Development of a Performance-based Approach for Collision Avoidance and Mitigation

by

Ji Hyun Yang

B.S., Seoul National University, 2001

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2003

Author_____
Department of Aeronautics and Astronautics
September 2003

Certified by_____
Eric Feron
Associate Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by_____
Edward M. Greitzer
H.N. Slater Professor of Aeronautics and Astronautics Chair, Committee on Graduate Students

Development of a Performance-based Approach for Collision Avoidance and Mitigation

by

Ji Hyun Yang

Submitted to the Department of Aeronautics and Astronautics
on September 2003, in partial fulfillment of the
requirements for the degree of
Master of Science in Aeronautics and Astronautics

Abstract

Many threat assessment algorithms are based on a collection of threshold equations that predict when a collision is to occur. The fact that there are numerous algorithms suggests a need to understand the underlying principles behind the equation design and threshold settings. This thesis presents a methodology to develop appropriate alerting thresholds based on performance metrics. This also allows us to compare different alerting algorithms and evaluate alerting systems.

The method is a performance-based approach in state-space. It can be used as a stand alone system for real-time implementation or a threshold design tool in conjunction with any chosen alerting algorithm or sensor system. Using carefully prescribed trajectory models (which may include uncertainties), the performance tradeoff with and without an alert can be predicted for different states along the course of an encounter situation. This information can then be used to set appropriate threshold values for the desired alerting logic.

The development of the threshold criteria for a rear-end collision warning system is given as an example. Though the approach given is presented as a threshold design tool, the methodology is self-contained as a threat assessment logic. The possibility exists to compute the performance measures on-the-fly from which alerting decisions can be made directly.

We demonstrate the methodology on Lincoln LS concept vehicle with a GPS-based system and a full-cab driving simulator as prototypes. Application examples, a collision mitigation by braking system and a face tracking warning system, are shown to handle the universality of the performance-based approach. For illustrative purposes, a vision-based system (post-processed off-line) is compared with the GPS-based system.

Thesis Supervisor: Eric Feron
Title: Associate Professor of Aeronautics and Astronautics

Acknowledgments

I would like to give special thanks to Professor Eric Feron, who has been a real advisor from the very first time I came to MIT. He is a professor whom I respect. He provided an important source of ideas, motivation and criticism from the beginning of the project.

I must thank to Dr. Lee Yang, who has not only been my project advisor, but has been a great mentor when I was at a loss during my first year in MIT. I appreciate his gentleness and I am impressed by profundity of his knowledge.

I would also express my cordial gratitude to Professor James Kuchar who is a pioneer of the evaluation of hazard alerting systems. His work including SOC curve is a foundation of my research. Professor Trevor Darrell gave me valuable advice about vision-based system. I'm grateful. I would like to thank Louis-Philippe Morency, Animesh Chakravarthy, Dr. Vishwesh Kulkarni and Dr. Kyung Yeol Song for consultation throughout the work. I would also like to thank Ron Miller, Perry Macneille, Jerry Engelman, Jeffrey Rupp, Peter Joh and Rebecca Seiler from Ford Motor Company. They provided me tireless support and great advice.

One of the coolest thing in MIT life is that I have been sharing the lab with bunch of original people. I was impressed by Ioannis Martinos for his serious attitude toward academia. It was great that we could take classes together. Vlad Gavrilets and Jan De Mot, they used to give me really useful advices. David Dugail helped me a lot in my first year at MIT. Rodin Lyasoff, he knows his business. Thank you, everybody. I'm grateful to Lauren Clark and Angela Copyak for there tireless support.

I would also like to thank my friends. To Jhongwoo for preparing the qualifying exam together and being my tennis teacher. To Eun Sun for the encouraging words and kindness. To Wonseon for sharing a passion for ballet. To Jiwon for taking care of me when I was in Michigan for a summer internship. Thank you so much Jaeho, you have been incredibly understanding and always there for me with remarkable patience.

Finally, my special thanks to my dear family and grand parents back in Korea. I cannot be here without all of them. They have been of great support to me in my times of need. My grand parents pray for me everyday and they provided me consistent support. I owe them so much. My parents, they take me under their wings and they make me a woman with self-confidence. I am happy and grateful. My sister and brother, they make me laugh. Thank you, thank you so much.

This work was funded by Ford-MIT Alliance program whose support is gratefully acknowledged.

Contents

List of Figures

List of Tables

Nomenclature

Roman

A	Alert space, Alert trajectory
a	Equation parameters
a	Equation parameters
a_L	Acceleration of the leading vehicle (LV)
a_F	Acceleration of the following vehicle (FV)
B	Buffer
b	Equation parameters
$b1$	Bias of hidden layer
$b2$	Bias of output layer
c	Number of classes
H	Hazard space
I_s	Intensity images associated with key frame s
N	Nominal trajectory
RT	Reaction time
r	Range between two vehicles
$\dot{r}$	Range rate between two vehicles
$\ddot{r}$	Acceleration between two vehicles
SP	Speed penalty
TTC	Time-to-Collision
THW	Time-to-Headway
TTI	Time-to-Impact
v_F	Velocity of the following vehicle (FV)

$\mathbf{x}$	State vector, sensor states
x_s	Pose of key frame s
$\mathbf{x}(t)$	State trajectory
$x_i(t)$	State variables
$w1$	Weights of hidden layer
$w2$	Weights of output layer
y_i	Activation function of the i^th output node
Z_s	Depth images associated with the key frame s

Subscripts

$_F$	Following vehicle
$_i$	Output node
$_L$	Leading vehicle
$_n$	Equation number
$_s$	Key frame

Events

CD	Correct Detection
CR	Correct Rejection
FA	False Alarm
FN	False-Negative
FP	False-Positive
MD	Missed Detection
SA	Successful Alert
TN	True-Negative
TP	True-Positive
UA	Unnecessary Alarm
P(SA)	Probability of Successful Alerts
P(UA)	Probability of Unnecessary Alarm

Acronyms

ACWAS	Automotive Collision Warning and Avoidance System
DOF	Degree of Freedom
FV	Following Vehicle
IVHS	Intelligent Vehicle Highway Systems
LV	Leading Vehicle
PM	Performance Metric
ROC	Receiver Operating Characteristic
SOC	System Operating Characteristic
TCAS	Traffic Alert and Collision Avoidance System

Chapter 1

Introduction

1.1 Objectives

The major objective of this thesis is to develop a performance-based approach for a collision avoidance/mitigation system and to present application examples including a rear-end collision warning system for ground vehicles. An alerting algorithm for a warning system which could be configured for any sensor system is proposed and it details the development of trajectory models. Trajectory models could include uncertainties in state trajectory estimation and some examples of performance metrics which evaluate alerting system performance are introduced to deal with uncertainties.

This algorithm aims at collision mitigation as well as collision avoidance, which could be achieved by introducing different performance metrics to the system, since collision avoidance and mitigation have distinct objectives. Trajectory models for collision avoidance and mitigation will be presented for ground vehicles.

Prototypes and application examples of collision warning system are provided. The alerting algorithm will be combined with several different sensor systems and alerting objectives so that we explore how we could configure different sensor systems and the algorithm together.

1.2 Thesis Overview

Chapter 2 reviews relevant terms and definitions used to describe the alerting system concept. As a way of introducing the concept, the state-space representation of alerting system developed by Kuchar [9, 10] is reviewed. It is based on multivariable control system theory [16]. The performance metric is presented to show the tradeoff between unnecessary alarms and necessary alarms via System Operating Characteristics (SOC) curve and Performance Metric (PM) analysis.

Chapter 3 presents a so-called Performance-based approach, which is the alerting algorithm we developed. This methodology deals with the tradeoff between positive and negative consequences of warnings as well as uncertainties in the model and trajectory estimation. A design process for a rear-end collision avoidance system of ground vehicles is given for an example. Trajectory models, such as a nominal (no alert) trajectory and an avoidance (alert) trajectory are built up for a rear-end collision avoidance system. An analysis based on System Operating Characteristics curve and Performance Metric plot is carried to predict performance of the warning system. Furthermore, the alert space is estimated based on the outcome of the analysis for a selected scenario.

Chapter 4 discusses prototypes application of the methodology. A GPS-based collision warning system is presented and the same warning system is implemented in a full-cab driving simulator. A rear-end collision mitigation by braking system for real-time implementation is studied and the basic trajectory model is developed. To demonstrate the universality of the approach, a drowsiness warning system using a face tracking system is carefully examined.

In chapter 5, the summary and contribution of this research are provided and future work related to this field is discussed.

Chapter 2

Review of Alerting System Concept

As preliminaries, it is worthwhile to begin with a description of the general alerting system concept. Most of contents in this chapter including a description of alerting system is cited from [16] and [9]. This study is a groundwork of developing a new methodology in Chapter 3.

2.1 Generic Alerting System

A hazard warning system is one of several safety components typically found in complex human systems [9, 16]. Its purpose is to monitor potential threats and issue warnings to human operators when undesirable events are predicted to occur [16]. A simplified diagram of a generic alerting system is shown in Figure 2-1.

Information from environment, human and control system is measured by sensors and some parts or all of the information are presented to the human operator by various types of displays, such as cockpits in ground vehicles or airplanes. The information from sensors also could be the input information of an alerting system to help determine the possibility of a hazardous situation [16]. In many cases, a hazard can be detected by the operator from the displays themselves; however, in other instances, the operator may not be fully aware of the situation or may need additional confirmation to aid in decision making [16]. Warning signals from the alerting system

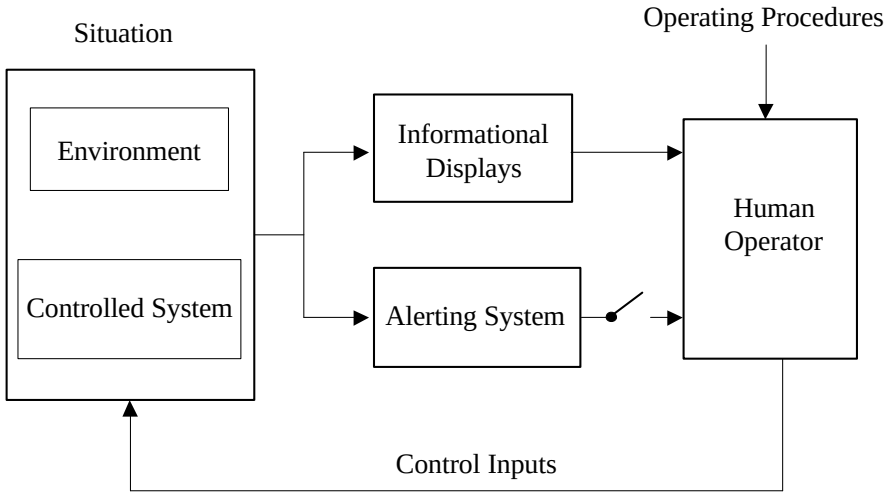

Figure 2-1: Generic alerting system [9, 16]

could be visual alerts, auditory warnings or a combination of both of them. Still, the human is ultimately responsible for making a final decision in most cases, such as activating a brake system or not when an alert is given to the driver. The degree of automation can vary, with some alerting systems providing a simple warning, while others give additional resolution advisories [16]. Three categories of applications that typically include alerting systems are outlined in Figure 2-2.

Alerting systems in process control applications are generally designed to alert the operator that certain parameters of the process are reaching dangerous values. Applications in transportation systems typically involve a controller that oversees a number of independent vehicles [9]. Alerting systems in single vehicles are generally designed to warn the vehicle operator that a hazard exists. In response to an alert, the operator typically changes the trajectory or configuration of the vehicle [9].

2.2 State-Space Representation of Alerting System

The use of state-space representation [9, 10, 16] is a good way of introducing the concepts and issues associated with alerting system design. The following is a brief description of state-space methodology and is applicable to any alerting system.

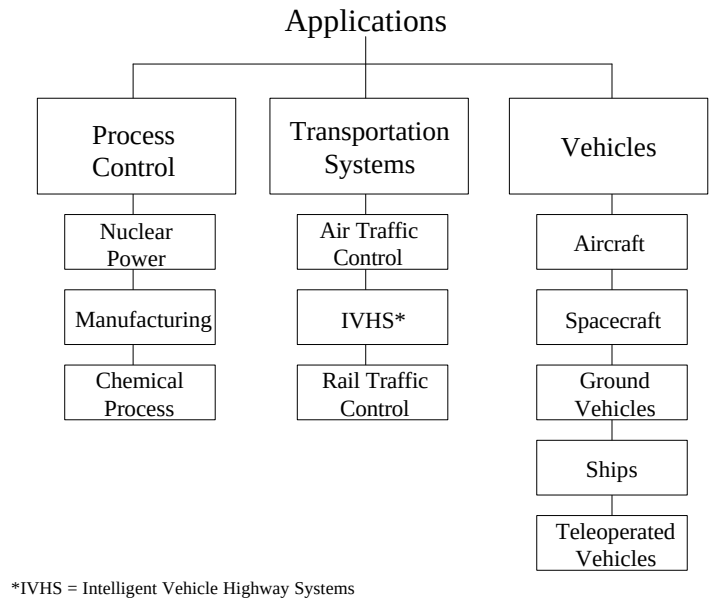

Figure 2-2: Typical alerting system application [9]

The state-space representation is based on multivariable control system theory such as described in [7]. The variables $y_1(t),\ y_2(t),\ \ldots, y_n(t)$ are defined as the values of the states of the system at time t. These states represent the complete set of parameters that define the dynamics of a hazardous situation. In a collision alerting system application, for example, these states include the relative positions and velocities of the own vehicle and an obstacle [9].

The processing components of the alerting system may not monitor some of the states $y_1(t),\ y_2(t),\ \ldots, y_m(t)$. As in control system design, unobservable states may have a large impact on the performance of the system [9]. (There is a component in the alerting system, which is the information sources that are used during normal operation of the system. Cockpit instruments, air traffic control communications, view through the windscreen, aeronautical charts [9] and speedometer of ground vehicles are examples of nominal information sources.) Note that the states from the information sources $x_1(t),\ x_2(t),\ \ldots, x_n(t)$ do not necessarily represent all states in $y_1(t),\ y_2(t),\ \ldots, y_m(t)$; some states may be unobservable to the rest of the control loop [9]. Observable states may be direct measurements of states in $y_1(t),\ y_2(t),\ \ldots, y_m(t)$

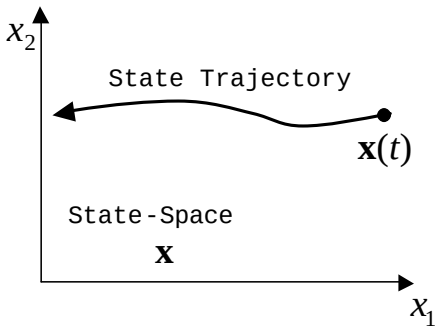

Figure 2-3: Example of state-space diagram [16, 9]

such as reading speed of airplane or car, altitude from a cockpit instrument or may be indirect approximations of a state in $y_1(t),\ y_2(t),\ \ldots, y_m(t)$. From now on, we describe the generalized model of alerting systems with states observable from the processing components of the alerting system.

The *state-space* of an alerting system is comprised of the states $\mathbf{x} = [x_1\ x_2\ \ldots x_n]$ available to the alerting logic to characterize a threat condition. Typically these states are sensor inputs or other pertinent information that can be used to describe the operating environment. An example could be $\mathbf{x} = [r\ \dot{r}]$ where r is the range between two vehicles and $\dot{r}$ is the corresponding range rate. At any given time t, the current state of the system, as known to the alerting logic, is identified by the *state vector* $\mathbf{x}(t) = [x_1(t)\ x_2(t)\ \ldots x_n(t)]$. The set of $\mathbf{x}(t)$ over a given time interval is termed the *state trajectory* as depicted in Figure 2-3.

