UC Berkeley
Research Reports

Title

Intelligent Diagnosis Based On Validated And Fused Data For Relilability And Safety Enhancement Of Automated Vehicles In An IVHS

Permalink

https://escholarship.org/uc/item/1mw2v298

Authors

Agogino, Alice
Chao, Susan
Goebel, Kai
et al.

Publication Date

1998

eScholarship.org Powered by the California Digital Library
University of California

This paper has been mechanically scanned. Some errors may have been inadvertently introduced.

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

Intelligent Diagnosis Based on Validated and Fused Data for Reliability and Safety Ehnancement of Automated Vehicles in an IVHS

Alice Agogino, Susan Chao, Kai Goebel, Satnam Alag, Bradly Cammon, Jiangxin Wang
University of California, Berkeley

California PATH Research Report
UCB-ITS-PRR-98-17

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

The contents of this report reflect the views of the authors who are responsible for the facts and the accuracy of the data presented herein. The contents do not necessarily reflect the official views or policies of the State of California. This report does not constitute a standard, specification, or regulation.

Report for MOU 231

April 1998

ISSN 1055-1425

INTELLIGENT DIAGNOSIS BASED ON VALIDATED AND FUSED DATA FOR RELIABILITY AND SAFETY ENHANCEMENT OF AUTOMATED VEHICLES IN AN IVHS

Final Report
PATH Project
MOU-231

January 1998

Principal Investigator

Alice Agogino
Tel.: (510) 642-6450
Fax.: (510) 643-8982
aagogino@me.Berkeley.EDU

Postdoctoral Researchers/Graduate Researchers
Susan Chao
Kai Goebel
Satnam Alag
Bradly Cammon
Jiangxin Wang

Department of Mechanical Engineering
University of California at Berkeley
Berkeley, CA 94720

Intelligent Diagnosis Based on Validated Fused Sensor Data for Reliability and Safety Enhancement of Automated Vehicles in an IVHS

Alice M. Agogino, Susan Y. Chao, Kai Goebel,
Satnam Alag, Bradly L. Cammon, and Jiangxin Wang
Department of Mechanical Engineering
UC Berkeley

I. Acknowledgments

Funding for this project was made available by PATH grant MOU 231. The methodology shown in this report is part of the dissertation by Satnam Alag (chapters 2-6) and Kai Goebel (chapters 2 and 4). We acknowledge the help of Brad Cammon who worked on this project as part of his MS project. We also appreciate the help of undergraduate student researcher Isela Villanueva, graduate student researcher Jiangxin Wang, visiting fellow Thomas Larsen, and PATH engineer Dr. Seibum Choi.

11. Keywords

Sensor Validation, Sensor Fusion, Data Fusion, Supervisory Control, Management of Uncertainty, Reliability, Safety, Bayes Networks, Fault Detection, Diagnosis, Influence Diagrams, Risk Analysis, Decision Making

111. Abstract

Vehicles in an IVHS system rely heavily on information obtained from sensors. ***So*** far, most control systems make the implicit assumption that sensor information is always correct. However, in reality, sensor information is always corrupted to some degree by noise which varies with operating conditions, environmental conditions, and other factors. In addition, sensors can fail due to a variety of reasons. To overcome these shortcomings, sensor validation is needed to assess the integrity of the sensor information and adjust or correct as appropriate. In the presence of redundant information, sensor data must be fused, accommodating the findings from the validation process.

In this report, we address the above issues which is an extension of the previous work completed by us in MOU 132 (Agogino et al, 1995) and MOU 157 (Agogino et al., 1997) in the area of sensor validation and fusion. The data driven supervisory control activities are concerned with fault detection, fault isolation, and control reconfiguration of the many sensors, actuators and controllers that are used in the control process. In order to carry out corrective actions (control maneuvers) that maintain the overall integrity of the IVHS system, the sources of uncertainty will be considered before arriving at the final diagnosis of the vehicle state.

To achieve this, we performed failure mode effect analysis, coupled with intelligent diagnosis techniques (such as influence diagrams/Bayesian networks and fuzzy logic) at the coordination layer. Furthermore, an intelligent decision adviser was designed which forecasts potential hazards and provides recommendations on potential maneuvers and other actions to the coordination level

controller in an optimal manner. This supervisory controller uses validated fused data, serves as a link between the vehicle sensors and the coordination layer and ensures the proper operation of the system in diverse and adverse operating conditions. This is an essential part of the ITS program required to provide a safe and reliable system. We also continued to update our sensor characterization models through a preliminary evaluation of GPS sensor testing.

IV. Summary

Presently, control systems in IVHS assume that data obtained from sensors are correct. However, sensor data are uncertain because of the presence of noise and sensor failure. This may result in control actions which are unsafe for passengers of the IVHS. We see the solution to the diagnosis as a five-module approach. The modules as explained in detail in (Alag, Goebel, and Agogino, 1995a) are: 1) Sensor validation, 2) Sensor fusion, 3) Fault Detection, 4) Hazard Analysis, and 5) Safety Decision Maker. The first two modules and, to a certain degree, the third module, were addressed in previous funding projects (MOU 132 and MOU 157).

