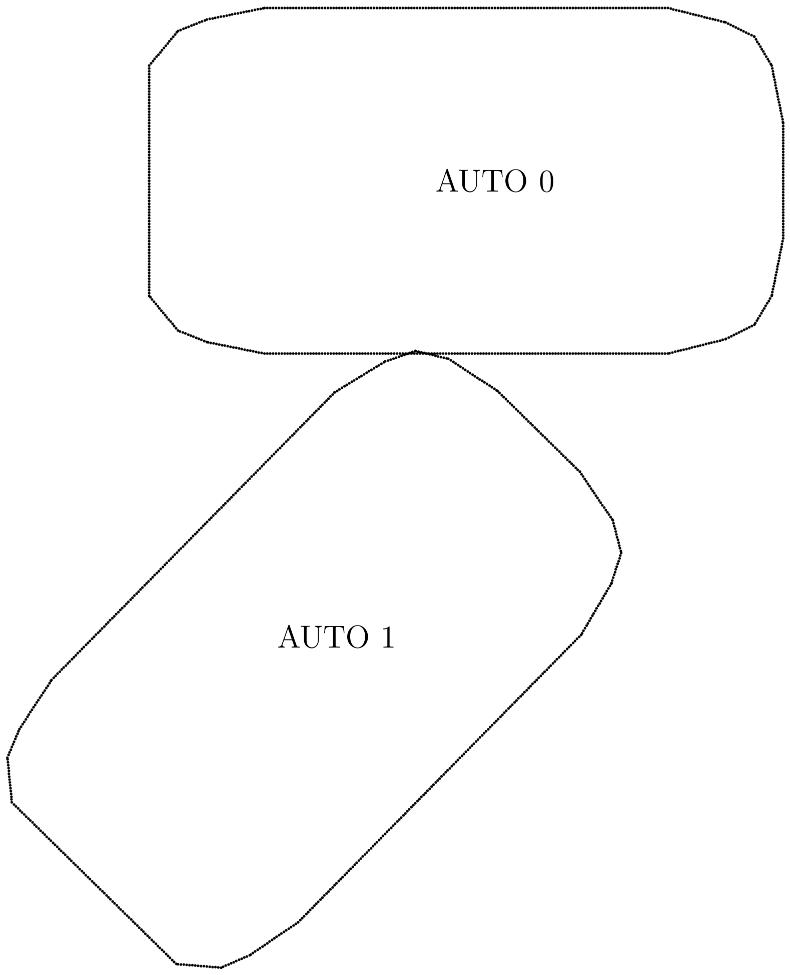


Figure C.4: Case 1 - Error in predicted approach velocity magnitude

C.4 Case 2



target	*relative $\partial x/\partial t$*, fps	*relative $\partial y/\partial t$*, fps
AUTO 1	-1.950	38.689