In certain regions of the state-space, there are domains where undesirable events can occur, such as a collision (e.g. $x_1 = r = 0$). These regions are termed *hazard space*. Whenever $\mathbf{x}(t)$ is allowed to enter a region of hazard space, a *missed detection* has occurred and the alerting system has failed to provide the necessary protection to prevent an unwanted event [16]. An example of hazard space as depicted in a state-space diagram is show in Figure 2-4.

The *alert space* is defined as the set of all state vectors $\mathbf{x}(t)$ in which the system will initiate an action or alert in order to prevent a possible intrusion into hazard space. By definition, no alerts are generated when $\mathbf{x}(t)$ is outside of alert space. The

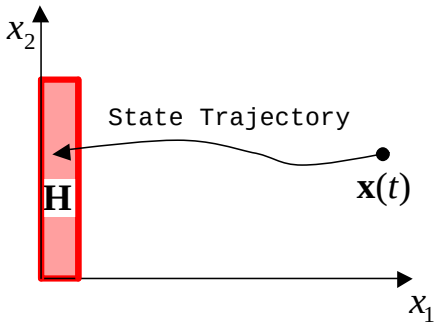

Figure 2-4: Example of hazard space in state-space diagram

boundaries of alert space are considered the alerting *thresholds* and basically define when alerts are given and when they are not. These critical points will be denoted $\mathbf{x}^*$. In Figure 2-5, an example alert space is shown in a state-space diagram. When the state trajectory first enters alert space, the thresholds are met and an alert is issued. At this point, the alerting logic has decided an intrusion into hazard space is likely if nothing is done. By initiating an alert, it is expected that some action will be performed to alter the course of the state trajectory in order to prevent a hazard from taking place.

The main difficulty with alerting is due to the uncertainties in the problem. There are errors in the current states, plus the path of the state trajectory is usually not known exactly. In Figure 2-6, we can see different trajectories based on the information of current states $\mathbf{x}(t)$. When the alerting system gives a vehicle operator a warning signal to notify him or her of a hazardous situation, the operator could make a proper action to avoid the hazard situation such as a collision. Or, the operator may respond too late to avoid the hazard situation.

We can think same result when no alarm is given to the human operator. For example, most drivers can determine if there is a hazardous situation or not and can make an action to avoid the collision without any warning. Still, there are possibilities that drivers are not aware of the situation so that the collision or other hazard situation could occur.

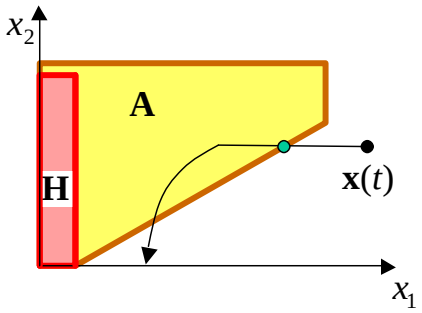

Figure 2-5: Example of alert space in state-space diagram

2.3 Alerting Outcomes and Performance Metrics

Ideally, an alert correctly notifies the driver when a hazard will occur if nothing is done about the current situation of the system [16]. To evaluate characteristics of the outcomes we could classify alerts as: correct detection(CD), missed detection(MD), false alarms(FA) or correct rejection(CR), etc. The definition of each category could vary with system designers. An example of the complete decision outcomes is depicted in Figure 2-8.

A similar concept is depicted in [9] with different terms. If the situation is truly hazardous, an alert may be necessary and is termed a true-positive, denoted **TP**. Similarly, a failure to alert when the situation is non-hazardous is a true- negative, denoted **TN**. An alert that is issued in a non-hazardous situation is a false-positive, denoted **FP**, and likewise, failure to alert to a truly hazardous situation is a false-negative, denoted **FN**. This is summarized in Table 2.1. The meaning of **T**, **F**, **N** and **P** is as follows: 'True' because the correct alerting decision is made, and 'positive' because an alert is issued. 'False' indicated that the alerting decision is wrong, and 'negative' indicates that an alert is not issued [9].

We introduce probabilistic performance metrics to quantify the performance outcomes of the alerting system. There could be several different types of performance metrics and, initially, we focus on describing two performance metrics we used for the collision warning/avoidance system in chapter 3. To quantify alerting system outcomes from both positive and negative points, we would like to weigh the outcome

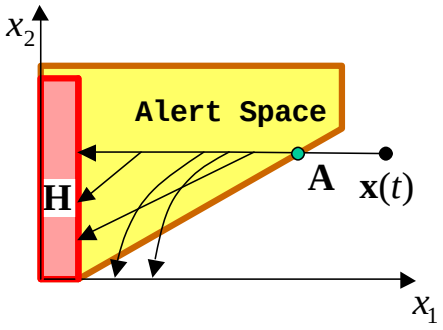

Figure 2-6: Different trajectories due to uncertainties : P(SA)

between initiating an alert or not at various $\mathbf{x}$ locations spanning the state-space region of operating interest. To do so, two trajectory models are needed at each state position - one if the alert is issued (referred to as the *Avoidance* or *Alert Trajectory*, $\mathbf{A}$) and one if no alert is issued (called the *Nominal* or *No-Alert Trajectory*, $\mathbf{N}$). These models are simply the vehicle dynamics of the encounter along with any uncertainties involved in predicting current and future states. Figure 2-6 and Figure 2-7 depicts a representative example of these two trajectories.

Because of the uncertainties, a statistical description of alerting performance is needed. We use the quantitative metrics *Probability of Successful Alert*, $P(SA)$, and *Probability of Unnecessary Alert*, $P(UA)$. Previous papers [16, 9] have used the term *Probability of False Alarm*, $P(FA)$ instead of Unnecessary Alert, but the author feels the wording of $P(UA)$ is more appropriate. Thus we define the following:

$$P(SA(\mathbf{x})) = 1 - P(C|\mathbf{A}(\mathbf{x})) \tag{2.1}$$

$$P(UA(\mathbf{x})) = 1 - P(C|\mathbf{N}(\mathbf{x})) \tag{2.2}$$

where $P(C|\mathbf{A}(\mathbf{x}))$ is the probability of a collision if an alert is given at $\mathbf{x}$ and $P(C|\mathbf{N}(\mathbf{x})))$ is the probability of a collision if no alert is given. Thus at each state $\mathbf{x}$ location, there is an associated positive consequence of initiating an alert (measured by $P(SA)$) and an subsequent negative consequence (measured by $P(UA)$).

Unless the state size is 2 or less, however, it is generally too difficult to visualize $P(SA)$ and $P(UA)$ in state-space. Instead, a sort of performance space can be utilized

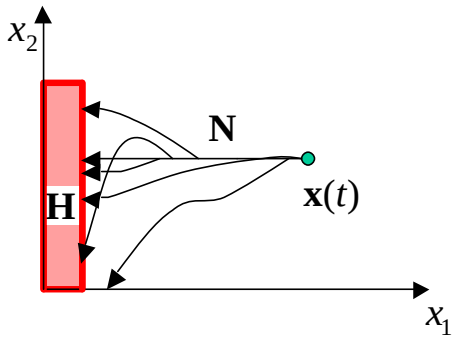

Figure 2-7: Different trajectories due to uncertainties : P(UA)

by plotting $P(SA)$ versus $P(UA)$ over the course of a specific encounter situation. This type of plot was referred to as a *System Operating Characteristic* (SOC) curve by Kuchar [10, 9], similar to ROC (Receiver Operating Characteristic) curve in Signal Detection Theory.

2.4 Performance Metric Analysis

A SOC plot (Figure 2-9a) depicts the tradeoff between the positive and negative consequences of alerting at different states during course of an encounter. Setting a threshold to alert early (and often) can prevent accidents from occurring (high $P(SA)$), but this comes at the expense of increased nuisance or unnecessary alarms(high $P(UA)$). However, by alerting late to reduce $P(UA)$, we lower the chances of avoiding a collision if there is indeed a threat.

Ideally one would like the operating point to be at the upper left corner of the SOC plot, where all alerts successfully avoid an accident and there is no nuisance alarms. However, the uncertainties involved in the prediction of future states usually prevent this from occurring. Consequently, the uncertainties modeled in the trajectories **A** and **N** actually define the achievable performance of the system (bend in the curve). If the future trajectories were to be deterministic (without any errors), then the operating point must lie at one of the 4 corner points of the SOC diagram. Generally, this is not the case and an ideal system is never achieved regardless of the setting.

Sometimes a more convenient way of presenting the performance tradeoff is by

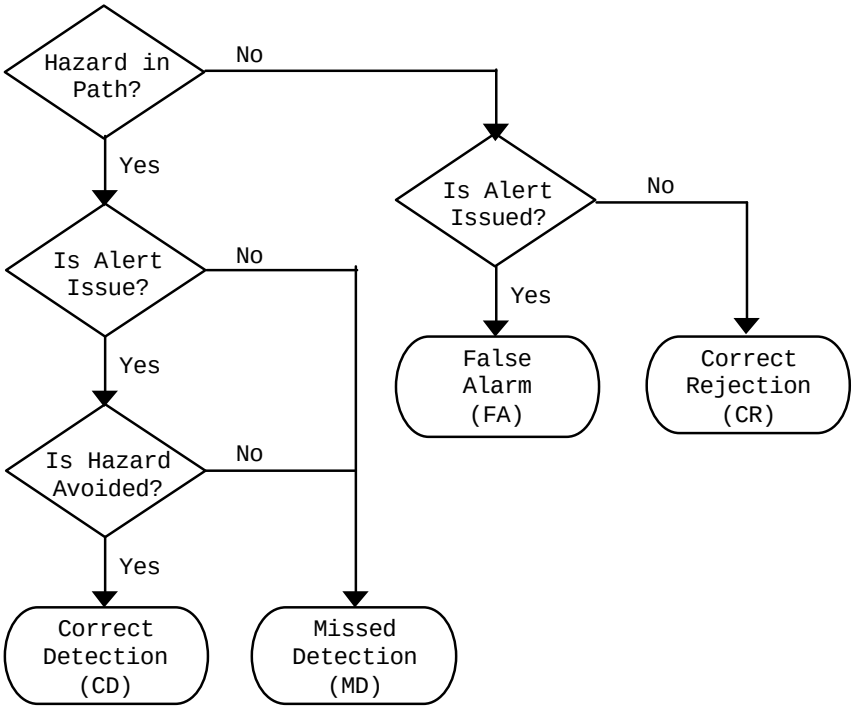

Figure 2-8: Alert decision outcomes [16]

plotting $P(SA)$ and $P(UA)$ separately as in Figure 2-9b. Here the gap between $P(SA)$ and $P(UA)$ is a measure of the system benefit of an alert at any given $\mathbf{x}$. This type of plot will be referred to as a *Performance Metric* (PM) plot [19].

2.5 Summary

Description of generic alerting systems is presented in the section 2.1 and the use of state-space representation was shown in the section 2.2 as a way of presenting the concepts representing alerting system design. We introduced a probabilistic performance metric to deal with uncertainties in state trajectory. As a method of analyzing the performance tradeoff of alerting systems, SOC curve and PM plot are highlighted and provide a framework for developing a performance-based approach in later chapters.

	True situation	
	Non-Hazardous (Incident does not occur along T)	Hazardous (Incident occurs along T)
Alert Not Issued	**TN:True-Negative**	**FN:False-Negative**
Alert Issued	**FP:False-Positive**	**TP:True-Positive**

Table 2.1: Alerting decision outcomes [9]

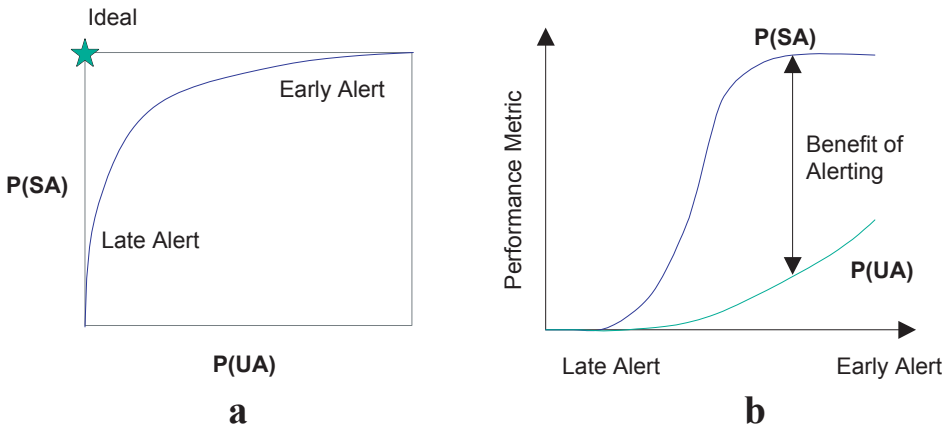

Figure 2-9: Conceptual diagram of SOC Curve and PM plot

Chapter 3

Performance-based Methodology for Alerting System Design

3.1 Introduction

Typically, alerting systems are developed through an ad hoc, evolutionary process, although many design issues are common across applications [9]. The purpose of this research is to provide a deeper understanding of alerting logic design and to provide a methodology to model and evaluate alerting system. It is not intended to be an in-depth evaluation of any specific algorithm, but instead provide insight to the design of the alerting equations.

We begin by presenting a conceptual model of the threshold design process. We discuss the development of performance metrics and an approach to set alert thresholds to achieve the desired level of performance based on a state-space view of alerting that facilitates a clear and generalizable description of the important elements involved in an alerting system.

3.2 Conventional Method of Threshold Placement

An alerting system is a discrete decision-making element that continually decides whether to remain silent or to warn of an impending hazard. Typically, the logic is based on a set of equations that determine the threshold for which the alert is to be

issued. For example, some of the more commonly used alerting equations in ACWAS (Automotive Collision Warning and Avoidance System) development extracted from [6, 1, 20, 14, 15, 8, 2, 5] are listed in Table 3.1. These are the six typical alerting equations for rear-end collisions.

No.	**Threshold Equations : Alert if the condition is met**
1	$r < \frac{(\dot{r} + v_F)^2}{2a_L} - \frac{v_F^2}{2a_F} + RT \cdot v_f + B$
2	$r < -\frac{\dot{r}^2}{2a_F} + RT \cdot v_f + B$
3	$\frac{r}{\dot{r}} < TTC$
4	$\frac{r - SP \cdot v_F}{\dot{r}} < TTC$
5	$\frac{r}{v_F} < THW$
6	$r < -TTI \cdot \dot{r} - \frac{1}{2}\ddot{r} \cdot TTI^2$

Table 3.1: Alerting threshold equations

r = Range between two vehicles $[m]$
$\dot{r}$ = Range rate between two vehicles $[m/s]$
$\ddot{r}$ = Acceleration between two vehicles $[m/s^2]$
v_F = Velocity of the following vehicle (FV) $[m/s]$
a_L = Acceleration of the leading vehicle (LV) $[m/s^2]$
a_F = Acceleration of FV $[m/s^2]$
RT = Reaction time $[sec]$
B = Buffer $[m]$
SP = Speed Penalty $[sec/(km/h)]$
TTC = Time-to-Collision $[sec]$
THW = Time-to-Headway $[sec]$
TTI = Time-to-Impact $[sec]$

Most alerting algorithms are based on the use of threshold equations : if conditions are met then the alert is given. Any threshold equation is an approxiamaion of the alert states in the system; some of them work well for certain situations and some of them do not. The first equation in Table 3.1, for example, will work well in the situation when the leading vehicle is decelerating since the alerting equation is derived from the assumption that the leading vehicle is braking. However, the equation may not work well and may not estimate alert states properly for other situations, such as when two vehicles are travelling with same speed or the leading vehicle is not moving. (This will be verified in later chapters as a result of performance-based approach.)