This report is mainly concerned with the remaining modules — Fault Detection, Hazard Analysis, and Safety Decision Maker — which takes the validated and fused sensor data as input and gives probabilities of hazards as output. It also analyses these hazards to recommend appropriate actions. It is crucial to not only detect failures in the system, but also determine its cause and to classify (for e.g. total or incipient) its effect on the system. Our past work has clearly shown that various sensors react in different ways under the influence of different operating conditions such as fog or rain. To systematically capture the effects and reasons for failures and to match the signature of faulty sensor readings with specific failures we carried out detailed failure mode effect analysis for the longitudinal sensors. This analysis allowed us to match faulty sensor readings with a specific failure.

Failures can to a degree be predicted and actions to situations sensitive to certain sensor readings can be taken in advance not as a reaction to an already hazardous situation but as a preventive measure. In previous work, we have been developing a detailed methodology for sensor validation and fusion. We further augmented that work by investigating (or developing) other techniques such as fuzzy data validation and fuzzy fusion for settings which include potentially emergency situations or the lead car control problem where no proper model of the system is given. Serving as a backup system which relies less on mathematical modeling, it may be advantageous also for emergency situations of the follower situation during maneuvers.

Another contribution in this report is the development of a monitoring system which uses sensor observation data about discrete events to dynamically construct a probabilistic model of the vehicle's world. This model is a Bayesian network which can reason under uncertainty about both the causes and consequences of the events being monitored. Belief networks can also be used to complement the analytical methods for fault detection that are being developed by other PATH researchers. The combination of both strategies allows the evaluation of all available information and knowledge of the system for fault detection such as degree of aging, the operational environment (e.g., the vehicle state whether lead vehicle or follower vehicle etc.). We will use influence diagrams (Bayes' networks with the addition of decision/control and cost/value nodes) for decision making to avoid or avert potentially dangerous states. These decisions are to be optimized with respect to safety, low jerk, and smooth riding criteria.

We used off-line optimization techniques because they are too slow to run in real time within the AVCS framework. Later, a real-time version can be developed by "compiling" out the control strategies. The state of the IVHS system on the platoon level can be modeled as an optimization problem, with possible states of each of the vehicles defining the feasible stochastic search space. The states of individual vehicles in a platoon are the design variables in the optimization process.

The objective function in this case was multi-attribute consisting of a list of possible hazards and the probability of each occurring obtained from the Intelligent Decision Module. Finally, continuing with our ongoing work in **PATH** we continued to test, characterize, and compare sensors operating under diverse and adverse conditions including new ranging sensors entering the scene, e.g. GPS sensors.

Table of Contents

List of Figures

List of Tables

Final Report

PATH
MOU 231

Intelligent Diagnosis Based on Validated Fused Sensor Data for Reliability and Safety Enhancement of Automated Vehicles in an IVHS

Alice M. Agogino, Susan Y. Chao, Kai Goebel,
Satnam Alag, Bradly L. Cammon, and Jiangxin Wang
Department of Mechanical Engineering
UC Berkeley

1. Introduction

The Intelligent Vehicle Highway System (IVHS) envisions significant increases in safety and highway capacity through the integration of control, communication and computing technologies (Varaiya, 1991; Varaiya and Shladover, 1991). An early analysis indicates that with a completely automated vehicle control system, freeway lane capacity could double or even triple under improved safety conditions. In the IVHS paradigm, closely spaced automated vehicles will travel at high velocities in their respective lanes. In order to perform its basic functions (such as longitudinal control, lateral control, platooning, maneuvering techniques, e.g., lane change, automated lane exit), the IVHS requires a large number of sensors for control at the coordination layer, the engine layer, for sensing, and for communication between the vehicles and the IVHS main controller.

For all subsystems to work well and reliably the IVHS system requires high sensor data fidelity. However, most generally, sensor readings are uncertain because internal and external sources add noise to the readings or cause a malfunction of the sensor altogether. Relationships between the sensor readings and the system being monitored are non-deterministic. No sensor will deliver accurate information at all times. Consequently, the safety of the IVHS system is affected. It is therefore desirable to find a way to avert the negative effect of the shortcomings of the sensors. An inconsistent sensor reading could be a result of a system failure, process failure, or sensor failure, and it is important to distinguish between these failure types. It is also important to account for sources of uncertainty and propagate them to the final diagnosis of the system state.

A framework for real-time monitoring and diagnosis of the automated vehicle in IVHS was developed in MOU 132 (Agogino et al., 1995). This framework consists of five modules within the automated vehicle control system (AVCS) hierarchy. They can be found on two layers, the regulation layer and the coordination layer. This intermediate supervisory controller combines the advantages of having access to the data of the regulation layer as well as the information from the coordination layer. It operates in every vehicle and serves the purpose of real-time monitoring and diagnosis of the components in the vehicle and between the vehicle and its environment (other objects in its neighborhood). It predicts incipient failures and recommends suitable corrective actions. This research builds on the framework and the preliminary validation and fusion methods developed in MOU 157. Therefore, we will briefly review the supervisory control methodology in the IVHS context and introduce the two new approaches taken by us for MOU 157 to build the supervisory controller.