Sensor Array Performance

- Measured Mean Vx = -1.91358 fps
- Measured Mean Vy = 38.5598 fps

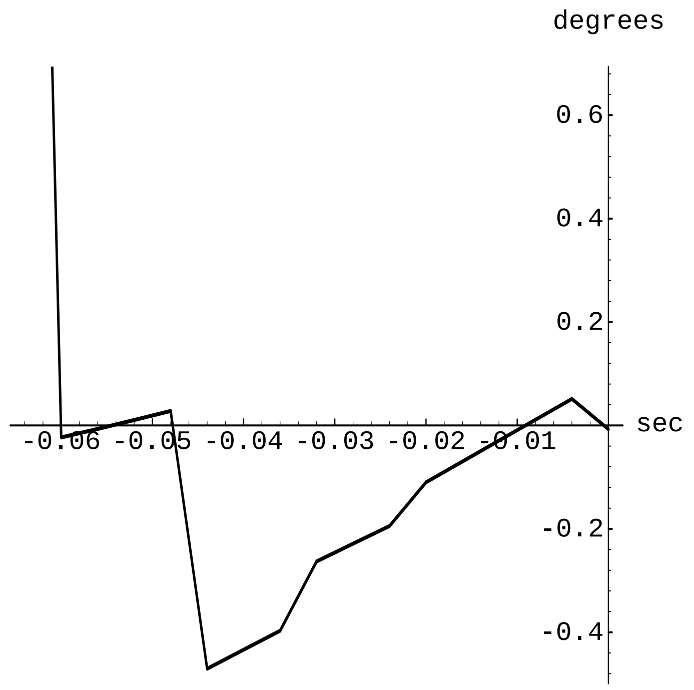


Figure C.5: Case 2 - Error in predicted approach angle

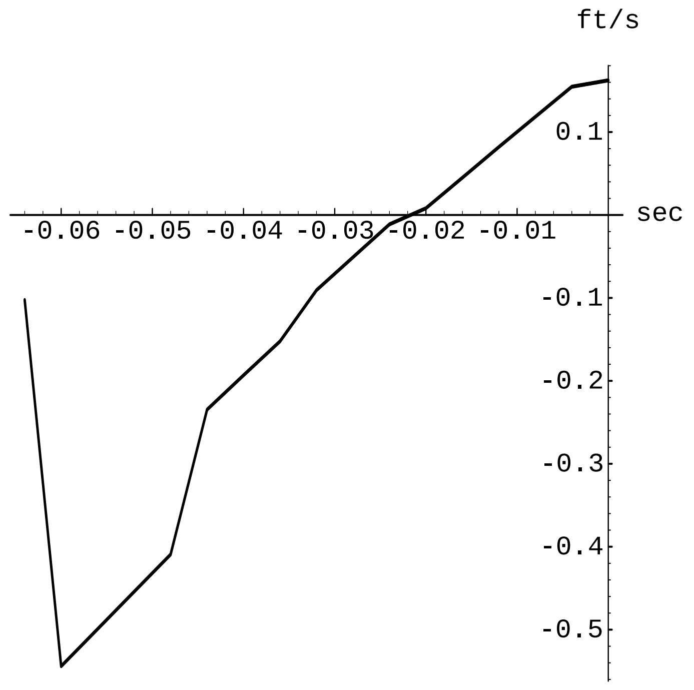
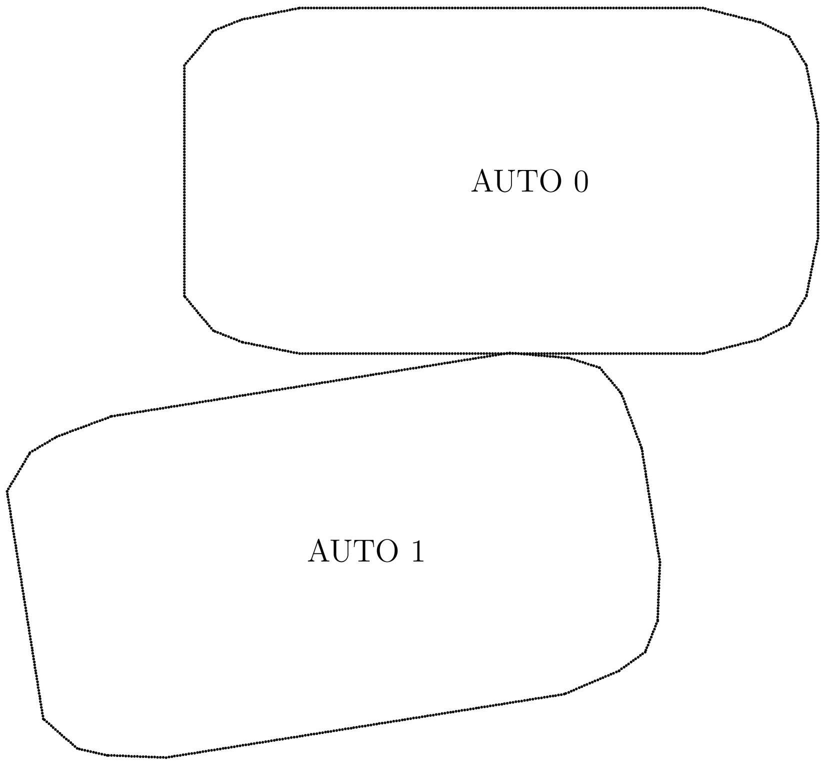


Figure C.6: Case 2 - Error in predicted approach velocity magnitude

C.5 Case 3



target	*relative* $\partial x/\partial t$, fps	*relative* $\partial y/\partial t$, fps
AUTO 1	12.602	8.585