The first two equations in Table 3.1 are often referred to as distance-based threshold algorithms while the latter four equations are known as time-based or perception-based algorithms. Usually, people try to tune parameters(RT, TTC, TTI, THW, etc.) in the threshold equations in order to force the threshold equations to give the best performance in alerting systems. The way people usually tune design parameters is through a lengthy iterative trial and error process.

Typically, we choose an alerting equation and start with an initial guess for the equation parameters, such as $TTC = 5sec$ or $RT = 2sec$ and run it through simulations with test scenarios. Based on the performance outcome, the parameters are modified in an iterative fashion until the desired performance is met. By repeating this procedure, we could define states of the alerting boundary, $\mathbf{x}^*$, for example warning distance and warning range rate.

3.3 Performance-based Threshold Placement

In this section, we develop a performance-based design method, which is a systematic approach to design threshold placement. If it is possible to predict the state trajectory without using any alerting equation and to decide whether we are in a safe situation or in an alert space $\mathbf{A}$. Then we can build an alerting algorithm independent from any of those equations.

The approach given here is a direct application of the concepts described in Chapter

2 and [9, 16]. The process begins by developing the alert trajectory (**A**) and nominal trajectory (**N**) models for a specific encounter scenario. Uncertainties in these state trajectories should be a part of the model and may include some characteristics of human behaviors, vehicle characteristics, sensor errors, environmental factors, etc.

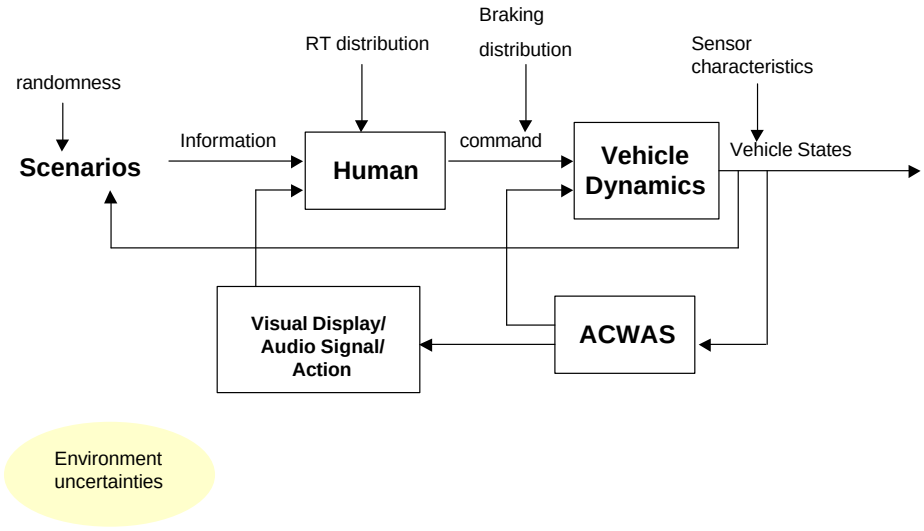

Figure 3-1: Schematic diagram of trajectory model

Given trajectory models, performance metrics, for example $P(SA)$ and $P(UA)$, can be obtained according to time over the course of a hazardous encounter situation. For our work, we use Monte-Carlo simulations to compute performance metrics for each scenario. A desired operating point corresponding to a specific alert state $\mathbf{x}^*$ is picked from the analysis using a SOC curve or PM plot. The performance-based approach is depicted in Figure 3-2.

Regardless of the choice of equations, the instance the alert is first given corresponds to a certain state of the system. Any alert state in the alert space **A** is a set of state variables $x_1(t),\ x_2(t),\ \ldots,x_n(t)$ describing current system statues. Thus, alert state boundary $\mathbf{x}^*$ could be understood as a combination of state variables, which describes the threshold between alert space **A** and safe space.

In general, the state-space threshold function may be expressed in the form $f(\mathbf{x}, \mathbf{a})$ where $\mathbf{x}$ is the system states and $\mathbf{a}$ is the set of equation parameters. We could say the state-space threshold is a set of $\mathbf{x}$ which forms the alert state boundary. The function

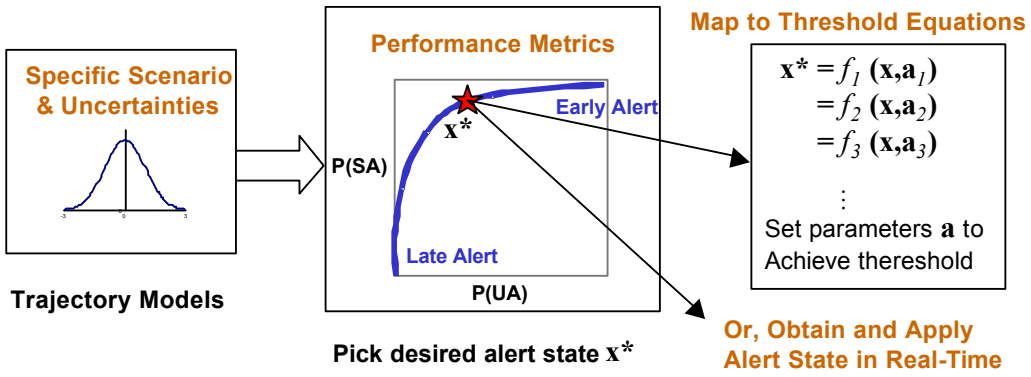

Figure 3-2: Performance-based approach

f is usually unknown, but may be approximated by any of the alerting equations. The condition to alert can then be written as

$$f(\mathbf{x}, \mathbf{a}) > 0 \tag{3.1}$$

The boundary between alerting and not alerting is then defined by the set of states $\mathbf{x}^*$ such that

$$f(\mathbf{x}^*, \mathbf{a}) = 0 \tag{3.2}$$

Though not normally written in this way, the equations listed in Table 3.1 can be reformulated in the functional form of Equation 3.1 to emphasize their structural differences. Even though these equations appear to be completely different in modality, the instance in which they alert is associated with some state combination $\mathbf{x}^*$. In fact, TTC, THW, TTI are really just equation parameters, much like the slope and y-intercept for the equation of a line. These parameters, as well as a_L, a_F, and RT are predefined and are not actual sensor readings. They can be and are often adjusted to match some desired result or performance criteria - in essence a curve-fitting activity. Using simple algebra, the rearranged form of equations is given in Table 3.2

In Function 1 for example, the current system states $\mathbf{x}$ are $[r\ \dot{r}\ v_F]$, the set of

No.	Threshold Function $f(\mathbf{x}, \mathbf{a})$
1	$f_1(\mathbf{x}, \mathbf{a}) = -r + \frac{(\dot{r} + v_F)^2}{2a_L} - \frac{v_F^2}{2a_F} + RT \cdot v_f + B$
2	$f_2(\mathbf{x}, \mathbf{a}) = -r - \frac{\dot{r}^2}{2a_F} + RT \cdot v_f + B$
3	$f_3(\mathbf{x}, \mathbf{a}) = -r - TTC \cdot \dot{r}$
4	$f_4(\mathbf{x}, \mathbf{a}) = -r - TTC \cdot \dot{r} + SP \cdot v_F$
5	$f_5(\mathbf{x}, \mathbf{a}) = -r + THW \cdot v_F$
6	$f_6(\mathbf{x}, \mathbf{a}) = -r - TTI \cdot \dot{r} - \frac{1}{2}\ddot{r} \cdot TTI^2$

Table 3.2: Alerting threshold functions

equation parameters $\mathbf{a}$ is $[a_L \ a_F \ RT \ B]$ and the alert boundary is the set of states

$$\mathbf{x}^* = [r^* \ \dot{r}^* \ v_F^*] \tag{3.3}$$

satisfying

$$f_1(\mathbf{x}^*, \mathbf{a}) = 0 \tag{3.4}$$

Thus, $\mathbf{x}^*$ can then be mapped to the alert function $f(\mathbf{x}, \mathbf{a})$ by solving for the parameter $\mathbf{a}$ in

$$f_1(\mathbf{x}^*, \mathbf{a}) = 0 \tag{3.5}$$

The result is essentially a curve fitting exercise in n-dimensional state-space.

Often the choice of parameters settings is accomplished through an iterative, trial-and-error process. First an alerting function $f(\mathbf{x}, \mathbf{a})$ is chosen with an initial set of parameters $\mathbf{a}$ defining the threshold boundary. These thresholds are evaluated over

one or many test scenarios (in simulation or in the field), then the equations and parameters are modified iteratively based on the performance outcome of the results. The final settings of $\mathbf{a}$ will in turn define the set of states $\mathbf{x}^*$ forming the alert state boundary.

For different encounter scenarios, it is usually necessary to compute different parameters or to use different functions in order to achieve a desired performance. The idea is similar to the modelling of a very complicated function with piecewise linear curve fits. The alerting logic would then implement a different $f(\mathbf{x}, \mathbf{a})$ depending on the expected scenario (for example, intersection environment versus freeway driving, or cut-in versus passing). This could be accomplished with *if-then* statements or *neural network classification* within the framework of a larger hierarchical decision logic. (Global design vs. situation-specific design is discussed in the next section.) Or, the parameters could be implemented as a function of the states themselves, i.e. $\mathbf{a} = g(\mathbf{x}, \mathbf{b})$. However, this has the same effect as just changing the threshold function,

$$f(\mathbf{x}, \mathbf{a}) = f(\mathbf{x}, g(\mathbf{x}, \mathbf{b})) = h(\mathbf{x}, \mathbf{b}) \tag{3.6}$$

The method explained thus far is largely an off-line exercise for setting threshold parameters. However, because information on the trajectory models can change constantly over time, it may become overly burdensome to derive a different $f(\mathbf{x}^*, \mathbf{a})$ for all encounter situations expected in the field. Instead, the possibility exists to compute the performance metrics on-line and make alert decisions based directly on $P(SA)$ and $P(UA)$ (e.g. alert if $P(SA) < 0.95$ and $P(UA) < 0.4$). Given any time, real-time Monte-Carlo simulation gives a value of performance metrics instead of the whole SOC curve or PM plot. An example of this on-line approach using real-time Monte-Carlo simulations for aircraft collision avoidance is given in [17, 16].

3.4 Global Design vs. Situation-Specific Design

A global design refers to a process in which the simulation used to set the thresholds is based on an aggregate mixture of different encounter scenarios; while a situation-specific design only considers the current situation at hand [16]. We can take an

automobile company designing a "world" car to be sold globally under on baseline model as an example. Suppose this company gathered the following data shown in Figure 3-3 on the height of drivers in country A,B and C.

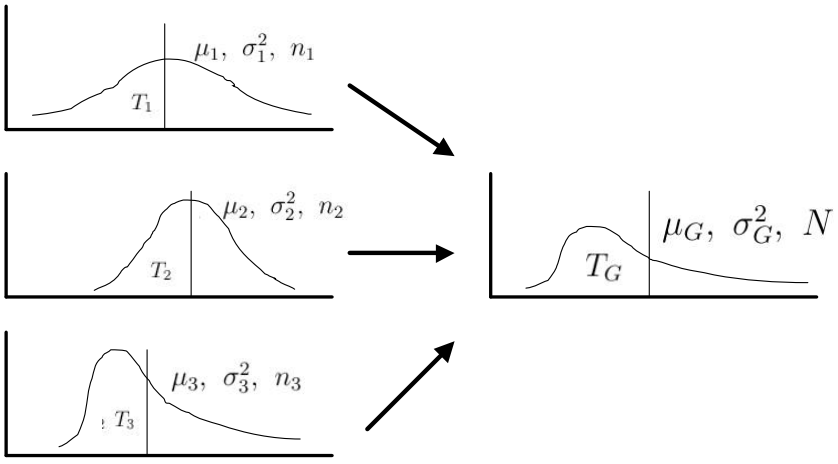

Figure 3-3: Examples of global design distribution [16]

Distributions will be associated with the random variables T_1, T_2 and T_3 respectively; with corresponding means, variances and sample sizes of $\mu_1,\ \sigma_1^2,\ n_1$; $\mu_2,\ \sigma_2^2,\ n_2$; and $\mu_3,\ \sigma_3^2,\ n_3$. If the company were to design different cars for each of these markets, the size and dimensions of each car would more than likely be tailored to meet the requirements of each country separately. However, if restricted to a one car design in which a combined global distribution, T_G with mean μ_G, variance σ_G^2 and sample size $N = n_1 + n_2 + n_3$, is utilized as the test data, then some compromises and added difficulties would be encountered. It is important to note here the following characteristics(see [16]):

$$\min(\mu_1,\ \mu_2,\ \mu_3) \leq \mu_G \leq \max(\mu_1,\ \mu_2,\ \mu_3) \tag{3.7}$$

$$\sigma_G^2\ \geq \min(\sigma_1^2,\ \sigma_2^2,\ \sigma_3^2) \tag{3.8}$$

There are three significant consequences that come out of these equations. First, the

mean of the global distribution will not be the same as the mean of the individual distributions, unless, of course, $\mu_1 = \mu_2 = \mu_3$. Second, the variance of the global distribution is larger than at least one of the variables of the other individual distributions(increase in overall uncertainty). And finally, the global distribution can be heavily biased toward individual distributions by having uneven sample sizes [16].

Use of a global distribution would likely result in modeling errors and therefore increase the overall uncertainty of the system since the design is not individually tailored to the current encounter situation. The threshold would instead be based on a weighted average of thousands of sample scenarios which may not be applicable to the current situation at hand [16]. Examples of scenario classification are presented in the next chapter.

3.5 Comparison of Iterative Trial and Error Method and Performance-based Design Approach

In this section, we are going to see the main difference between iterative trial and error method and performance-based approach.

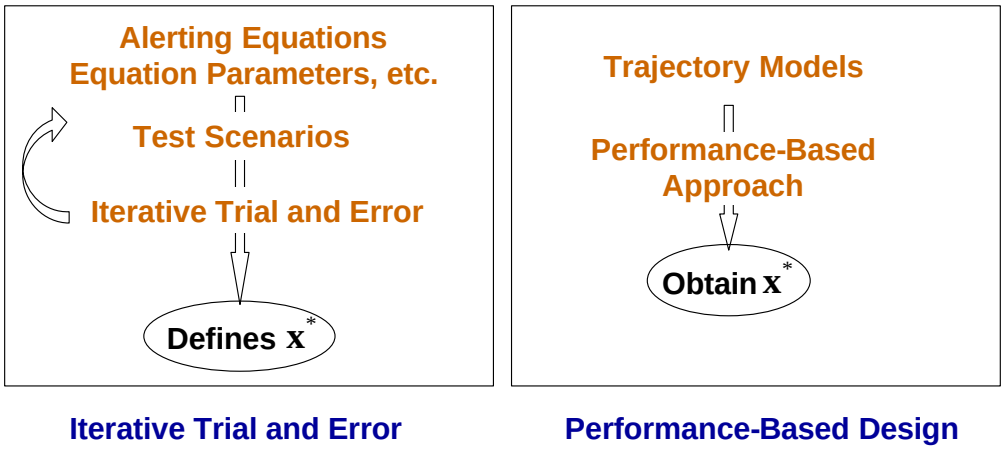

Figure 3-4: Schematic diagram - Iterative trial and error vs. Performance-based design approach

In the performance-based approach, rather than starting with alerting equations,

we start off by developing trajectory models of the vehicles that could also include uncertainties. Taking these trajectory models, we predict future states and determine what happens when an alarm is given or when it is not given, which then allows us to determines the performance of the system. We choose $\mathbf{x}^*$ based on this prediction.