Since safety is of prime importance in an IVHS system, it is imperative to first validate and fuse the uncertain readings obtained from the numerous sensors. Therefore, the first task to be carried out by the supervisory controller is to validate the sensor readings and get an estimate for the various

parameters to be used in the monitoring and fault diagnosis part. Our experience in past work (Kim et al., 1992) has shown that real-time fault diagnosis for complex systems benefits from a hierarchical information processing structure with the selection of faults on one layer, focusing in more detail on these candidate faults at a higher layer, and finally looking for facts to confirm the ultimate diagnosis and make repair/recovery recommendations at yet a higher layer. The approaches taken include the integration of heuristics and model-based reasoning, procedures for fusing qualitative and quantitative data for developing assessments for estimation of probabilities or fuzzy memberships, and explicit reasoning about the time constraints inherent in real-time processing of large amounts of data.

We therefore proposed a multi-layer architecture for real-time monitoring and diagnosis of the automated vehicle, which consists of modules for sensor validation, sensor fusion, fault diagnosis, hazard analysis, and a safety decision advisor. Before the different modules of the system are introduced, a brief discussion will illustrate how the supervisory controller proposed here is integrated into the AVCS. Figure 1-1 shows the outline of the complex hierarchical structure of the AVCS system control architecture which – in addition to the shown link, coordination, regulation, and physical layer – consists of the network layer at the top.

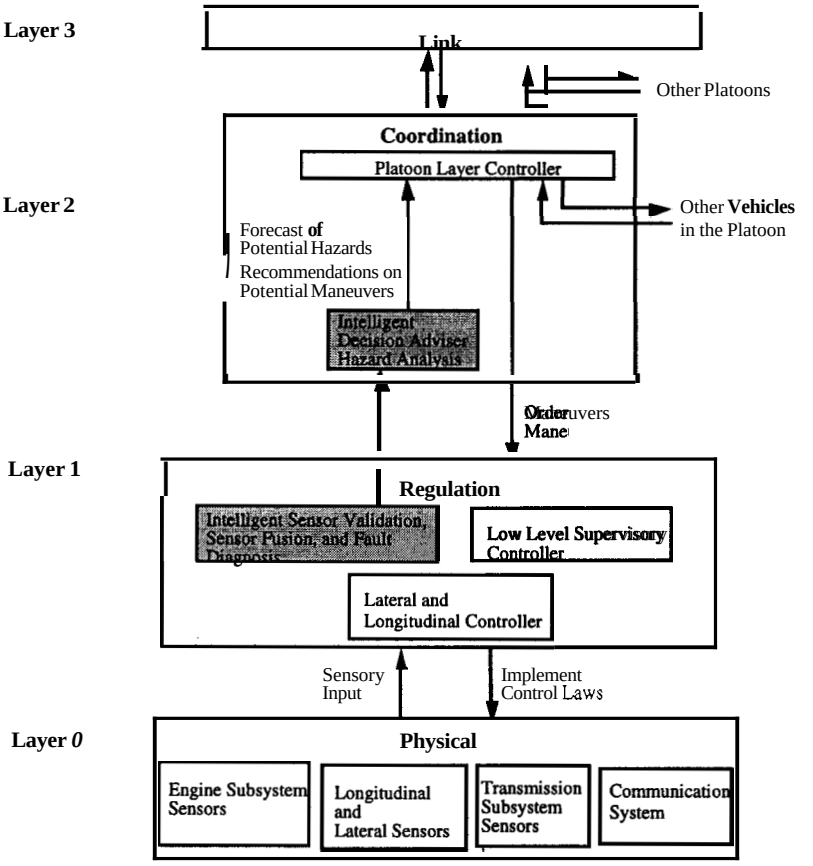


Fig. 1-1: Position of Intelligent Sensor Validation, Sensor Fusion, Fault Diagnosis, Hazard Analysis, and Intelligent Decision Advisor in the AVCS Control Hierarchy

The supervisory decision advisor considers the uncertainties in sensor readings and forms a link between the coordination layer controller and the regulation layer controller and rectifies aberrant sensor readings by taking into account the information of several partly redundant sources.

Sensor validation and sensor fusion take place at the regulation layer (shaded box on the regulation layer in Fig. 1-1). Input are data from the sensors of the physical layer. The fault diagnosis of the various subsystems is also located at the regulation layer. Some of the reasons for uncertainty in sensory information are measuring device error, environmental noise, and flaws or limitations in the data acquisition and processing systems. Extracting information from raw data is often difficult because of noise, missing data or occlusions. Phenomena may show up at disparate locations and can have a variety of time scales, from low frequency signals to high frequency vibrations. Therefore, the regulation layer seems to be the appropriate place for sensor validation and sensor fusion as it permits access to various sensor values at the same time. On the coordination layer, output from the sensor validation and fusion module as well as from the fault diagnosis module are used to perform hazard analysis and intelligent decision making because this central location within the control architecture allows for integration of all relevant information.

The modules are displayed in an isolated format in Fig. 1-2. The input and output for each module are shown on the right hand side. On the lowest layer is the sensor validation module, responsible for detecting sensor failures and sensor faults. After the validation, in the case of multiple sensors, or a group of sensors measuring a set of related quantities, sensor fusion takes place. Redundancies of the sensors as well as correlation of processes measured by different sensors are utilized to find fused or corrected sensor values.

In MOU 132 we investigated the suitability of the Probabilistic Data Association Filter (PDAF). Based on the results of the sensor validation and fusion modules, a fault diagnosis module looks at potential failures of the various subsystems and calculates their respective probability or fuzzy likelihood. Here, subsystem influence diagrams are used to capture the influences of various failures on subsystem parameters. This information (either Bayesian or fuzzy) will then be used in a hazard analysis module to compute the likelihood of various hazards.