Sensor Array Performance

- Measured Mean Vx = 17.2392 fps
- Measured Mean Vy = 9.8098 fps

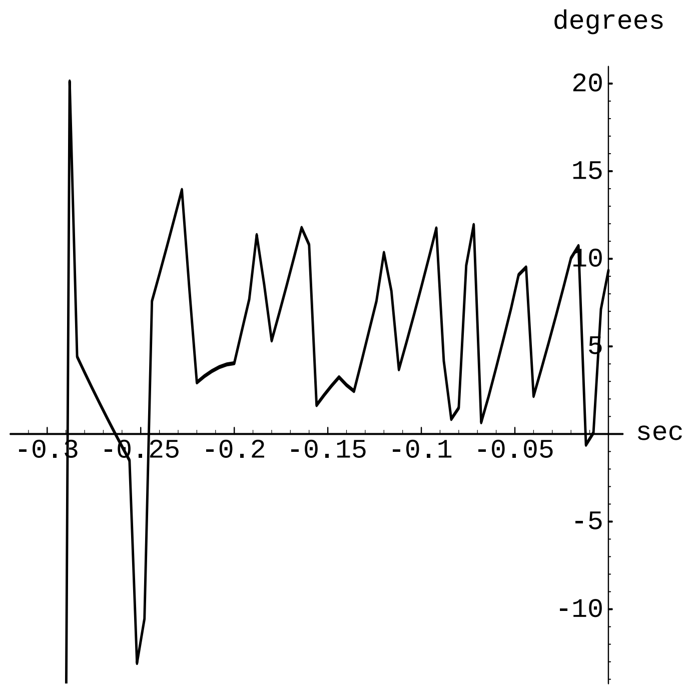


Figure C.7: Case 3 - Error in predicted approach angle

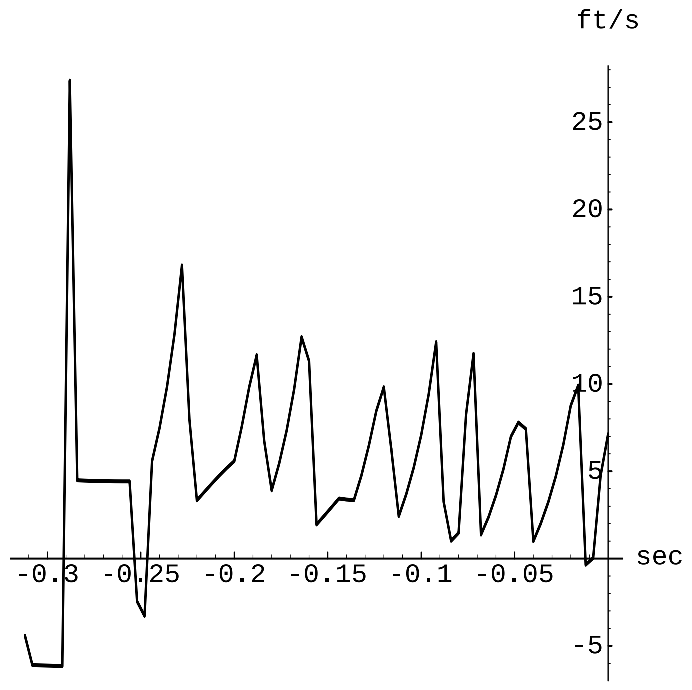


Figure C.8: Case 3 - Error in predicted approach velocity magnitude

Appendix D

Passive Restraint System – parameter optimizations

D.1 force profile – crsh.dat

D.2 force profile – impulse.dat

D.3 force profile – trace1.dat

D.4 force profile – trace2.dat

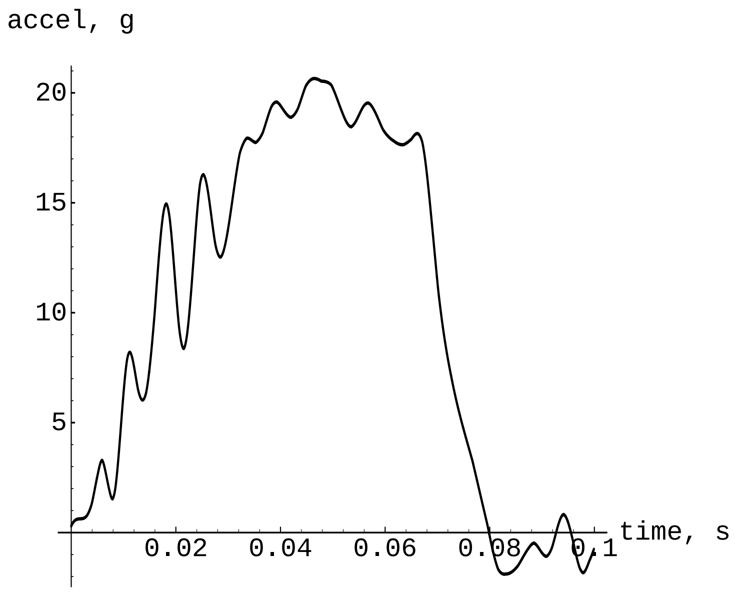


Figure D.1: crsh.dat

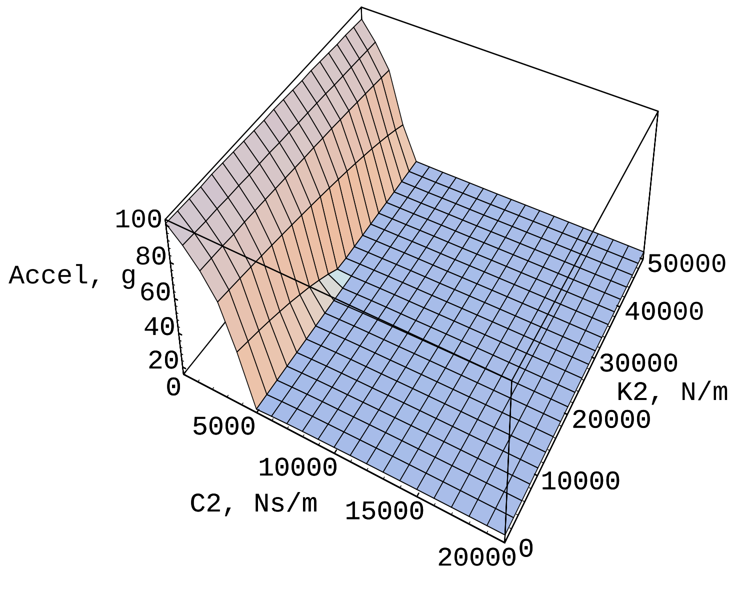


Figure D.2: acceleration versus K and C

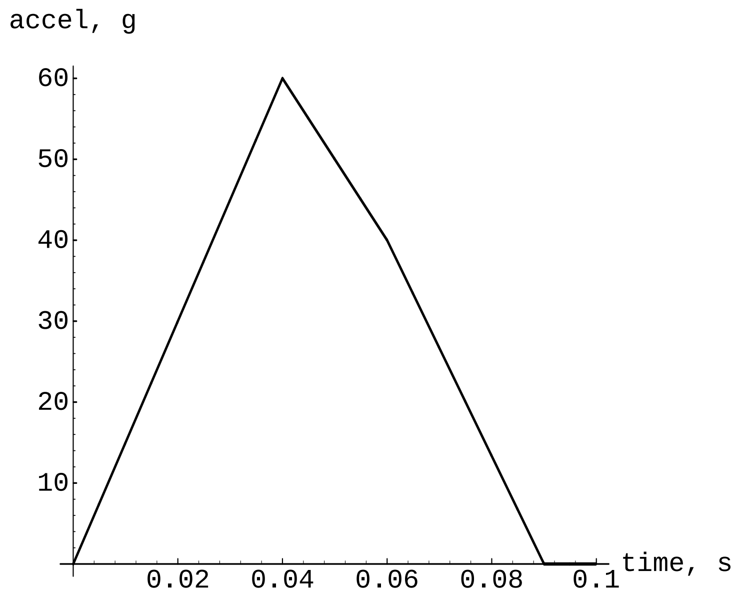


Figure D.3: impulse.dat

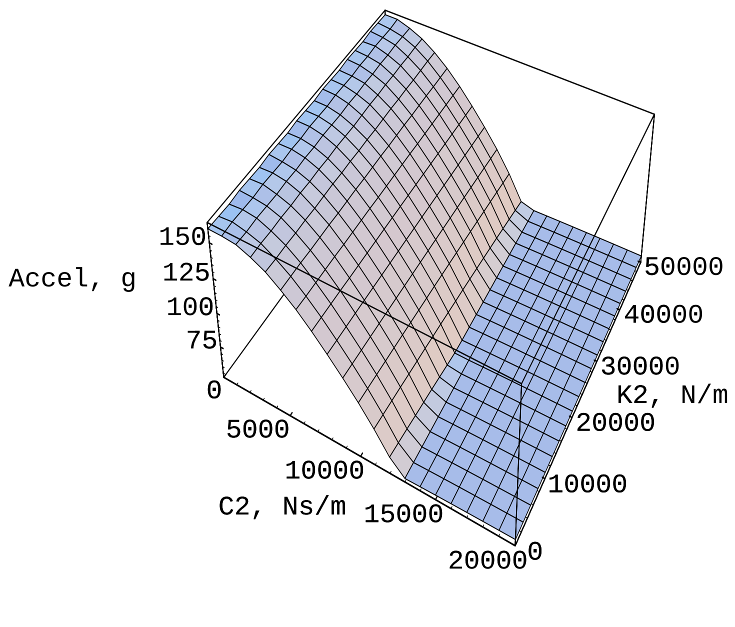


Figure D.4: acceleration versus K and C

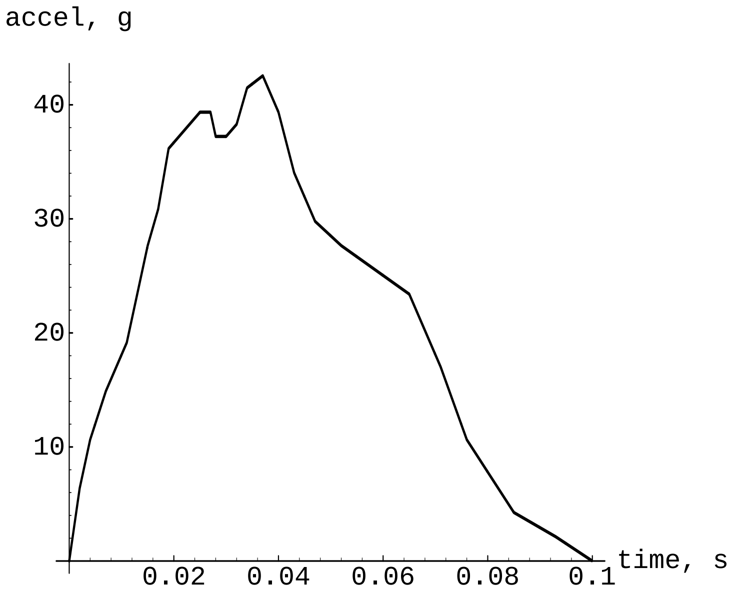


Figure D.5: trace1.dat

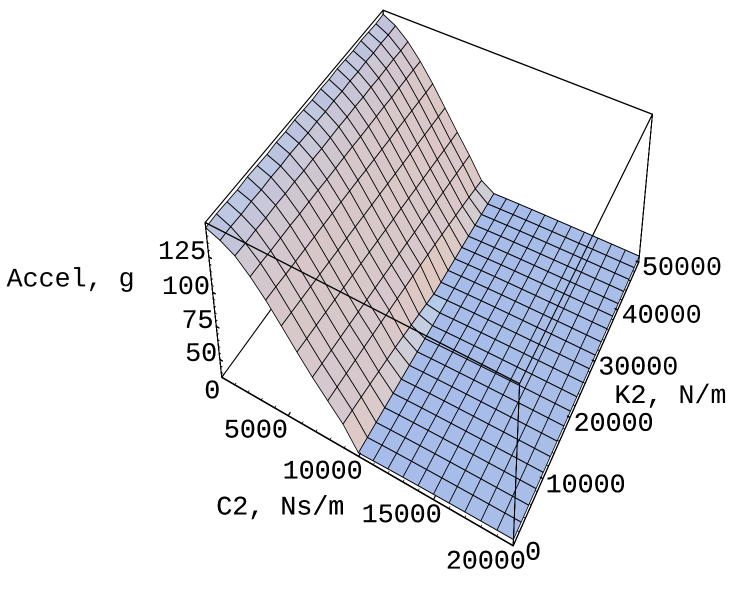


Figure D.6: acceleration versus K and C

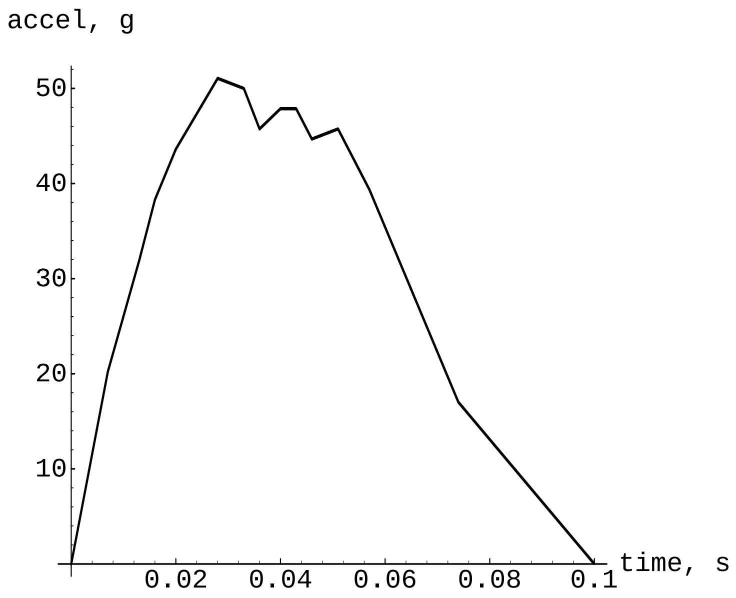


Figure D.7: trace2.dat

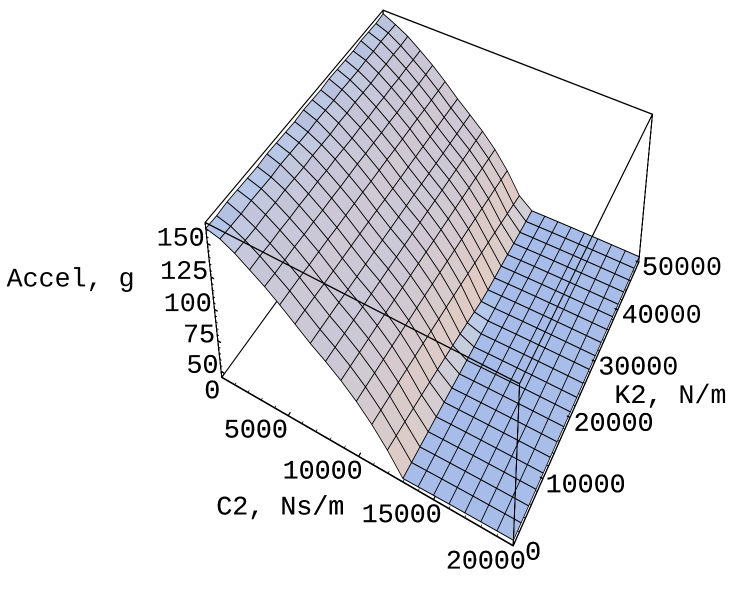


Figure D.8: acceleration versus K and C

Appendix E

Numerical solution of dynamic programming problems