Using $\mathbf{x}^*$, we can map back to any alerting equations to determine equation parameters. Essentially what we are doing is that we have $\mathbf{x}^*$, and choose any function f and solve this alert boundary equation for $\mathbf{a}$. A new, direct method is also proposed to overcome some of the limitations in the ad hoc approach [16].

3.6 Implementation of the Methodology : Design Process for a Rear-End Collision Avoidance System for Ground Vehicles

We now give an example of this methodology to design the threshold parameters for a rear-end collision avoidance system. In addition to setting the threshold, we will determine $\mathbf{a}$ for each of the 6 threshold functions from Table 3.2 to show the general applicability of this method, .

3.6.1 Define Operating/Encounter Scenarios

Using the NHTSA report [15] as a guideline, we examine three common types of rear-end collision scenarios listed in Table 3.3. Scenarios here are for two vehicles with LV (denoting the Leading Vehicle) and FV (denoting the Following Vvehicle). The alerting system mounted on the FV. Separate thresholds will be developed for each scenario and we assume the available states from sensors are $\mathbf{x} = [r\ \dot{r}\ v_F]$. To classify scenarios from raw sensor data, we introduced *if-then* statements and *neural network classification* within the framework of a larger hierarchical decision logic in the next Chapter.

A description of the scenarios is listed in Table 3.3. In Scenario 1, the leading vehicle LV is assumed to be stopped on a straight road and the following vehicle FV is approaching at constant speed. Scenario 2 means that the leading vehicle LV travels slower than the following vehicle FV with both vehicles at constant speed. In

scenario 3, the leading vehicle suddenly decelerates when the leading vehicle LV and the following vehicle FV travel at the same speed.

No.	Scenario
1	FV constant speed, LV stopped, straight road
2	FV constant speed, LV constant lower speed, straight road
3	FV and LV same constant speed, straight road, LV then decelerates

Table 3.3: Scenario definitions

3.6.2 Set Trajectory Models

The design process begins with defining trajectory models for both the nominal (no alert) and avoidance (alert) trajectories. The parameters used are shown in Table 3.4 and were extracted from distributions used to generate test scenarios for evaluating ACWAS systems in [15]. In Table 3.4, **Stdev** means standard deviation.

In alert trajectory for rear-end collision, which assumes the alert is given at the time, the driver is assumed to trigger the brake system only when he or she hears the warning signal. The reaction time for a human, when hearing the warning signal, is set as a normally distributed random variable with lower and upper bound. The mean of the normal distribution is 1.1 second and standard deviation is 0.305 second. This assumes that the driver begins to decelerate 1.1 second on average after hearing the warning signal. System delay of the vehicle is assumed as 0.2 second without any uncertainties. It has uncertainties but the amount of uncertainty is negligible compared to other distributions such as reaction time of human and braking deceleration. This means that the brake system of the vehicle is triggered 0.2 second after the driver starts to step on the brake pedal. The amount of deceleration the driver gives to the system is normally distributed with mean 0.4g and standard deviation 0.1g with bounds. Upper bound is 0.7g and lower bound is 0.2g.

In nominal trajectory, the driver is assumed to trigger the brake system when he or she perceives that the situation is hazardous. (Nominal trajectory for collision mitigation by braking is different. We could include steering actions for the nominal

Variable	Distribution	Mean	Stdev	Min	Max
Alert Trajectory					
Reaction Time	Gaussian	1.1 sec	0.305 sec	0 sec	2 sec
System Delay	Impulse	0.2 sec	0 sec	0.2 sec	0.2 sec
Braking Deceleration	Gaussian	0.4 g	0.1 g	0.2 g	0.7 g
No Alert Trajectory					
Inattention	Uniform	$\frac{1}{2}(r-2)$ m	$\frac{1}{\sqrt{12}}(r-2)$ m	0 m	r-2 m
Reaction Time	Lognormal	1.27 sec	0.72 sec	0 sec	∞
System Delay	Impulse	0.2 sec	0 sec	0.2 sec	0.2 sec
Braking Deceleration	Gaussian	0.4 g	0.1 g	0.2 g	0.7 g
Sensor Characteristics (GPS)					
Range (r) noise	Gaussian	0 m	3 m		
Range rate ($\dot{r}$) noise	Gaussian	0 m/s	0.4 m/s		

Table 3.4: Trajectory models

trajectory for collision mitigation by braking. We only include braking action without steering action.) System delay and braking deceleration is assumed same as in alert trajectory. We assumed that the time driver makes a decision (trigger the brake system) is uniformly distributed between the leading vehicle LV and the following vehicle FV. The reaction time of the driver in nominal trajectory is lognormally distributed. When the logarithm of the random variable is normally distributed, the distribution of the variable is called lognormal distribution. The distribution has a long tail and is not symmetric. The schematic diagram to compare both distributions - normal distribution and lognormal distribution - is depicted in Figure 3-5.

$$f_{\text{normal}}(x) = \frac{\exp\left\{-\dfrac{(x-\mu)^2}{2\cdot\sigma^2}\right\}}{\sigma\cdot\sqrt{2\cdot\pi}} \tag{3.9}$$

$$f_{\text{lognormal}}(x) = \frac{\exp\left\{-\dfrac{\ln\left(\dfrac{(x-\theta)}{m}\right)^2}{2\cdot\sigma^2}\right\}}{(x-\theta)\cdot\sigma\cdot\sqrt{2\cdot\pi}} \tag{3.10}$$

$$\tag{3.11}$$

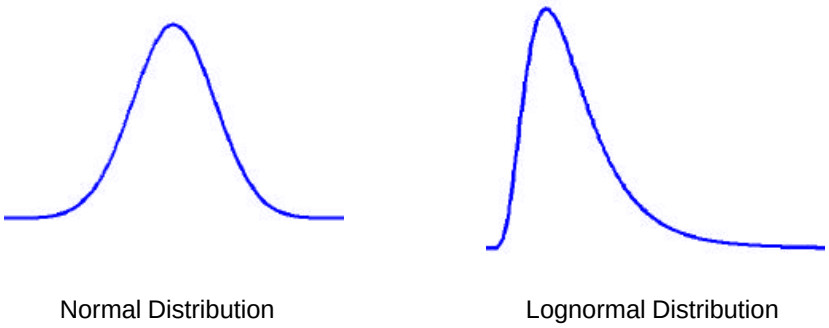

Figure 3-5: Normal distribution and lognormal distribution

3.6.3 SOC Analysis by Monte Carlo Simulation

Given the trajectory models and assuming simple dynamics of each vehicle, Monte Carlo simulations are done for different settings. States such as range between two vehicles(r), velocity of FV(v_F), velocity of LV(v_L), acceleration of FV(a_F) and acceleration of LV(a_L) are generated with given scenarios. Each scenario and simulation has different settings of the states. For example, $a_L = 0$, $v_F = 0$ for scenario 1. Assuming constant accelerations for both vehicles, the following simple dynamics equation is used for the simulation.

$$a = \frac{dv}{dt} \tag{3.12}$$

$$v = \frac{ds}{dt} \tag{3.13}$$

$$s = s_0 + v_0 \cdot t + \frac{1}{2} \cdot a \cdot t^2 \tag{3.14}$$

$$v^2 - v_0^2 = 2 \cdot a \cdot (s - s_0) \tag{3.15}$$

Monte Carlo simulations are done for $v_F = 25mph,\ 35mph,\ 45mph$, and $55mph$. For each of the four velocities, the SOC curve is generated from the results of the simulations along with a PM plot as a function of alert range, R. An example for Scenario 1 and $v_F = 35mph$ is shown in Figure 3-7.

Figure 3-6: States in Monte-Carlo simulation

3.6.4 Determination of Desired Alert Space

From each SOC curve, the desired operating point can be selected as a performance tradeoff between $P(SA)$ and $P(UA)$. This point corresponds to a specific combination of states $\mathbf{x}^*$ which we would like to the alert occur. Performance requirements, i.e., minimum $P(SA)$ and maximum $P(UA)$, may be set based on a designer's choice. For our examples, we chose the criteria $P(SA) \geq 0.95$ and $P(UA) \leq 0.25$ for Scenario 1 and 2, and $P(SA) \geq 0.90$ and $P(UA) \leq 0.60$ for Scenario 3.

Thus given a specific scenario, a threshold alert state $\mathbf{x}$ can be picked from SOC curves and/or PM plots. As shown in Figure 3-7, $68m$ is picked as the alert range for Scenario 1 and $v_F = 35mph$ since this alert state satisfies the previously defined design specification. Thus, $\mathbf{x}^* = [r^* \ \dot{r}^* \ v_F^*] = [68m, \ -35mph, \ 35mph]$. This procedure is repeated to span the over the region of state-space expected for an encounter. The results of $\mathbf{x}^*$ for the three scenarios are tabulated in Table 3.5.

3.6.5 Equation Parameter Determination from Alert States

Using the desired $\mathbf{x}^*$ data from Table 3.5, the alert boundary can be approximated with a number of appropriate functions including those from Table 3.2. The procedure is simply to solve for $\mathbf{a}$ to satisfy $f(\mathbf{x}^*, \mathbf{a}) = 0$. We compute a least-squares error solution for $\mathbf{a}$ - to minimize distance between the curve and data points. . An example using Scenario 1 ($v_L = 0$) is shown in Figure 3-8 and in Table 3.6.

Because $\dot{r}^* = -v_F$ in Scenario 1, only a 2-D plot of the state-space using $[r \ v_F]$ is shown in Figure 3-8 to reduce clutter. For f_1 and f_2, we chose to hold $B = 2m$ and

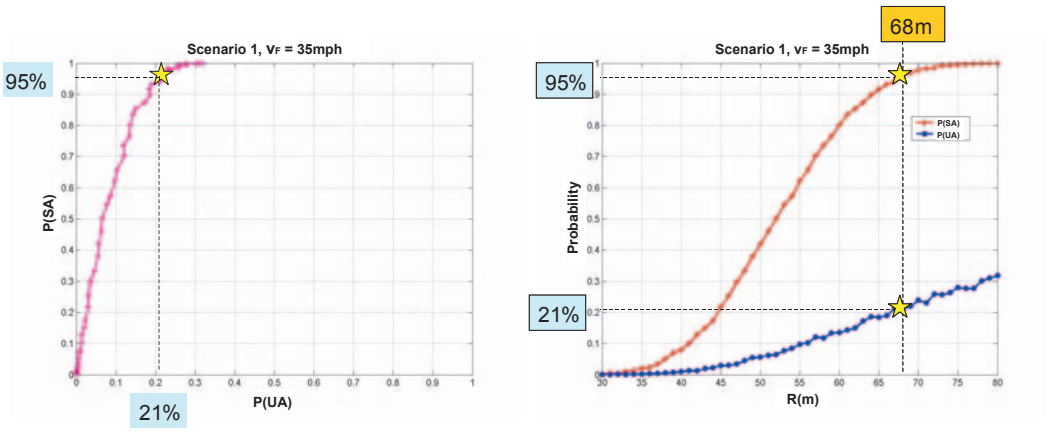

Figure 3-7: Example of warning distance selection from SOC curve and PM plot with GPS-based system

solve for the other equation parameters. The results show that all the functions in Table 3.2 can be used as suitable approximations of $\mathbf{x}^*$ in this scenario. However, f_1 and f_2 are slightly better than f_3 - f_6 because the latter functions are lines confined to pass through the origin. An additional curve fit was done using f_3 plus an added buffer parameter B to obtain a slightly better fit than without.

The resulting parameter values for all 6 alerting functions in the three scenarios are provide in Table 3.6. For Scenario 2 (LV travelling at a constant slower speed), f_2 provided the best least-squares fit, while for Scenario 3 (LV decelerating), f_1 was the best (shown in Figure 3-9). To obtain better results with the other equations, it is possible to break the functions into smaller piecewise curve fits and use *if-then* statements. However, this in effect is reformulating a new function to better match the desired $\mathbf{x}^*$.

The purpose is to have a simplified function or surrogate (e.g. f_1 - f_6) that would give the same result as if a tedious performance analysis was done instantaneously. Of course, this all assumes the trajectory model is correct in the first place.

As mentioned previously, mapping to a surrogate equation is unnecessary if the performance metrics can be computed in real-time [18, 16, 17].

Scenario1		$\dot{r}^*$			
		-25 mph	-35 mph	-45 mph	-55 mph
v_F^*	25 mph	42 m	·	·	·
	35 mph	·	68 m	·	·
	45 mph	·	·	102 m	·
	55 mph	·	·	·	142 m
Scenario2		$\dot{r}^*$			
		-5 mph	-15 mph	-25 mph	-35 mph
v_F^*	25 mph	25 m	31 m	·	·
	35 mph	34 m	39 m	52 m	·
	45 mph	42 m	48 m	61 m	81 m
	55 mph	50 m	56 m	70 m	95 m
Scenario3		$\dot{r}^*$			
		-5 mph	-10 mph	-15 mph	-20 mph
v_F^*	25 mph	31 m	35m	40 m	41 m
	35 mph	46 m	52 m	59 m	62 m
	45 mph	60 m	70 m	78 m	87 m
	55 mph	77 m	88 m	100 m	109 m

Table 3.5: The alert states $\mathbf{x}^* = [r^* \ \dot{r}^* \ v_F^*]$ from SOC analysis

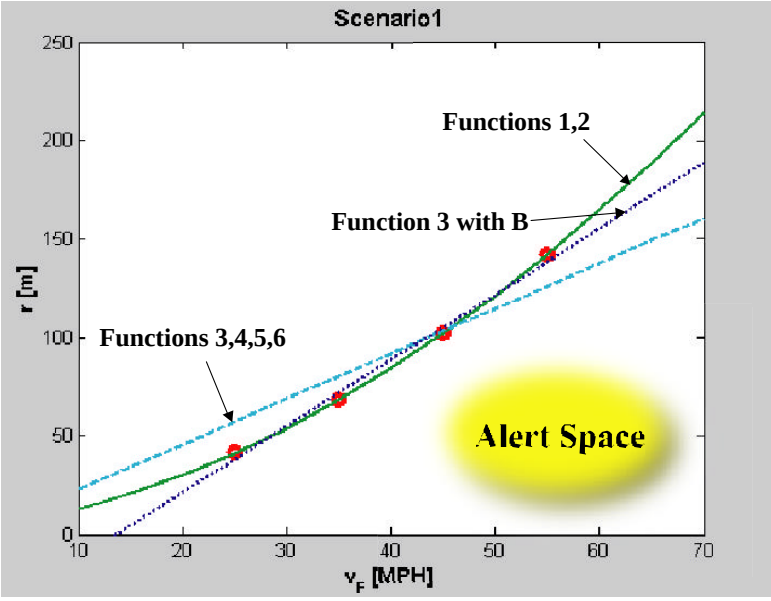

Figure 3-8: Alert boundary for scenario 1

No.	Scneario1	Scenario2	Scenario3
f_1	$\cdot$ $a_F = -3.0765$ $RT = 1.7$ $B = 2$	$a_L = -3.7907$ $a_F = -5.0$ $RT = 1.8$ $B = 2$	$a_L = -3.8961$ $a_F = -3.2$ $RT = 1.8$ $B = 2$
f_2	$a_F = -3.0765$ $RT = 1.7$ $B = 2$	$a_F = -2.8527$ $RT = 1.9$ $B = 2$	$a_F = -2.5908$ $RT = 3.2$ $B = 2$
f_3	$TTC = 5.1324$	$TTC = 5.9353$	$TTC = 10.1827$
f_3B	$TTC = 7$ $B = -36.66$	$TTC = 4.2$ $B = 21.2216$	$TTC = 3.2$ $B = 46.8075$
f_4	$TTC + SP/0.2778 = 5.1324$	$TTC = 3.4$ $SP = 0.4117$	$TTC = 1.8$ $SP = 0.8725$
f_5	$THW = 5.1324$	$THW = 2.9977$	$THW = 3.6625$
f_6	$TTC = 5.1324$	$TTC = 5.9353$	$TTC = 10.1827$