Finally, an intelligent decision advisor is proposed which provides recommendations on potential maneuvers and other actions to the coordination layer controller. This decision advisor needs to arrive at the optimal decision in real-time. Since reaction time has to be small, optimization techniques which are generally very slow are trained off-line to obtain optimal responses for various scenarios. These can be implemented by means of look-up tables and pattern recognition systems and then used on-line in real-time. In this way a link between the vehicle sensors and the coordination layer is achieved. This will improve the operation of the system in diverse and adverse conditions.

This report introduces the new methodology that we developed for MOU 231 to be used for fault detection, diagnosis, and decision making. To begin, a fuzzy approach for diagnosis is introduced in Chapter 2 followed by a probabilistic approach in Chapter 3. Chapter 4 shows the fault tree analysis and hazard analysis and in Chapter 5, the decision making tool is introduced. Preliminary analysis on new sensors is described in Chapter 6 and a SmartPATH simulation of previous completed work on sensor validation and fusion is discussed in Chapter 7.

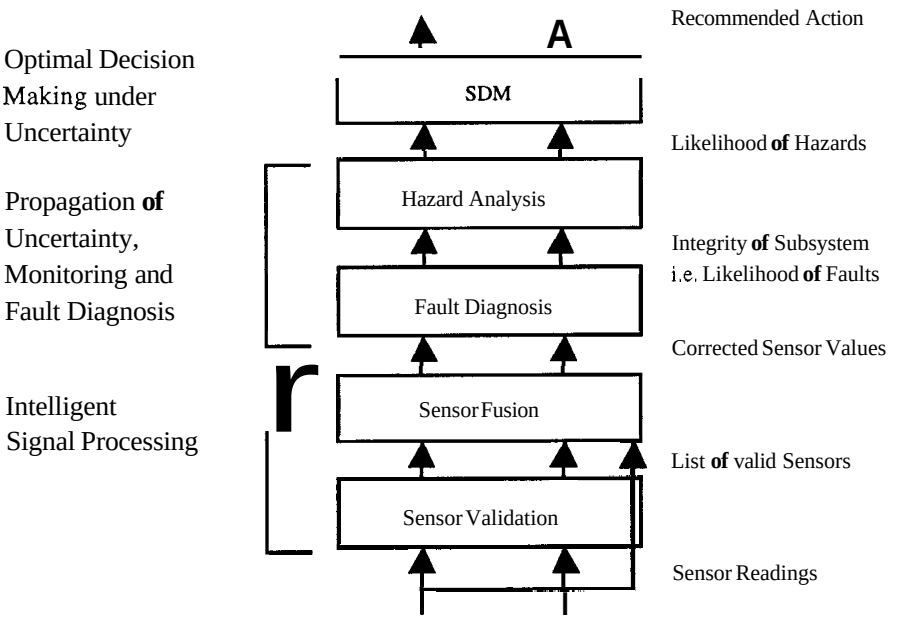


Fig. 1-2: Framework of 5 modules for sensor validation, fusion, fault diagnosis, hazard analysis, and the safety decision maker

2. Sensor Validation and Fusion Using A Fuzzy Approach

Section 2.1.1 introduces a new algorithm for fuzzy diagnosis. Mechanical diagnoses can be problematic in that there are usually several symptoms which correspond to several failures. Whereas there are a wide variety of techniques for "forward" reasoning from cause to effect (failure to symptom) using deterministic, probabilistic, or fuzzy means, "backward" reasoning from symptom to failure is in contrast poorly understood.

Bayes' rule can be used in a probabilistic diagnostic framework to infer failure from the observation of a symptom. We develop such a system in Section **3.** However, obtaining the probabilities necessary may be a very difficult task. They must be obtained through measuring the events over a long period of time or through estimation from an expert. Furthermore, general probabilistic inference is an NP-hard problem, sometimes requiring compromising assumptions of conditional independence to be computationally tractable. In this Section we develop an alternate method based on the use of fuzzy logic. No knowledge about the frequency of symptoms is necessary. Rather, expert knowledge is taken from fuzzy cause-effect relationships modeled via causal diagrams. "Backward" reasoning becomes possible with the introduction of an appropriate fuzzy measure.

In Section 2.1.2, examples from platooning in IVHS are used to show how the system manifests sensor failure when the vehicles are subject to a set of maneuvers. Real time diagnosis is instrumental in ensuring a safe ride. Information from this diagnosis will be used to carry out actions designed to avert negative effects of sensor and actuator failure and in the case of the IVHS example to safeguard the vehicles and their passengers in the presence of an emergency.

2.1 Fuzzy Diagnosis

This section introduces a new method for diagnosis of sensor and actuator malfunctions using fuzzy techniques. The key to fuzzy diagnosis is the inversion of the cause-effect relation to conclude from the presence of symptoms to a particular failure. To achieve this, a fuzzy measure is introduced which will assign a degree of similarity with each possible failure. Both sudden and gradual malfunctions can be diagnosed in a real time fashion.