The conversion of an optimal control problem into a dynamic programming problem has been previously covered in this text. Once this conversion has been done, there are the following structures:

state vector

$$X_k = \left\{ \begin{array}{c} x_k \\ v_k \end{array} \right\}$$

system equations

$$G(X_k, f_k) = \left\{ \begin{array}{c} x_{k+1} \\ v_{k+1} \end{array} \right\} = \left\{ \begin{array}{l} \Delta v_k + x_k \\ v_k - \frac{\Delta}{M}(C_1(x_k, v_k) + K_1(x_k) - f_k + MF(k\Delta)/M_{auto}) \end{array} \right\}$$

instantaneous cost

$$A_k(X_k, f, k) = \left| \frac{1}{M}(C_1(x_k, v_k) + K_1(x_k) - f_k \right|$$

recurrence relation

$$U_k(X_k) = \min_f \left[\max\{A_k(X_k, f_k), U_{k+1}(G(X_k, f_k))\} \right]$$

optimal control vector $f_k^* = f_k$ that minimizes U_k

boundary condition

$$J_n = |(C_1(x_k, v_k) + K_1(x_k))/M|$$

Since f_k^* is possibly not unique, f_k^* was arbitrarily chosen to be the smallest acceptable candidate for f_k^* on the interval of f_k tested. Since the system equations are nonlinear, and U_k is an infinity norm, any analytic solutions for f^* and U would are very difficult, if not impossible, to obtain. A numerical solution for f^* and U over some finite domain propagated backwards from step n to step 1 is the way to proceed.

Most references on dynamic programming simply say to represent a finite domain with a lattice of nodal points that define interpolation functions over the solution domain. Then the solution proceeds backwards by solving for U_k and f_k^* only on the lattice points, using an interpolation on the surface U_{k+1}. [2]

However, several important, and rather interesting, issues are left unmentioned (and unresolved) which need to be considered before a successful solution may be obtained. These issues are:

- What are the bounds of the solution domain?
- How can one be sure that the solution of U_k is not in error due to extrapolations on U_{k+1} beyond the defined domain?
- How much error is introduced by the discretization of the solution?

On what domain does one solve the problem?

In this case, the only desired answer was the solution that originated at $k = 1$ at the point $\{x, v\} = \{0, 0\}$ (the occupant's nominal starting position). *A priori*, one does not know what portion of the domain that the solution for $\{x, v\} = \{0, 0\}$ will travel. Indeed, that trajectory is actually the desired solution.