Table 3.6: Equation parameters developed from performance-based approach (see Table 3.2 for units)

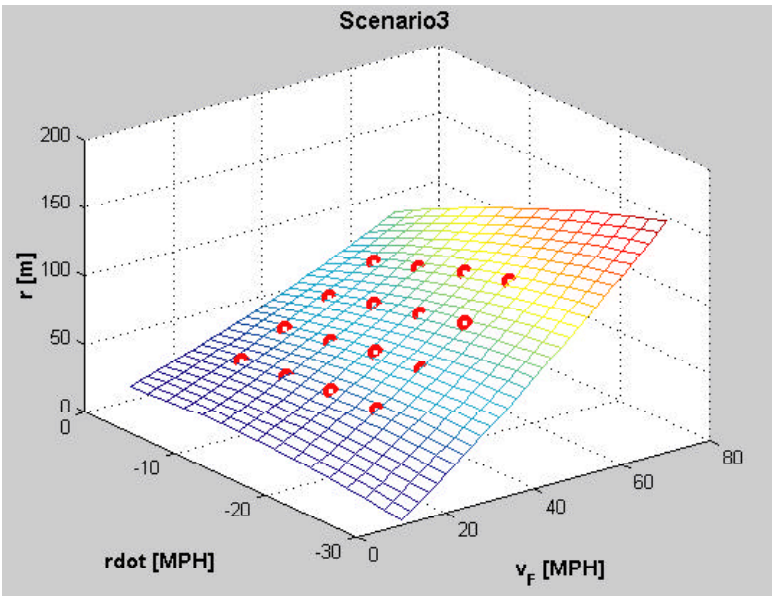

Figure 3-9: Alert boundary for scenario 3 using f_1

Chapter 4

Prototypes and Application Examples

In this chapter, prototypes of a rear-end collision warning system applying performance-based approach are presented along with several application examples. The methodology is installed in a ground vehicle and a driving simulator. A rear-end collision mitigation by braking system for real-time implementation is explored and the basic trajectory model is developed. To demonstrate universality of the approach, a vision-based collision warning system and a face tracking warning system is studied.

4.1 GPS-based System

4.1.1 System Architecture

A rear-end collision avoidance warning system based on the performance-based approach is implemented in a Lincoln LS concept vehicle. The original vehicle system structure other than warning system is developed by Ford Science Research Laboratory. The system description is mostly cited from a technical report [5]. The experiment equipment includes two vehicles (Lincoln LS and Ford Escape) with onboard GPS (Global Positioning System), a central processing unit in each vehicle and a laptop which runs the alerting algorithm in the Lincoln LS concept vehicle. The alerting algorithm was coded in the program Visual C++. Figure 4-1 shows the central processing unit in Lincoln LS concept vehicle and the laptop with the warning

system connected to the vehicle. The central processing unit is a SONY VAIO laptop PCG-Z505LEK.

Figure 4-1: Lincoln LS Setup- a central processing unit and a warning system unit

For the purpose of threat detection and mitigation, it is desired to have relative positions known to sub-meter accuracy with update times on the order of 10Hz [5]. Upon initial installation of a low cost GPS in the vehicle, the system will provide updates of the vehicle's position and velocity. The advantage of using GPS in the vehicle is that the service is available around the world and there is no cost to the user beyond the cost of the preinstalled unit. However, the performance of the system may degrade based on the constellation of GPS satellites at any given time. The performance depends on good reception of the satellites scattered across the sky for accurate positioning. The device is Garmin 35 12 channel receiver that updates at 1HZ and has 3-10 meters accuracy (with differential correction). The unit is self-contained in a weatherproof enclosure in the vehicle. It has built-in Kalman filters that compensate for selective availability and for statistical errors generated in the antenna and receiver[5].

Peer-to-peer wireless communication is achieved by using Orinoco PCMCIA WaveLAN cards, which has an outdoor range of 80 $\sim$ 100 meters. HyperAmp bi-directional amplifier(HA2401-AG1000 with automatic gain control 1 Watt, 2.4GHz)is used to enhance the wireless range of the Orinoco cards. From empirical test[5], the resulting range was improved to 400 meters, effective range to 200 $\sim$ 300 meters.

4.1.2 Scenario Classification

Using the performance-based approach, alerting thresholds were determined for three types of scenario situations as listed in Table 3.3. For scenario 0, there is no hazard and it shows a safe situation. The three scenarios were chosen using results from [15] as a guideline. Equation 1 of Table 3.2 was utilized as the threshold function for scenarios 1-3 while equation 5 was used for scenario 0. As explained previously in Chapter 3, other equations could have been used as the mapping function or the performance-based approach could be used in real-time. Application examples of real-time implementation is shown in the next section with a rear-end collision mitigation by braking system.

Before a threshold equation can be applied, the correct scenario situation must be determined. In the prototype algorithm, the decision tree logic in Figure 4-2 was used to select the corresponding scenario. Inputs from the GPS sensor are converted to relative range (r), relative range rate ($\dot{r}$), and velocity of the following vehicle FV (v_F). A neural net classifier is presented in the next section as another possible way of a scenario classification.

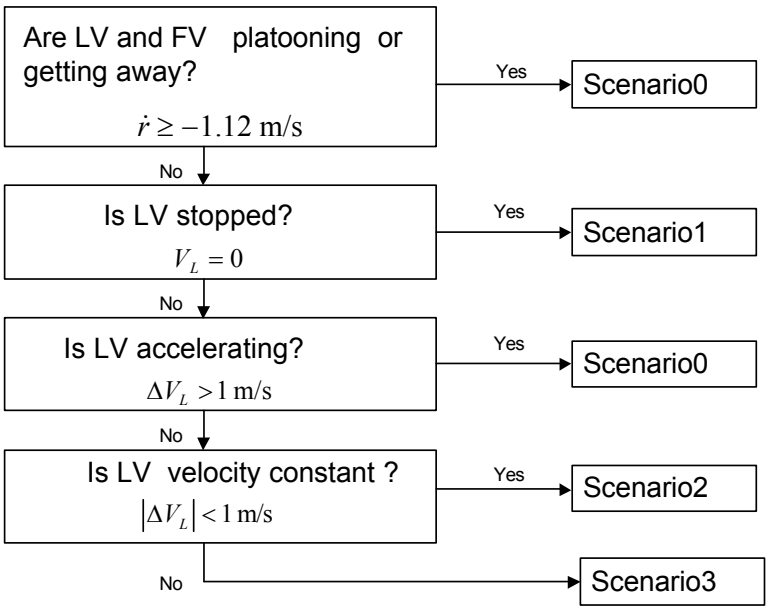

Figure 4-2: Scenario decision tree

4.1.3 Test Drive

The Lincoln LS was used for several test drives in city roads around Dearborn,MI, to examine the response of the alerting system. In an aggressive driving period of 146 seconds, 14 total alerts were triggered. In this situation, many alarms will alert the driver and it depends on the driver's preference if the alarms are annoying. It may be possible to carry out subjective statistical evaluations of the alerting performance. The overall performance of the alerting system can then be adjusted using this feedback. Or, if a proper trajectory model can be obtained in real-time, the system can be optimized to any specific driver.

As a side note, the driver in our test drives had commented that the alarms appeared late. This latency comes from the GPS delay which was not included in the trajectory models used in developing the alerting logic. This can be improved with a modification of the trajectory model or by using a upgraded GPS system.

4.2 STI Driving Simulator

STISIM is a low-cost, interactive driving simulator from Systems Technology, Inc. STISIM *Drive* Simulation System operates on personal computers with simulation hardware and uses Windows operating system interface. Simulation configuration options are easily accessible and simple scenario definition language(SDL) can be used for developing driving scenarios. The simulator operates main display(optional), driving display(s) and head-mount display(optional)[3]. AgeLab at MIT has a full-cab STISIM *Drive* simulation system. Figure 4-3 shows the appearance of the simulator. The desktop is a control station and we can see immediate visual and auditory feedback from the station.

At the AgeLab, we implement the same collision avoidance warning system as used in the vehicle but in the simulator. Real-time monitoring interface is captured in Figure 4-4. The open module is coded in Visual Basic; STISIM provides a template for additional user-defined algorithms in the simulator.

STISIM *Drive* with a collision warning system implemented aids to verify trajectory models originally used in the warning system. SDL(Scenario Definition Lan-

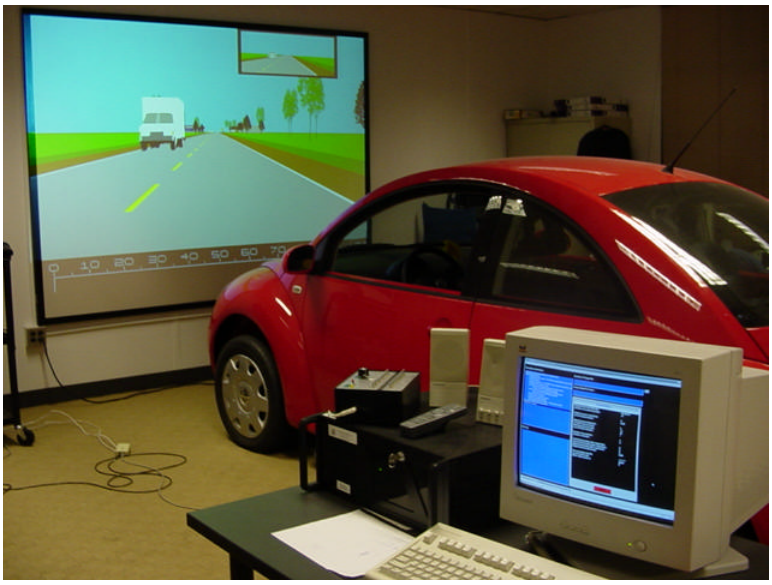

Figure 4-3: STISIM *Drive* Simulation system

guage) and open module allows one to set and to control environments and scenario sequences. Thus, driver behavior characteristics could be studied through the simulator with the warning system and without it. This can be utilized to build more realistic trajectory models.

4.3 Collision Mitigation-Simulation of Real-time Implementation

4.3.1 Data Collection

To simulate real-time collision mitigation system, we collect data from on-board sensor(radar) vehicle and feed the data through the Matlab Simulink model. Scenario 1 is examined only for a collision mitigation system.

The data collected from the test track for scenario 1 is plotted in Figure 4-5. First few seconds, the radar couldn't detect the stationary object since the distance between the vehicle and the object are larger than the radar range coverage. The radar starts to detect the target so that the vehicle ID changed from 0 to 1. At the same time, the relative range is obtained and we can see the relative range between

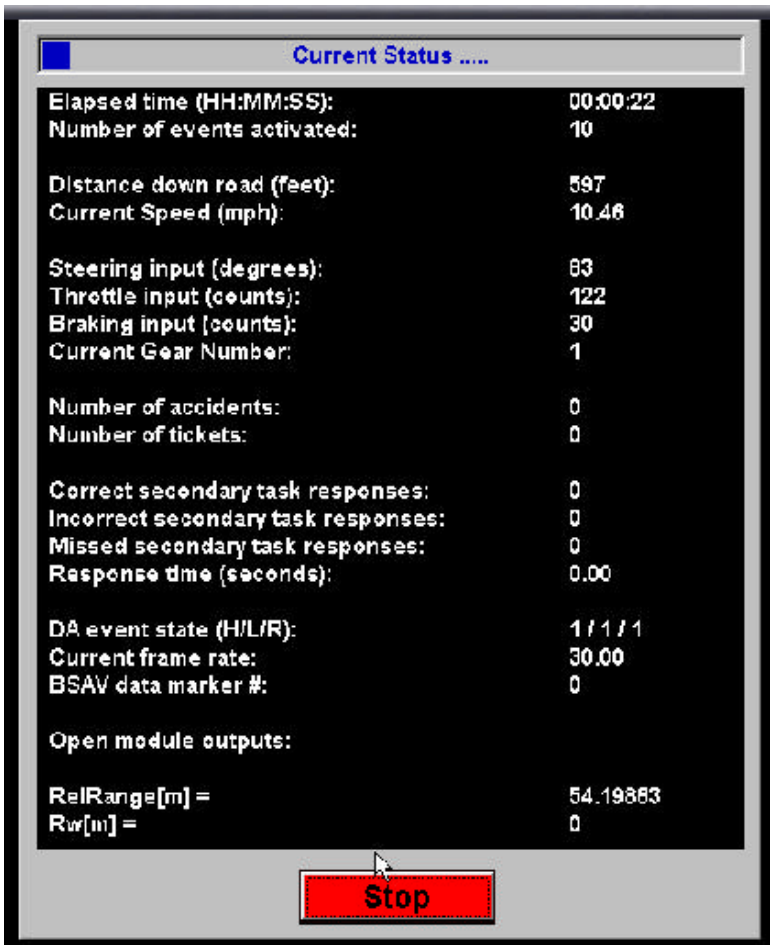

Figure 4-4: STISIM real-time monitoring interface

the vehicle and the object, which is decreasing according to the time since the vehicle is approaching the object. Relative range rate has constant negative value since the host vehicle was travelling in constant speed and the object was stationary.

4.3.2 Trajectory Model and Performance Metrics

In nominal trajectory for a collision mitigation system, the driver is assumed to trigger the brake system and the steering system simultaneously. Reaction time of the driver in nominal trajectory is assumed as a normally distributed random variable with lower bound 0 and upper bound 2. System delay of the vehicle is assumed as 0.2 second without any uncertainties. Imminent braking deceleration of the driver is assumed as a normal random variable with mean 0.5g and standard deviation 0.1g.