We build on the notion of abduction using a fuzzy scheme. Inference in abduction is a method of selecting one specific solution from a large number of hypothetical solutions consistent with the data. In binary logic both rule and symptom are evaluated with respect to their truth. **A** rule is added to the possible hypothesis set only when both rule and symptom are evaluated to be true. In multi-valued logic — such as in fuzzy or probabilistic systems — both rules and results are always true to some extent and therefore all rules can be hypothesized to some degree. It is therefore necessary to come up with a way to find a method which identifies the most likely hypothesis.

Such a scheme will be introduced here. This scheme makes use of fuzzy causal diagrams. To begin, failure-symptom relationships are expressed in fuzzy causal diagrams as displayed in Fig. 2.1-1 where the f_n represent the failures and the s_m stand for the symptoms. To avoid overcrowding of the graph, links with strength zero were omitted. This means that a fault f_n causes a number of symptoms s_m to occur to some extent. That is, some symptoms have stronger mappings to faults than others. Other faults may cause the same symptoms but with a different degree of strength.

The fuzzy connection between fault and symptom can be encoded in a fault-symptom matrix. Each fault causes a number of symptoms to some degree. With the assumption that several faults will cause the maximum value of both individual symptoms and that there are no mutually excluding failures, modeling of multiple concurrent faults can be achieved as seen in Table 2.1-1. Here, all

possible failure combinations are enumerated from no failure at all to the case where all failures occur simultaneously.

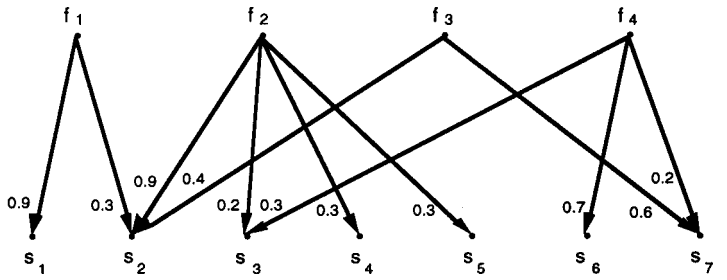


Fig. 2.1-1: Fuzzy causal diagram

s 1	s 2	s 3	s 4	s 5	s 6	s 7	f1	f2	f3	f4
0	0	0	0	0	0	0	0	0	0	0
0	0	.3	0	0	.7	.2	0	0	0	1
0	.4	0	0	0	0	.6	0	0	1	0
0	.4	.3	0	0	.7	.6	0	0	1	1
0	.9	.2	.3	.3	0	0	0	1	0	0
0	.9	.3	.3	.3	.7	.2	0	1	0	1
0	.9	.2	.3	.3	0	.6	0	1	1	0
0	.9	.3	.3	.3	.7	.6	0	1	1	1
.9	.3	0	0	0	0	0	1	0	0	0
.9	.3	.3	0	0	.7	.2	1	0	0	1
.9	.4	0	0	0	0	.6	1	0	1	0
.9	.4	.3	0	0	.7	.6	1	0	1	1
.9	.9	.2	.3	.3	0	0	1	1	0	0
.9	.9	.3	.3	.3	.7	.2	1	1	0	1
.9	.9	.2	.3	.3	0	.6	1	1	1	0
.9	.9	.3	.3	.3	.7	.6	1	1	1	1

Table 2.1-1: Fault-symptom matrix

Symptoms rarely appear as crisp occurrences. Rather, all symptoms will at any time be measured to some degree. Therefore, symptoms are encoded via fuzzy logic and are then used as input to the proposed diagnostic system. The goal is to decide which of all possible hypotheses is the most likely one. Therefore, we require a ranking scheme which will discard the less likely hypotheses and rank the most plausible one on top. If two fault combinations are equally likely, the set of failures with minimal cardinality will be chosen in accordance with parsimonious covering theory.

A fuzzy measure of closeness is proposed which is motivated by the notion of subsethood (Kosko, 1992) and its Lukasiewicz equivalent (Dalton, **1994).** We distinguish two cases: faults can occur in either a crisp manner (power outage, electrical short, etc.) or in a soft manner (gradual failure, increasing bias, dependency of performance on temperature, etc.). These two situations are accounted for with two related closeness measures.

2.1.1 Crisp Failures

If faults are known to be crisp, then the distance of the measured symptom to the symptom set for the closest fault will be determined. Evidence $S^+(F_I)$ is aggregated by summing up the Euclidean distance of the observation to the symptom set of a particular fault combination in the symptom-failure space as shown for a two-dimensional case with two symptoms and four faults in Fig. 2.1.1-1. The missing fourth fault is the Ø-fault which is assumed to be at the origin. $S(F_I)$ expresses the symptom set for a fault combination F_I. By definition, $S(F_I)$ is either 0 or 1 in the crisp case.