In some cases, there may be some hard physical bounds on the states. Then, one would just limit the solution domain to within those bounds. In this case, however, there are no obvious limits on the solution domain. It may be that the only way to solve the problem is as an iterative process. First, choose boundaries that intuitively seem to contain the solution trajectory. After the problem is solved on the initial domain, test the solution trajectory to see if it goes outside the known domain. If so, widen the solution domain and repeat the process.

How can one be sure that solutions are not in error due to over-reliance on extrapolation?

One does not want a solution that is based on extrapolations of U_{k+1} surfaces. Extrapolation could lead arbitrarily large error. As an aside and a graphic illustration of extrapolation error, consider the following example. The function

$$z = |\sin(x^2 + y^2)^{1/2}|$$

was considered over the region

$$||x, y||_1 \leq 5$$

The function is represented by the same interpolation functions used in the solution of the dynamic programming problem. As one can see in Figure E.1 the representation is very good over the defined domain, but the quality deteriorates rapidly outside of

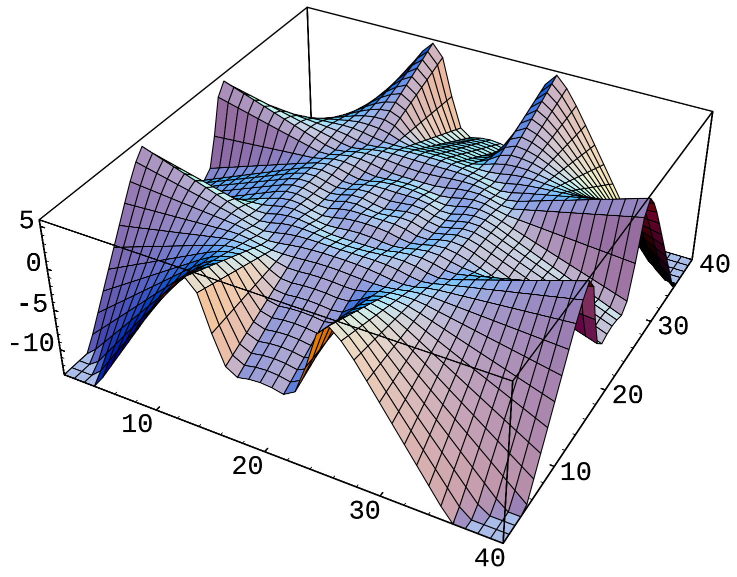


Figure E.1: When good functions go bad.

the domain boundaries.

By careful programming, one might change the definition of the problem domain on every timestep to make sure that every point on the cost surface was formed from an interpolation of the U_{k+1} cost surface. Such an approach would be the rigorous way to program the problem. In this case, however, it was hoped that by choosing a suitably large enough solution domain, only the edges might be polluted by extrapolation. In addition, the cost surface of this problem turned out to be forgiving in that small extrapolations probably would not produce much error – the cost surfaces were all quite smooth in the regions near where the solution lay.

How much error is introduced by a discretized representation of the solution?

In this particular case, this question is especially important. Because of the discontinuous nature of the forces applied by the car body, there was bound to be some

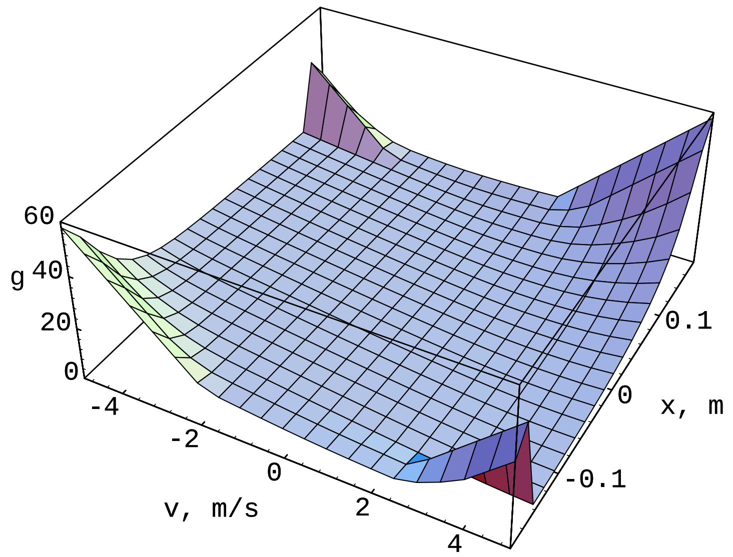


Figure E.2: Cost surface at $k = 1$.

error incurred by the use of continuous interpolation functions. The question is can a believable answer still be obtained, even with error incurred by the discretization scheme. In this case, the answer was yes, even though discretization error did have a noticeable effect.