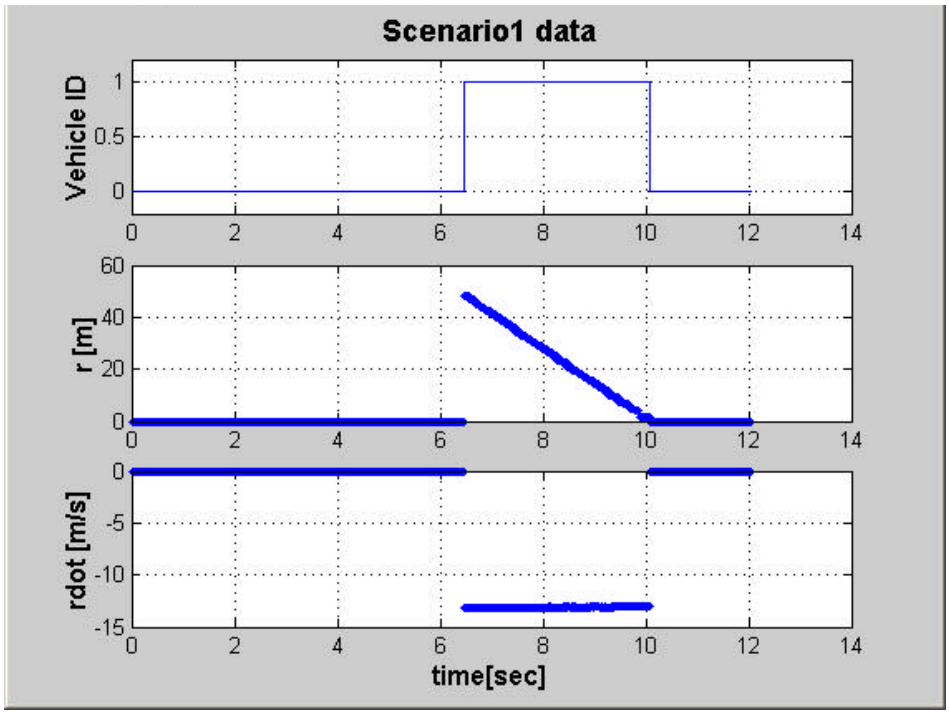

Figure 4-5: Scenario 1 data collected from a test track

Steering action is approximated to a linear motion and lateral acceleration is assumed uniformly distributed between 0.3g and 0.5g. (More accurate trajectory model could be developed considering tire dynamics, i.e., coupling between braking and steering force.)

In alert trajectory, the vehicle is assumed to trigger brake system automatically when the system decides that it enters a hazard space. System delay is assumed the same as in the nominal trajectory and the braking deceleration of the vehicle is assumed as a normal random variable with a mean 0.5g. In the alert trajectory for a collision mitigation system, human uncertainties are not included since we are assuming automatic braking system. The parameters used in trajectory models are shown in Table 4.1

Performance metrics for a collision mitigation system are defined as

$$P(SA) = 1 - P(\Delta v < 5\text{m/s}|\mathbf{A}) \quad (4.1)$$

$$P(UA) = 1 - P(C|\mathbf{N}) \quad (4.2)$$

Variable	Distribution	Mean	Stdev	Minimum	Maximum
Alert Trajectory					
System Delay	Impulse	0.2 sec	0 sec	0.2 sec	0.2 sec
Imminent Braking Deceleration	Gaussian	0.5 g	0.1 g	0.4 g	0.6 g
No Alert Trajectory					
Reaction Time	Gaussian	1.00 sec	0.333sec	0 sec	2 sec
System Delay	Impulse	0.2 sec	0 sec	0.2 sec	0.2 sec
Imminent Braking Deceleration	Gaussian	0.5 g	0.1 g	0.4 g	0.6 g
Lateral Acceleration	Uniform	0.4 g	0.0033 g	0.3 g	0.5 g

Table 4.1: Trajectory models

where Δv means the velocity difference of two vehicles at the instance when the collision is to occur it there is one. $P(SA)$ measures how well the mitigation system is working if the alert is is given at the time. The mitigation threshold is set to $\Delta v < 5$m/s. This is a different definition from $P(SA)$ in a collision avoidance system since having a collision or not doesn't result having a severe injury or not. The collision mitigation system aims to mitigate the severity of the collision so that the performance metric should be different from a collision avoidance system. $P(UA)$ has a same definition but the trajectory model is different.

Following shows a dynamic equations to calculate velocities at the instant when the collision is to occur. Equation 4.3 is a basic dynamic equations which could be derived from equations 3.12.

$$v_2^2 - v_1^2 = 2 \cdot a \cdot (s_2 - s_1) \tag{4.3}$$

Subscript 1 and 2 refers to corresponding instances. From the above equation, we can calculate the velocity of the following vehicle given initial speed, range, and some equation parameters.

$$v_{F_{collision}}^2 - v_F^2 = 2 \cdot a \cdot (R - v_F \cdot RT_{system}) \tag{4.4}$$

$$v_{F_{collision}} = \sqrt{v_F^2 + 2 \cdot a \cdot (R - v_F \cdot RT_{system})} \tag{4.5}$$

Through the Monte-Carlo simulation, we can obtain $P(SA)$ as follows where N is the

total number of runs.

$$P(SA) = 1 - \frac{\text{Number of collisions}, \Delta v < 5\text{m/s}}{N} \tag{4.6}$$

4.3.3 Performance Outcomes

A simulation result through the Simulink model with the real data gives performance outcomes for all corresponding simulation time step, which is set 0.01 sec in the model. Figure 4-6 shows $P(SA)$ and $P(UA)$ according to simulation time. From beginning to the time when the radar starts to see a target($\sim$ 6.5 sec, see Figure 4-5), both $P(SA)$ and $P(UA)$ are 1 since there are no objects, which means there are no potential threats. Whenever the radar starts to see a target, the performance metric calculation based on the trajectory model starts. After the radar detects a target, one can see that both $P(SA)$ and $P(UA)$ decreases to 0 eventually. Both

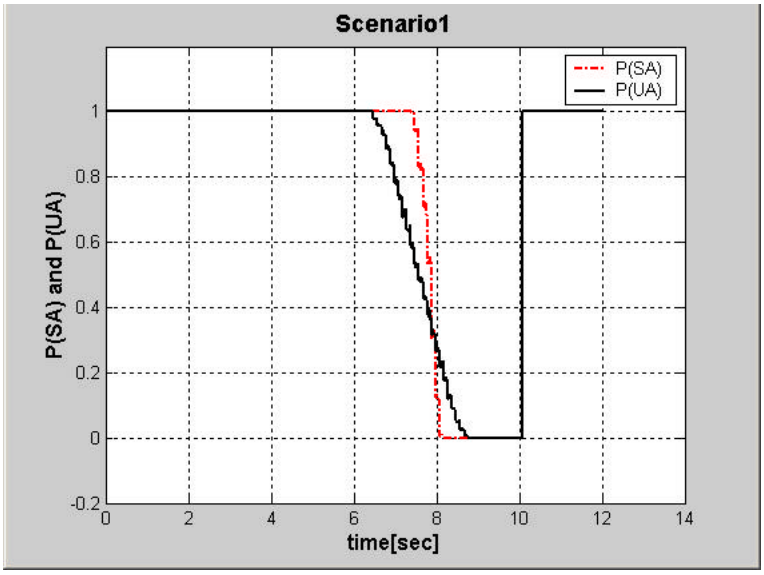

Figure 4-6: Real-time simulation of performance metric calculation

performance metric fall off as the host vehicle approaches to the object, which means that the threat level becomes higher and higher so that the probability to mitigate the collision decreases either giving an alarm or not. Though one sees that when or how both performance metrics decreases has different profile according to the

time. The difference comes from diverse distribution settings in trajectory models (see Table 4.1). Since the nominal trajectory model(for calculating $P(UA)$) has more uncertainties, $P(UA)$ starts to decrease earlier than $P(SA)$. We want to give a warning or an action at the time when $P(SA)$ is high enough so that the system has a chance to mitigate the collision. Though we want low enough $P(UA)$ not to bother the driver. This shows the typical tradeoff issues in alerting system design.

As the host vehicle approaches to the object, both $P(SA)$ and $P(UA)$ drops to 0, which means that there is no way to mitigate collision either given any action or not. Once the vehicle passes the stationary object, the threat is removed and both metric becomes 1.

4.4 Face Tracking Warning System

Since the performance-based approach can be used with any type of sensors, we can examine how we can use a face tracking technology as a sensor for hazard warning systems. Face tracking system referred to here is from Trevor Darrell's group at MIT AI (Artificial Intelligence) lab. The algorithm for the face tracking could be found in [13, 12]. A brief review of the face tracking system cited from [13, 12]tracking system to hazard warning system.

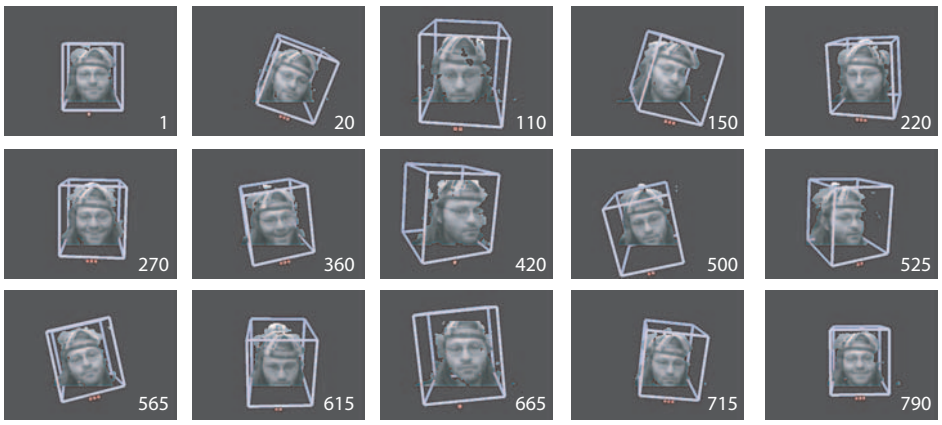

Figure 4-7: Head pose estimation using an adaptive view-based appearance model [13]

Adaptive view-based model consists of a collection of pose-annotate key frames

acquired by using a stereo camera during tracking. For each key frame, the view-based model maintains the following information:

$$\mu_s = \{I_s, Z_s, x_s\} \tag{4.7}$$

where I_s and Z_s are the intensity and depth images associated with the key frame s. The adaptive view-based model is defined by the set $\{\mu_1 \cdots \mu_k\}$, where k is the number of key frames. The pose of each key frame is considered as a Gaussian random variable whose distribution is to be refined during the course of tracking.

$$x_s = [T_x \ T_y \ T_z \ R_x \ R_y \ R_z] \tag{4.8}$$

is a 6 dimensional vector consisting of the translation and the three euler angles, representing the mean of each random variable. The view-based model also maintains the correlation between these random variables in a matrix Λ_χ, which is the covariance of these poses when they are stacked up in a column vector.

Tracking and pose adjustments to the adaptive view-based model are performed simultaneously. As the tracker acquires each frame, it seeks to estimate the new frame's pose as well as that of the key frames, using all data seen so far. That is, we want to approximate the posterior density:

$$p(x_t, x_\mu | y_{1\cdots t}) \tag{4.9}$$

where x_t is the pose of the current frame, $y_{1\cdots t}$ is the set of all observations from the registration algorithm made so far, and x_μ contains the poses of the key frames in the view-based model, $x_\mu = \{x_1 \cdots x_k\}$.

To analyze quantitatively the algorithm, Trevor's group compared results to an *Inertia Cube*^{2} sensor form InterSense. *Inertia Cube*^{2} is an inertial 3-DOF orientation tracking system. InterSpace reports an absolute pose accuracy of 3° RMS when the sensor is moving. Four sequences are recorded with ground truth poses using the *Inertia Cube*^{2} sensor. The sequences were recorded at 6 Hz and the average length is 801 frames (∼133sec). During recording, subjects underwent rotations of about 125 degrees and translation of about 90cm, including translation along the Z axis. The

RMS errors for all 4 sequences are shown in Table 4.2. The RMS error are within the accuracy limit of an attached inertial sensor.

	Pitch	Yaw	Roll	**Total**
Sequence1	2.88°	3.19°	2.81°	2.97°
Sequence2	1.73°	3.86°	2.32°	2.78°
Sequence3	2.56°	3.33°	2.80°	2.92°
Sequence4	2.26°	3.62°	2.39°	2.82°

Table 4.2: RMS error for each sequence. Pitch, yaw and roll represent rotation around X,Y, Z axis, respectively [13]

Since the face tracking system provides a full 6 DOF pose information and uncertainty characteristics, the system could be used in the hazard warning system with a proper trajectory model which predicts performance outcomes of the alerting system. However, the human driver behavior should be studied beforehand.

4.5 Vision-based system vs GPS-based system

To demonstrate the flexibility of the methodology, alerting criteria were developed for a second sensor system using a imaging device. For illustrative purposes, we videotaped the scene from a mounted video camera in the Lincoln LS test vehicle when the GPS-based system was working. Image processing was done off-line to extract the state variables.

Alerting thresholds for the vision-based system were determined utilizing the same method used in the GPS-based system. The only change was due to different sensor characteristics in the modeling of the trajectories. For the vision system, the range rate noise was assumed to be $3m/s$ while other components of the trajectory models in Table 3.4 were kept the same.

The uncertainties in the states derived from the sensor is a function of the image quality and the image processing logic. Assuming one-dimensional space and flat earth, an image processing logic was implemented to obtain a relative range between the FV and the LV. The main idea was to estimate the vertical pixel distance p_x to the edge between the road and the tires of the LV. As depicted in Fig.4-8, relative range (r) can be calculated in the vehicle frame from p_x. The Kalman filter was then

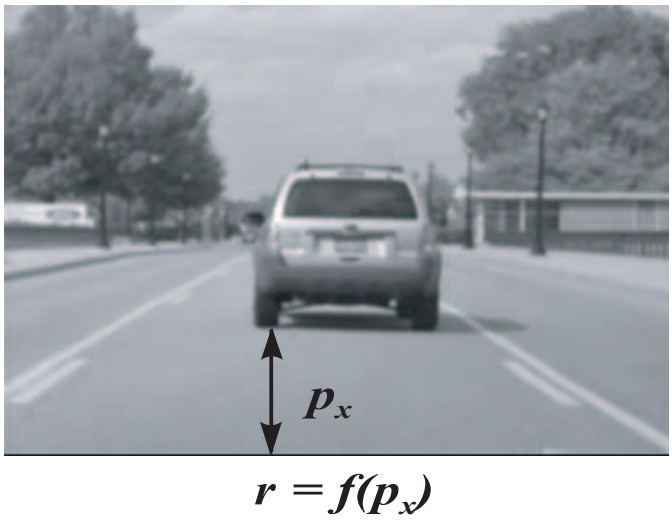

Figure 4-8: Image from on-board vision sensor

applied to obtain estimates of the true r and $\dot{r}$

The sensor characteristics between the two systems will result in slightly different performance. One of PM plot of $P(SA)$ vs. $P(UA)$ is compared in Fig.4-9 for Scenario 1 and $v_F = 35mph$. To achieve $P(UA) = 95\%$ for both systems, $P(UA) = 21\%$ for the GPS-based system while $P(UA) = 30\%$ for the vision-based system. The corresponding warning distance for the vision-based system is $75m$, which is longer than the $68m$ for the GPS-based system. The earlier alert distance is to account for the larger uncertainty in the vision-based system.

A sample time history with triggered alarms is depicted in Fig.4-10 for both systems. Since the prototypes are identical other than the sensor element, all differences come from different sensor characteristics. Region **A** shows that the vision-based system triggered the alarm earlier than the GPS-based system. However, since the GPS update rate is 1Hz and the vision sensor update rate is 15Hz, the vision-based system can detect the variation of the situation more quickly than the GPS-based system (region **B**). Given the same level of $P(SA)$ is required, we expect higher rate of false alarms in the vision-based system which appears in region **C**.