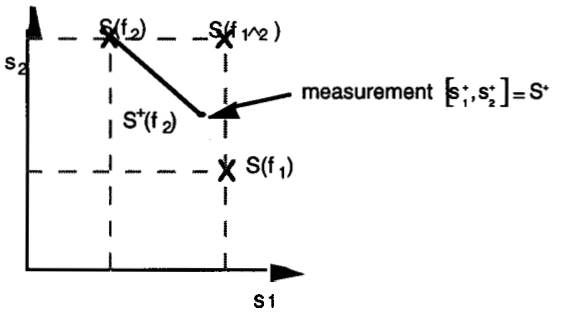


Fig. 2.1.1-1: Aggregation of Evidence in the symptom-failure space for crisp faults and fuzzy symptoms

Ranking takes place after the distances to all fault combinations have been obtained. The highest numeric value of the closeness measure is taken as the most likely one. The closeness measure g_c for the crisp case is of the form

$$\gamma_c\left(S(F_I), S^+\right) = 1 - \min\left(1, 1 - S(F_I) + S^+(F_I), 1 + S(F_I) - S^+(F_I)\right)$$

where

$S(F_I)$ is the particular symptom set for fault combination F_I

$$S^+(F_I) = \sqrt{\sum_{i=1}^{n}\left(s_i(F_I) - s_i^+\right)^2}$$

n is the number of observations.

This measure allows the occurrence of observations which are larger than the maximum symptoms defined for any fault. This provides some flexibility in modeling the faults and acknowledges that there may be modeling errors. Some observations may be larger than symptoms originally predicted but they should still be assigned to a fault. This situation is depicted in Fig. 2.1.1-2. Here two faults are modeled where fault 1 causes both symptoms that fault 2 causes but to a smaller degree. Although the measurements are larger than the two symptoms for fault 1, the closeness measure will still assign fault 1 a higher numerical value than fault 2 because the measurement is closer to fault 1 than to fault 2.

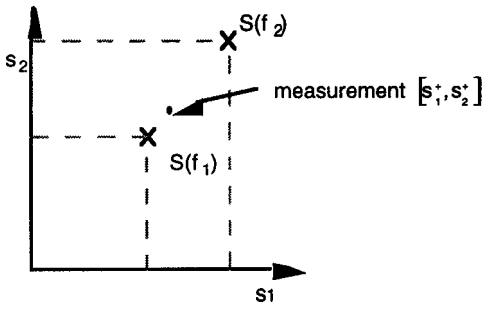


Fig. 2.1.1-2: Measurement larger than symptoms modeled

Measure $\gamma_c\left(S(F_I), S^+\right)$ shares an important property for the assignment of truth with abduction as displayed for crisp cases in Table 2.1.1-1. Note the difference to implication which is only false where the antecedent is true and the consequence is false. The closeness measure is also not true when the antecedent is false and the consequent is true, an important property in abductive reasoning.

$S(F_I)$	S^+	$S(F_I) \rightarrow S^+$	$\gamma_c\left(S(F_I), S^+\right)$
0	0	1	1
0	1	1	0
1	0	0	0
1	1	1	1

Table 2.1.1-1: Truth table for implication and closeness measure

An important concept introduced through allowing symptoms to occur to some degree is the notion of a distribution for the failure. This distribution should have $\gamma_c = 1$ at the modeled fault and be smaller further away. It should also be $\gamma_c = 0$ when no symptom is observed. From the truth table we have already seen that this is the case for the crisp failures at the values one and zero. The true power however comes to light when looking at the symptoms in the fault-symptom space. To begin, a one-dimensional case is shown in Fig. **2.1.1-3.** The fault is modeled at s=0.7. As can be seen, the diagnosis is $\gamma_c = 0$ where no symptom is observed and it rises to $\gamma_c = 1$ at the modeled fault, after which it decreases again.

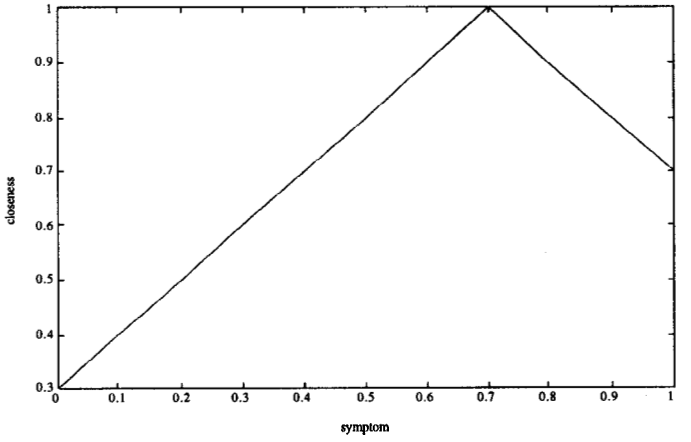


Fig. 2.1.1-3: Distribution for crisp failure for 1-dimensional symptoms

Fig. 2.1.1-4 shows a similar situation for 2-dimensional symptoms. The failure was modeled at $s = \begin{bmatrix} 0.7 \\ 0.7 \end{bmatrix}$. At the origin, i.e., where no symptom is observed at all, the diagnosis is $\gamma_c = 0$. Further away, the diagnosis for the failure increases and has its peak of $\gamma_c = 1$ at the modeled failure. Fig. 2.1.1.1-4 shows how the diagnosis for the failure is distributed around the modeled failure with observed symptoms 1 and 2. Also shown are the contours under the mesh which are radial, centered at the modeled failure.