Figure E.2 represents the predicted cost of each initial position. For the initial position $x, v = 0, 0$, the cost surface predicts a final cost of 8.77 g's. Is this cost to be believed? The solution is to walk through the set of force solution to determine the set of forces that yields this conclusion. Figure E.3 is the displacement history of the solution. Figure E.4 is the acceleration history of the solution. Figure E.5 is the control history of the solution. Looking at Figure E.4, one can see that the 8.77 g acceleration is generated starting at the beginning of the problem. However, a larger acceleration spike is generated later on in the solution when the solution

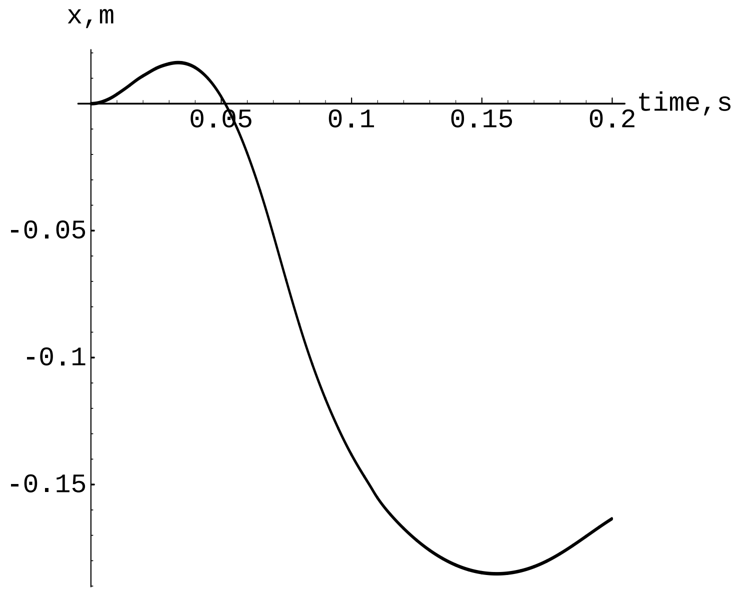


Figure E.3: Displacement history.

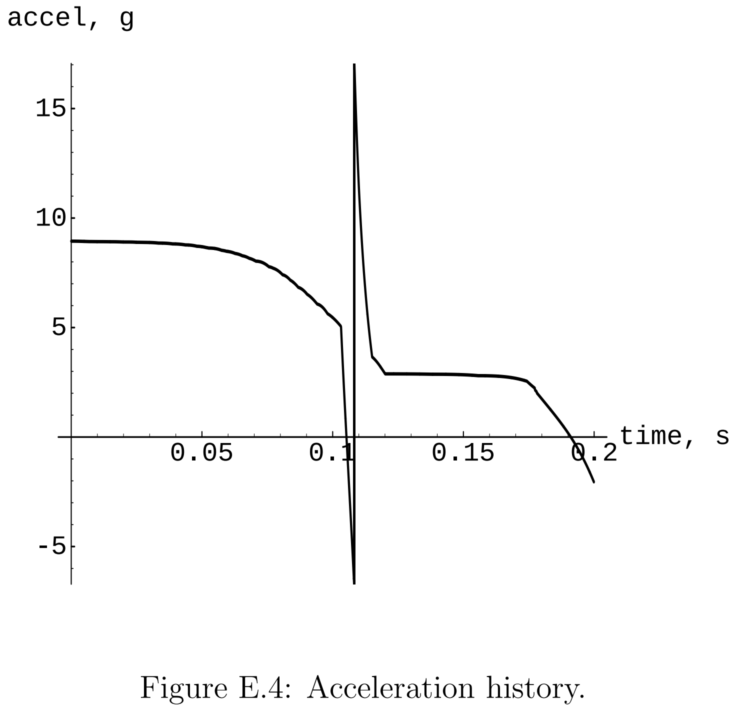


Figure E.4: Acceleration history.

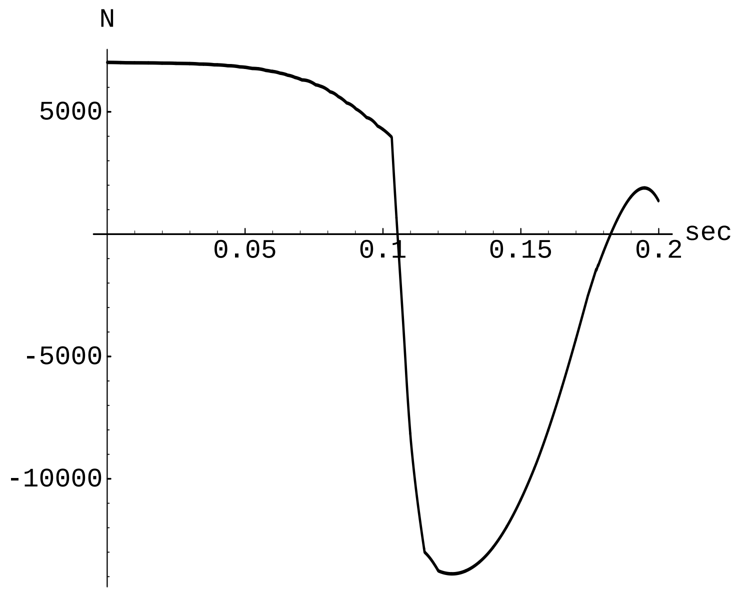


Figure E.5: Force history solution.

crosses over the discontinuity. To avoid a spike at the discontinuity, the force has to be discontinuous as well. From Figure E.5, one can see that although the force changes very rapidly at the discontinuity, it starts the change before the discontinuity is hit at $x = -0.15$ and does not complete the step until well after this point.

Although a discontinuous step is prescribed by the true solution, the discretized solution cannot adequately reflect that step. The surface is approximated by functions of the form

$$z = ax + by + cxy + d$$

which is an interpolation function for a rectangular domain. By the discretization used in this problem, the discontinuity lies directly in the middle of one set of the rectangular elements.