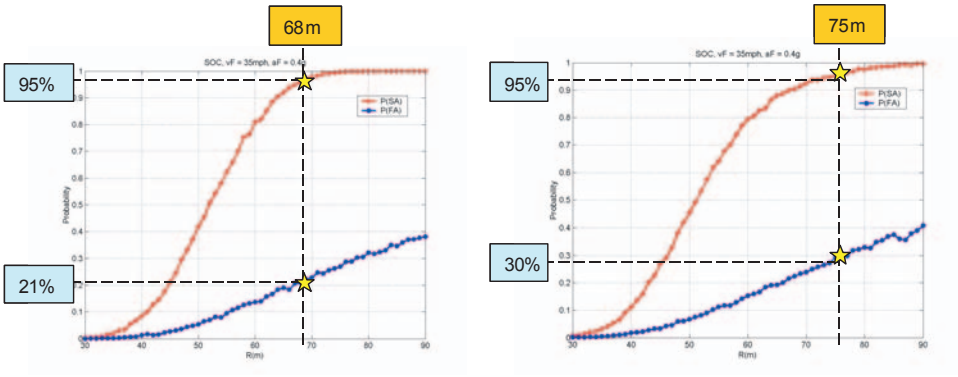

Figure 4-9: GPS-based system vs. vision-based system

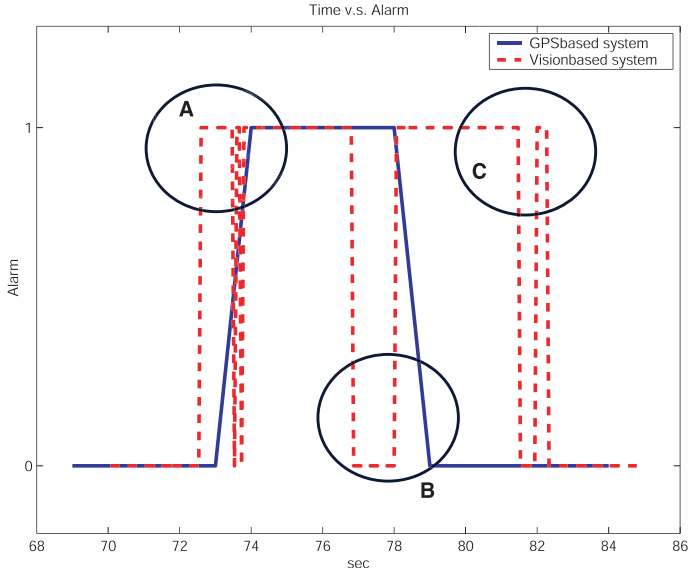

Figure 4-10: Comparison of triggered alarms : GPS-based system vs. vision-based system

4.6 Scenario Classification

Global design vs. situation-specific design is discussed in the previous chapter and a way to classify situations(scenarios) is given in this chapter in order to accomplish situation-specific design. Since there are huge uncertainties in trajectory models, a single global warning system or logic do not provide satisfactory performance. This requires a hybrid/hierarchical decision logic. Which is similar to hybrid control system with switching logic. The individual decision logics for each scenario can be thought of as multiple controllers with the classifier as the higher level supervisor deciding which one to use. A decision tree logic in Figure 4-2 was used for the prototype and a classification using neural net is presented in this section. The overall concept is depicted in Figure 4-11.

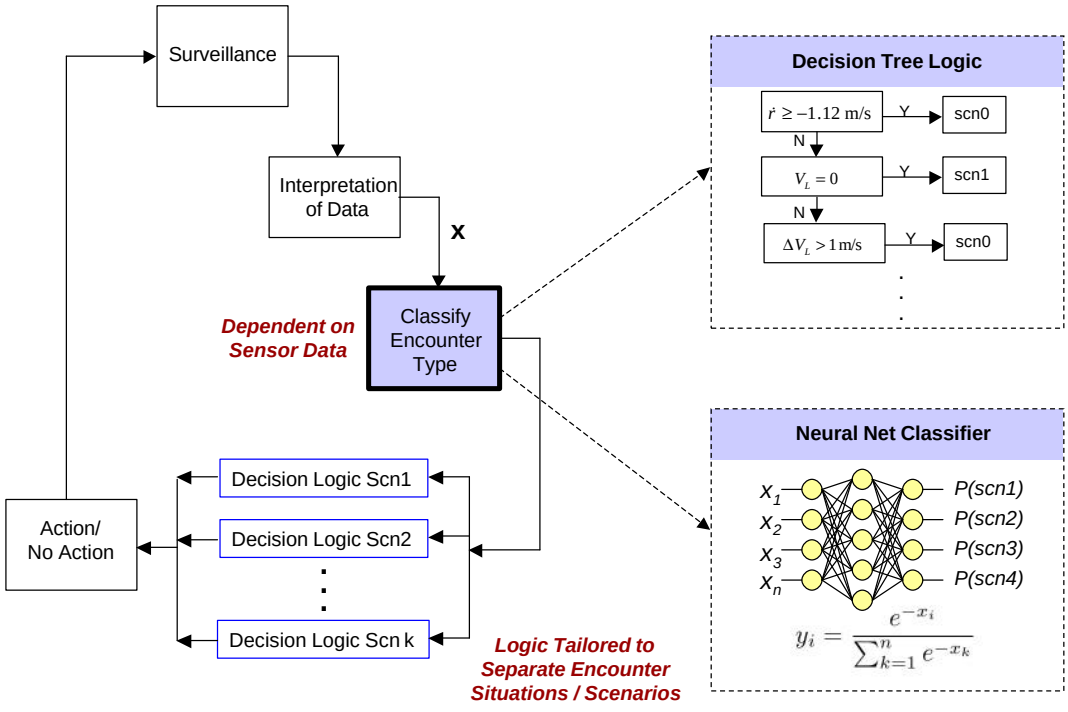

Figure 4-11: Hierarchical/hybrid decision logic

The goal of classification is to able to classify a given input into a relatively small

number of classes. We need *training data* for building models, *validation data* for stopping criterion and *test data* for evaluation of generalization accuracy [11]. Data set for each scenario is generated with some uncertainties in the trajectories.

Softmax(normalized exponential unit) is defined as

$$y_i = \frac{e^{-x_i}}{\sum_{k=1}^{n} e^{-x_k}} \qquad i = 1 \cdots n \tag{4.10}$$

where y_i is the activation function of the i^th output node and c is the number of classes. Note that this has the following good properties; it is always a number between 0 and 1, it gives a weight update proportional to (t-y) when combined with the error function[4]. Thus, the output of the neural net is probability of each event(scenarios) with n possible outcomes which sum up to 1.

Generated data set of scenario1, scenario2, scenario3, scenario0 and scenario p(passing scenario) is the classification object. Number of inputs to the neural net is 4,which consists of state of each scenario = $[r\ \dot{r}\ v_F\ a_L]$, where r is range between two vehicles, $\dot{r}$ is range rate between two vehicles, v_F is velocity for following vehicle and a_L is acceleration of leading vehicle. The six node hidden layer and the six node out layer is tried in this case. Weights and bias for each layer is sum up in Table 4.3.

net.w1	0.0722	-1.0458	0.0031	-0.1359	0.0351	-0.0008
	-9.8694	0.0840	-2.0937	1.1290	-0.3086	-1.3091
	0.0069	0.3834	-2.0684	1.0925	-0.5304	0.0143
	-0.6826	-0.0334	1.5663	2.3841	-1.1610	13.4638
net.b1	-2.8513	0.3702	1.5536	-0.1019	3.9612	0.0872
net.w2	2.9635	-1.7173	0.3699	-7.2356	4.9229	
	-0.5358	0.1378	0.7342	-2.2626	2.0018	
	7.8758	-2.4715	-2.0005	1.0613	-4.7065	
	-1.1873	-3.2549	-1.4780	3.5937	2.5868	
	0.8997	4.1645	1.6790	1.1689	-7.7634	
	1.0497	2.9531	-7.6873	10.0060	-5.6992	
net.b2	0.5832	-0.7989	-0.0603	-1.3642	1.3878	

Table 4.3: neural net - weights and bias

One of the result from the neural net is shown in Figure 4-12. Upper plot show

the training data of scenario 2- so all data set has $P(sc2) = 1$. If we put test data through the net, we obtain the bottom plot. Most of the case $P(sc2) \approx 1$, at least $P(sc2)$ has the maximum value among all other outputs.

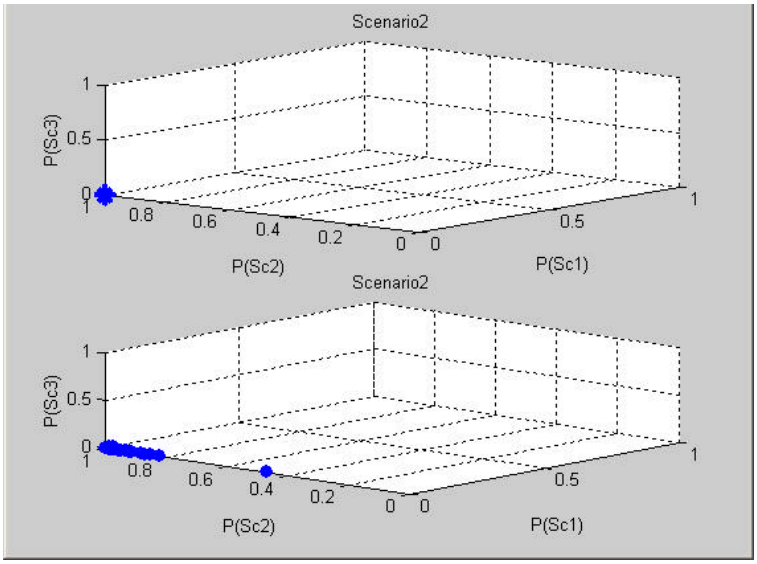

Figure 4-12: Training data and results from test data

Chapter 5

Summary and Conclusions

5.1 Summary

A brief overview of generic alerting system and state-space representation of alerting system are presented to provide as a groundwork for studying a performance tradeoff between positive and negative consequences of hazard alerting system. A review of System Operating Characteristic(SOC) curve and Performance Metric(PM) plot were given to provide an organized way of analyzing the performance outcomes.

A survey of conventional method of threshold placement was given and a unified methodology, a performance-based approach, was developed based on the study in previous chapters. Hazard alerting problem was focused on a threat prediction problem in the presence of uncertainties. Global design vs. situation-specific design issue was presented to show that a global design is a compromise between more situation-specific designs. An example of implementing the methodology was given for a rear-end collision avoidance warning system for ground vehicles.

Prototypes and application examples are shown utilizing the performance-based approach. The methodology was installed in a ground vehicle and a driving simulator. Several application examples of the performance-based approach was given as well. A rear-end collision mitigation by braking system for real-time implementation is shown and the basic trajectory model was developed. To demonstrate universality of the approach, a vision-based collision warning system and a face tracking warning system was studied.

5.2 Conclusions

Hazard alerting system such as a collision warning system can be effective components of the human-operated systems by providing protection against unforseen threats. The principal issue of the alerting system design is balancing a trade-off between increasing successful alarms and reducing annoying, unwanted unnecessary(false) alarms. A performance-based approach which can be used to model and evaluated alerting system performance was extended from [9, 16] and applied to a collision avoidance/mitigation system for ground vehicles. Diverse application examples, trajectory models are given after the methodology. The performance-based approach is a systematic method to the alerting system design. It can be used as a stand alone system or a threshold design tool in association with any threat assessment equation. Once proper trajectory models are determined, the alert space can be decided from an analysis of performance metrics, successful alerts and nuisance alarms. Furthermore, the methodology can be applied to any sensor system if the sensor characteristics can be included in the trajectory model. According to objectives of alerting system, the performance metric of the alerting system should be changed. The trajectory model can be modified by changing the metrics to include crash severity as opposed to simply whether a collision occurs or not for a collision mitigation system. To achieve maximum performance of the system, i.e. to operate at the optimized point on SOC curve, all parts of the hazard alerting system should be carefully developed. This includes characteristics of sensor, filter issues, classification of encounter scenarios and trajectory models which reflects a real world.

.1 Simulink Model : Collision Mitigation

Figure -1,-2,-3 and -4 shows the performance-based approach in a collision mitigation by braking described in Chapter 4. Figure -1 shows the Simulink model of the system which receives the real data and run through the block. Outputs of the system are $P(SA)$ and $P(UA)$. Vehicle ID, target range, rate of target range and vehicle speed are inputs of the system.

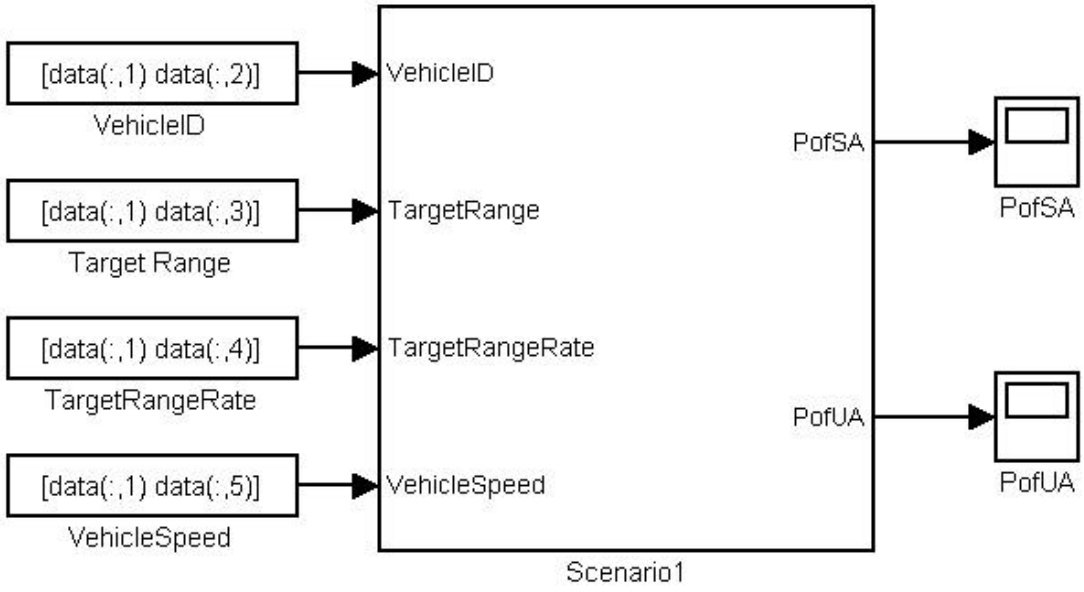

Figure -1: Collision mitigation by braking system

Subsystem of Figure -1 is described in Figure -2. The subsystem consists of two separate blocks; one is for generating random numbers based on inputs and the other is used for calculating performance metrics from trajectory models defined in Chapter 4.

Figure -3 shows random number generation routines according to the distribution given in trajectory models in Chapter 4. For example, RT is generated as uniform random number following settings in the trajectory model. Figure -4 shows how to calculate probability of collision when there is alarm or not. The calculation algorithm is explained in Chapter 4.