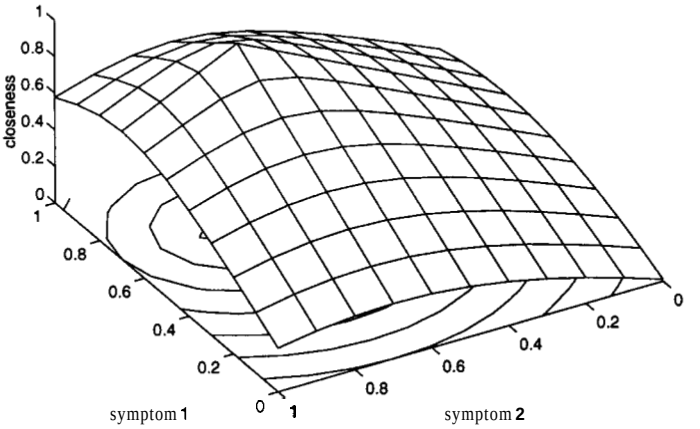


Fig. 2.1.1-4: Distribution for crisp failure for 2-dimensional symptoms

The measure obtained through $\gamma_c\left(S(F_1), S^+\right)$ is a "dissemblance" measure and meets requirements of symmetry and anti-reflexivity, but not of co-transitivity (Kaufmann, 1971) as outlined below.

Symmetry:

$$\begin{aligned}\gamma_c(x,y) &= 1 - \min\left(1, 1-\mu_x+\mu_y, 1+\mu_x-\mu_y\right) \\ &= 1 - \min\left(1, 1-\mu_y+\mu_x, 1+\mu_y-\mu_x\right) \\ &= \gamma_c(y,x)\end{aligned}$$

Anti-Reflexivity:

$$\gamma_c(x,x) = 1 - \min(1,1,1) = 0$$

Co-Transitivity:

$$\gamma_c(x,z) = 1 - \min\left(1, 1-\mu_x+\mu_z, 1+\mu_x-\mu_z\right)$$

$$\gamma_c(x,y) = 1 - \min\left(1, 1-\mu_x+\mu_y, 1+\mu_x-\mu_y\right)$$

$$\gamma_c(y,z) = 1 - \min\left(1, 1-\mu_y+\mu_z, 1+\mu_y-\mu_z\right)$$

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=\min\left(1-\min\left(1,1-\mu_x+\mu_y,1+\mu_x-\mu_y\right),1-\min\left(1,1-\mu_y+\mu_z,1+\mu_y-\mu_z\right)\right)$$
$$=\max\left(\min\left(1,1-\mu_x+\mu_y,1+\mu_x-\mu_y\right),\min\left(1,1-\mu_y+\mu_z,1+\mu_y-\mu_z\right)\right)$$
$$=1-\min\left(\min\left(1,1-\mu_x+\mu_y,1+\mu_x-\mu_y\right),\min\left(1,1-\mu_y+\mu_z,1+\mu_y-\mu_z\right)\right)$$
$$=1-\min\left(1,1-\mu_x+\mu_y,1+\mu_x-\mu_y,1-\mu_y+\mu_z,1+\mu_y-\mu_z\right)$$

Case 1:

$\mu_x=\mu_y=\mu_z$:

$\gamma_c(x,y)\wedge\gamma\ (y,z)=1-\min(\min(1,1,1),\min(1,1,1))=0$

$\gamma_c(x,z)=1-\min(1,1,1)=0$

$\Rightarrow\gamma_c(x,z)=\gamma_c(x,y)\wedge\gamma_c(y,z)$

Case 2:

$\mu_x=\mu_y$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min\left(1,1,1,1+\mu_x-\mu_z,1-\mu_x+\mu_z\right)$$
$$=1-\min\left(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z\right)$$
$$=\gamma(x,z)$$

Case 3:

$\mu_x=\mu_z$:

$\gamma_c(x,z)=0$

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min\left(1,1+\mu_z-\mu_y,1-\mu_z+\mu_y,1+\mu_y-\mu_z,1-\mu_y+\mu_z\right)$$
$$=1-\min\left(1,1+\mu_y-\mu_z,1-\mu_y+\mu_z\right)$$

$\Rightarrow\gamma_c(x,z)\le\gamma_c(x,y)\wedge\gamma_c(y,z)$

Case 4:

$\mu_y=\mu_z$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=\min\left(1,1-\left(\mu_y-\mu_y\right),1-\left(\mu_y-\mu_y\right),1-\left(\mu_x-\mu_z\right),1-\left(\mu_x-\mu_z\right)\right)$$
$$=1-\min\left(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z,1+\mu_z-\mu_z,1-\mu_z+\mu_z\right)$$
$$=1-\min\left(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z\right)$$
$$=\gamma_c(x,z)$$

Case 5:

$\mu_x>\mu_y,\mu_z>\mu_y$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min\left(1,1+\mu_x-\mu_y,1-\mu_x+\mu_y,1+\mu_y-\mu_z,1-\mu_y+\mu_z\right)$$
$$=1-\min\left(1,1-\mu_x+\mu_y,1+\mu_y-\mu_z\right)$$
$$=1-\min\left(1-\mu_x+\mu_y,1+\mu_y-\mu_z\right)$$

$$\gamma_c(x,z)=1-\min\left(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z\right)$$
$$=1-\min\left(1+\mu_x-\mu_z,1-\mu_x+\mu_z\right)$$

but $\mu_x>\mu_y$ & $\mu_z>\mu_y$

$$\Rightarrow 1-\mu_x+\mu_y<1-\mu_x+\mu_z$$
$$1+\mu_y-\mu_z<1+\mu_x-\mu_z$$
$$\Rightarrow \gamma_c(x,z)<\gamma_c(x,y)\wedge\gamma_c(y,z)$$