If the discretization were much finer, it is hypothesized that the transition over

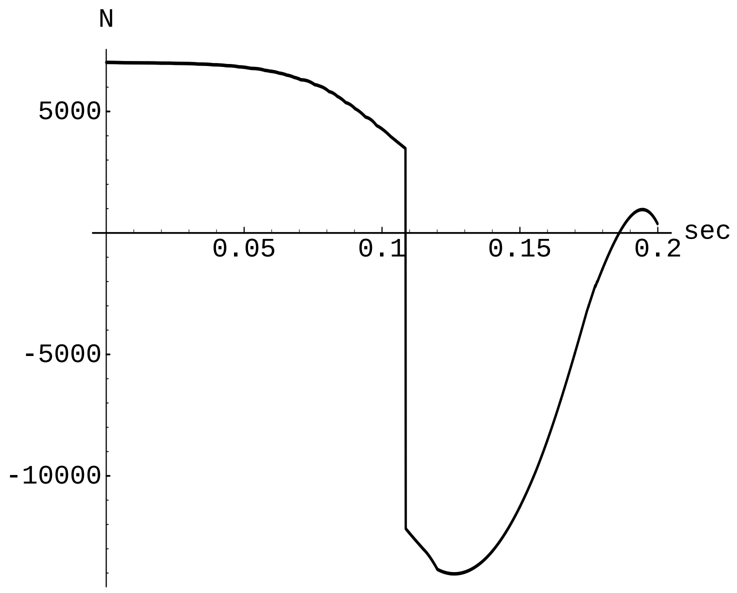


Figure E.6: Corrected force history.

the discontinuity could be taken in such a way that 8.77 max g could be realized by the program. 8.77 max g is a realizable result – by altering the applied forces by hand such that the discontinuity is passed over smoothly, the 8.77 g run is achieved. See figures E, E.7, E.8.

Errors were unquestionably incurred by the above mechanisms. However, the answers produced by the dynamic programming method for the optimal restraint problem are good ones. If not necessarily the absolute optimum, they are definitely close to the best performance obtainable.

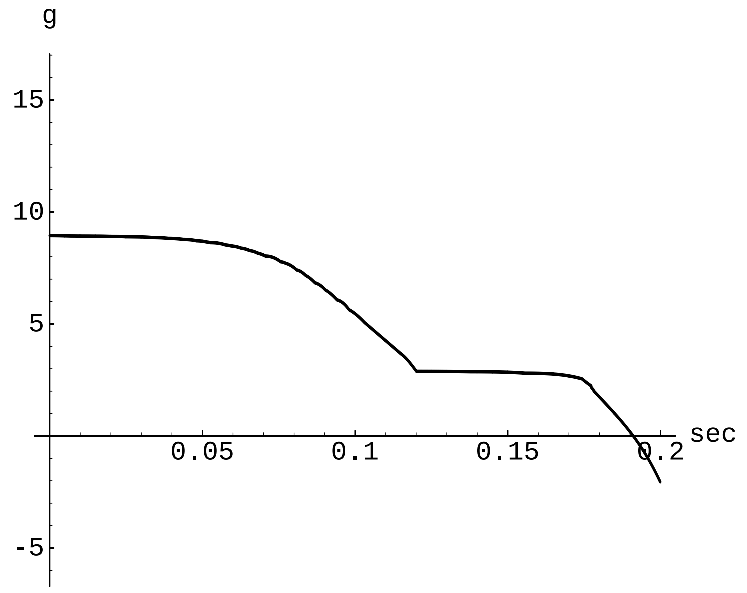


Figure E.7: Occupant acceleration from corrected force.

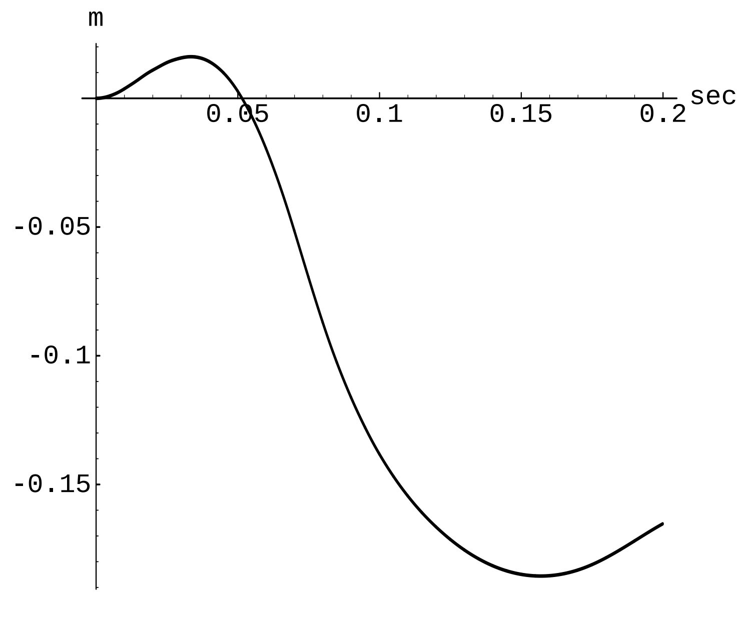


Figure E.8: Occupant displacement from corrected force.