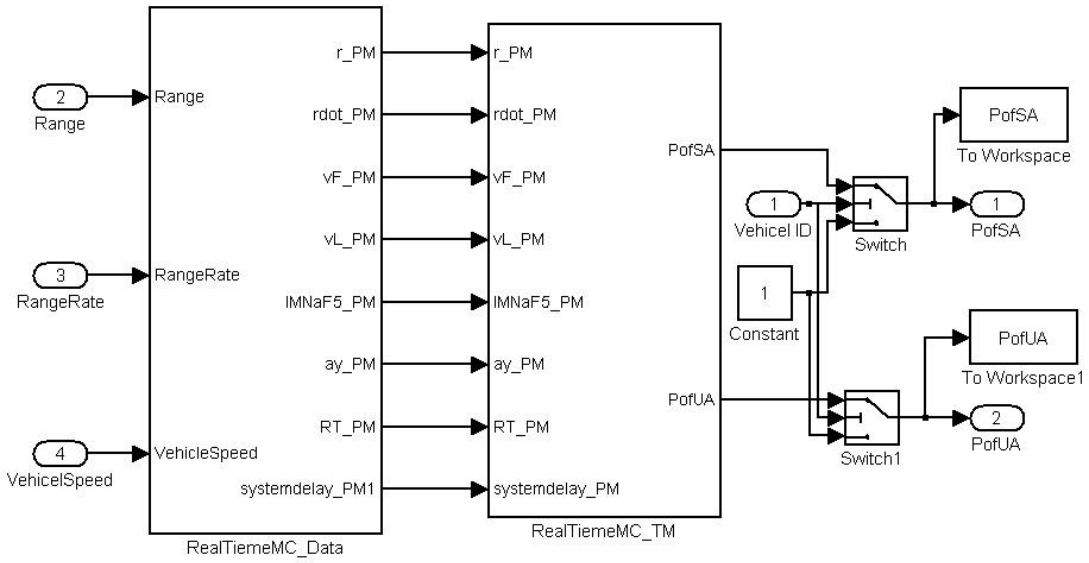

Figure -2: Subsystem of collision mitigation by braking system

.2 Linearization of vision system

From the range(r) obtained from the image sequences, relative position(x_1) can be calculated in Cartesian coordinate system. We assumed yaw angle(target angle) 0 for now. We use the range r in the image coordinate as measurements, which has a nonlinear transformation with relative position between the detected objects and the vehicle .

Since the states consist of relative position, velocity and acceleration, which is $x = [x_1\ x_2\ x_3]^T$, the state space equations are given as below. Assume the system is time-invariant and apply point-mass dynamics. (Free-Body Diagram and A matrix will be added here)

$$\dot{x} = Ax + Bu \tag{1}$$

$$y_v = C(x) + v \tag{2}$$

x : state vector $[x_1\ x_2\ x_3]$

A : state transition matrix in the absence of a forcing function

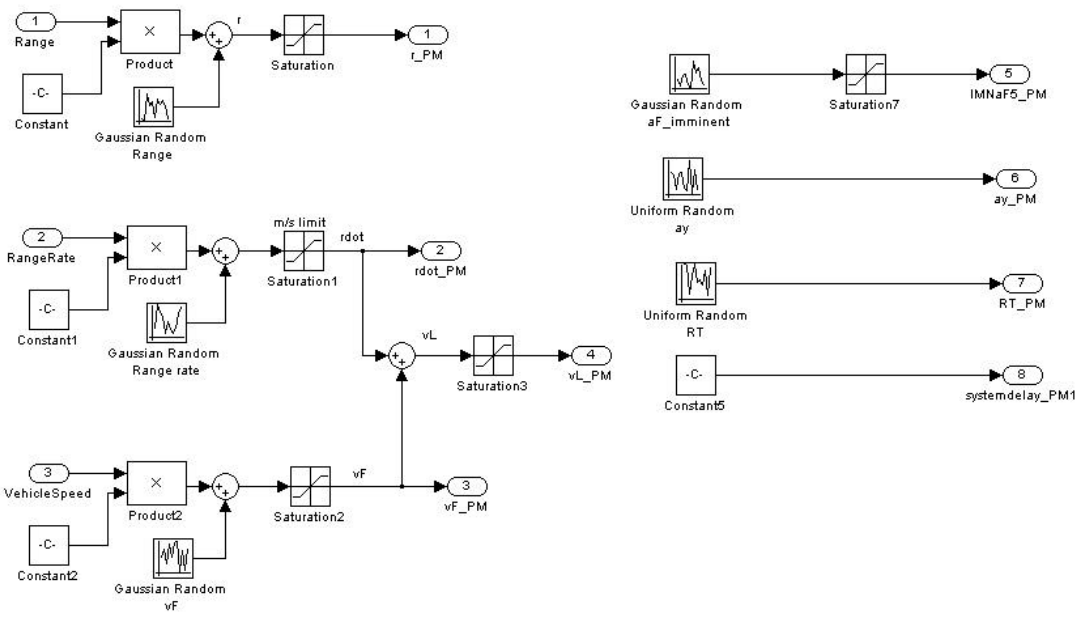

Figure -3: Random number generation block

$C(x)$: Nonlinear transformation giving the ideal(noiseless) connection between the measurement and the state vector

v : Measurement error- assumed to be a white sequence with known covariance structure having zero cross correlation with the

u : No input assumed

y_v : Measurement (Range only from the image sequences)

We are going to include 'Heading angle' as a measurement later on. 'Yaw rate' or 'Heading angle' could be used as elements of the state vector to describe circular maneuver more effectively.

A detailed explanation of nonlinear transformation $C(x)$ is presented here. The pixel values obtained from image sequence are related to the range between the detected object and the vehicle by the following nonlinear relations.

$$y = h \cdot \left(\frac{f \cdot \sin\alpha + x \cdot \cos\alpha}{f \cdot \cos\alpha - x \cdot \sin\alpha} \right) \tag{3}$$

f: the focal length of the lens system

h: the height of the focal point above the ground

α: the elevation angle of the optical axis

Then, linearize the transformation about prior estimate of $[x_1\ x_2\ x_3]$, which we call $[\bar{x}_1\ \bar{x}_2\ \bar{x}_3]$. Then apply Kalman filter technique to the linearized version of $y_v = C(x) + v$

$$\begin{aligned} dy &\doteq y - \bar{y} && (4)\\ &\cong \left.\frac{\partial C}{\partial x}\right|_{x=\bar{x}} (x - \bar{x}) + v && (5)\\ &\doteq \frac{\partial C}{\partial x} dx + v && (6)\\ & && (7) \end{aligned}$$

Therefore, measurement matrix C can be determined with given variables.

$$y - \bar{y} = [C_1 \quad C_2 \quad C_3] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + v \tag{8}$$

$$\begin{aligned} C_1 &= \left(\frac{\partial y}{\partial x_1}\right)_{x_1=\bar{x}_1, x_2=\bar{x}_2, x_3=\bar{x}_3} && (9)\\ &= \frac{fh^2}{(fh\cos\alpha - h\sin\alpha)^2} && (10)\\ C_2 &= \left(\frac{\partial y}{\partial x_2}\right)_{x_1=\bar{x}_1, x_2=\bar{x}_2, x_3=\bar{x}_3} && (11)\\ &= 0 && (12)\\ C_3 &= \left(\frac{\partial y}{\partial x_3}\right)_{x_1=\bar{x}_1, x_2=\bar{x}_2, x_3=\bar{x}_3} && (13)\\ &= 0 && (14) \end{aligned}$$

$$C = \begin{bmatrix} \dfrac{fh^2}{(fh\cos\alpha - h\sin\alpha)^2} & 0 & 0 \end{bmatrix} \tag{15}$$

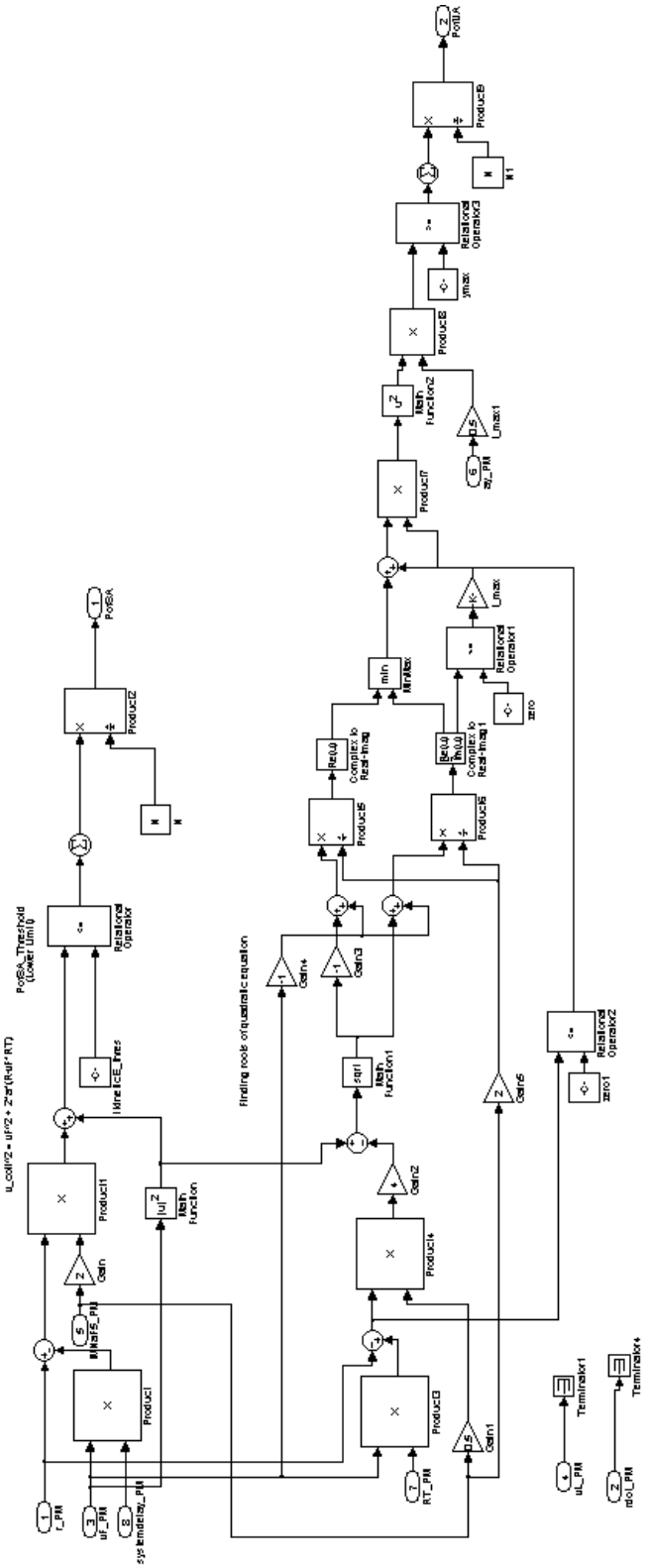

Figure -4: Performance metric calculation

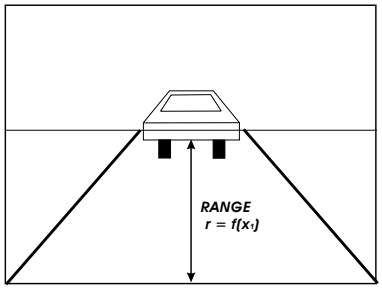

Figure -5: Schematics of Images Obtained from the Camera Inside the Vehicle

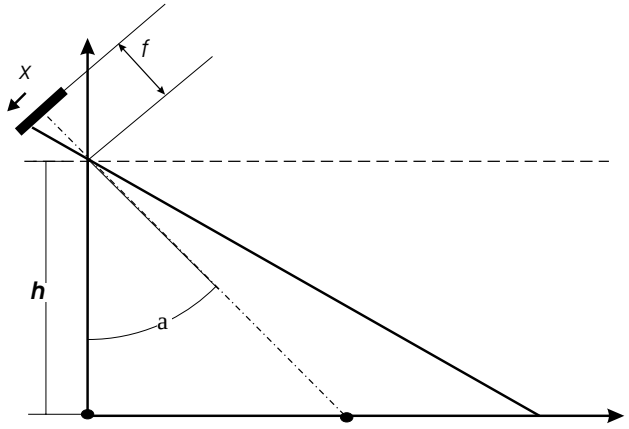

Figure -6: Schematics of Longitudinal Axis Perspective Transformation

Bibliography

[1] Brown, T. L., Lee, J. D., and McGehee, D. V. " Human Performance Models and Rear-End Collision Avoidance Algorithms". *Human Factors, Vol. 43*, No. 3 (Fall 2001).

[2] Hirst, S., and Graham, R. *Ergonomics and Safety of Intelligent Driver Interfaces,The Format and Perception of Collision Warnings.*

[3] http://www.systemtech.com.

[4] http://www.willamette.edu/ gorr/classes/cs449/Classification. Doing Classification Correctly.

[5] Huang, Q., Miller, R., MacNeille, P., DiMeo, D., and Roman, G. C. "Development of a Peer-to-Peer Collision Warning System". Tech. rep., Washington University, Ford Research Lab.

[6] Kim, Y. S. "Effects of Driver, Vehicle, and Environment Characteristics on Collision Warning System Design". Master's thesis, Department of Science of Techonology, Linkoping Institute of Technology.

[7] Kirk, D. E. *Optimal Control Theory.* Prentice-Hall, Inc., 1970.

[8] Knipling, R. R., Mironer, D. L. Hendricks, L., Everson, J., Allen, J. C., and Wilson, C. "Assessment of IVHS Countermeasures for Collision Avoidance: Rear-End Crashes". Tech. Rep. DOT HS 807 995, NHTSA, May 1993.

[9] Kuchar, J. K. *A Unified Methodology for the Evaluation of Hazard Alerting Systems.* PhD thesis, Department of Aeronautics and Astronautics, MIT, 1995.

[10] Kuchar, J. K. "Methodology for Alerting System Performance Evaluation". *AIAA Journal of Guidance, Navigation and Dynamics, Vol. 19*, No. 2 (March-April 1996).

[11] Lab(SCAI), S. C. A. I., 2001. Bioinformatics Chapter 5. Neural Networks: The Theory.

[12] Morency, L.-P., and Darrell, T. "Stereo Tracking using ICP and Normal Flow". In *Proceedings Int. Conf. on Pattern Recognition* (2002).

[13] Morency, L.-P., Rahimi, A., and Darrell, T. "Adaptive View-based Appearance Model". In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition* (2003).

[14] (NHTSA), N. H. T. S. A. "Automotive Collision Avoidance System Field Operational Test/First Annual Report". Tech. Rep. DOT HS 809 196, NHTSA, December 2000.

[15] (NHTSA), N. H. T. S. A. "Automotive Collision Avoidance System Field Operational Test/Phase 1 Interim Report". Tech. Rep. DOT HS 809 453, NHTSA, May 2002.

[16] Yang, L. C. *Aircraft Conflict Analysis and Real-Time Conflict Probing Using Probabilistic Trajectory Modeling*. PhD thesis, Department of Aeronautics and Astronautics, MIT, 2000.

[17] Yang, L. C., and Kuchar, J. K. "Using Intent Information in Probabilistic Conflict Analysis". In *AIAA Guidance, Navigation, and Control Conference, Boston, MA* (August 10-12 1998).

[18] Yang, L. C., and Kuchar, J. K. "Performance Metric Alerting: A New Design Approach for Complex Alerting Problems". *IEEE Transactions on Systems, Man and Cybernetics, Vol. 32*, No. 1 (January 2002).

[19] Yang, L. C., Yang, J. H., Feron, E., and Kulkalni, V. "Development of a Performance-Based Approach for a Rear-End Collision Warning and Avoidance

System for Automobiles". In *IEEE Intelligent Vehicle, Columbus, OH* (June 9-11 2003).

[20] Zador, P., Krawchuck, S., and Voas, R. "Automotive Collision Avoidance System (ACAS) Program/First Annual Report". Tech. Rep. DOT HS 809 080, NHTSA, August 2000.