Case 6:

$\mu_y>\mu_x,\ \mu_y>\mu_z$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min(1,1+\mu_x-\mu_y,1-\mu_x+\mu_y,1+\mu_y-\mu_z,1-\mu_y+\mu_z)$$
$$=1-\min(1,1+\mu_x-\mu_y\ 1+\mu_z-\mu_y)$$
$$\gamma_c(x,z)=\min(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z)$$

but $\mu_y>\mu_x$ & $\mu_y>\mu_z$

$$\Rightarrow 1+\mu_x-\mu_z>1+\mu_x-\mu_y$$
$$1+\mu_z-\mu_x>1+\mu_z-\mu_y$$
$$\Rightarrow \gamma_c(x,z)>\gamma_c(x,y)\wedge\gamma_c(y,z)$$

Case 7:

$\mu_x>\mu_y>\mu_z$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min(1,1+\mu_x-\mu_y,1-\mu_x+\mu_y,1+\mu_y-\mu_z,1-\mu_y+\mu_z)$$
$$=1-\min(1,1-\mu_x+\mu_y,1-\mu_y+\mu_z)$$
$$=1-\min(1-\mu_x+\mu_y,1-\mu_y+\mu_z)$$
$$\gamma_c(x,z)=\min(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z)$$
$$=1-\min(1-\mu_x+\mu_z$$

but $\mu_x>\mu_y>\mu_z$

$$\Rightarrow 1-\mu_x+\mu_z<1-\mu_x+\mu_y$$
$$1-\mu_x+\mu_z<1-\mu_y+\mu_z$$
$$\Rightarrow \gamma_c(x,z)>\gamma_c(x,y)\wedge\gamma_c(x,z)$$

Case 8:

$\mu_y>\mu_x,\ \mu_y>\mu_z$:

$$\gamma_c(x,y)\wedge\gamma_c(y,z)=1-\min(1,1+\mu_x-\mu_y,1-\mu_x+\mu_y,1+\mu_y-\mu_z,1-\mu_y+\mu_z)$$
$$=1-\min(1,1+\mu_x-\mu_y,1-\mu_y+\mu_z)$$
$$=1-\min(1+\mu_x-\mu_y,1-\mu_y+\mu_z)$$
$$\gamma_c(x,z)=\min(1,1+\mu_x-\mu_z,1-\mu_x+\mu_z)$$
$$=\min(1+\mu_x-\mu_z,1-\mu_x+\mu_z)$$

but $\mu_y>\mu_x$ & $\mu_y>\mu_z$

$$\Rightarrow 1+\mu_x-\mu_y<1+\mu_x-\mu_z$$
$$1+\mu_z-\mu_y<1+\mu_z-\mu_x$$
$$\Rightarrow \gamma_c(x,z)<\gamma_c(x,y)\wedge\gamma_c(y,z)$$

The above results (cases 5 and 8) show that the measure is not co-transitive.

2.1.2 Gradual Failures

For gradual failures, a means is provided which takes into account the distance of the observation to the closest symptom set for a fault as well as to what degree the fault may occur. This necessitates the measurement of two quantities: one is the closest distance from the observation to the fault line. The fault line denotes the line on which all gradual instances of one particular fault are assumed to lie. It starts at the origin and is monotonically increasing. The other quantity involves the measurement of the intersection of the closest distance to the observation on the failure line (S_R^+) to the point where no fault occurs. This situation is depicted in Fig. 2.1.2-1.

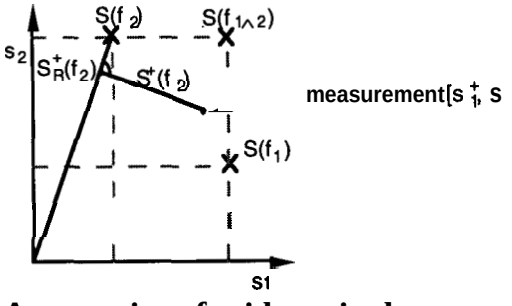


Fig. 2.1.2-1: Aggregation of evidence in the symptom-failure space for fuzzy faults and symptoms

The resulting measure is of the form

$$\gamma_s\left(S(F_I), S^+\right) = 1 - \min\left(1, 1 - S_n(F_I) + S^+(F_I), 1 + S_n(F_I) - S^+(F_I)\right)$$

where

$S(F_I)$ is the symptom set for failure F,

S^+ are the observed symptoms

$S(F_I) > S_R^+(F_I)$:

$$S_n^+(F_I) = \frac{l^2\left(S_R^+(F_I), \varnothing\right)}{l^2\left(S(F_I), \varnothing\right)} = \frac{\sqrt{\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - \varnothing\right)^2}}{\sqrt{\sum_{i=1}^{n}\left(s_i(F_I) - \varnothing\right)^2}} = \frac{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}}$$

$S(F_I)\ c\ S_R^+(F_I)$:

$$S_n^+(F_I) = \frac{l^2(S(F_I),\varnothing)}{l^2(S_R^+(F_I),\varnothing)} - \frac{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}}$$

$S_n^+(F_I)$ is the degree to which the failure occurred expressed by the length of the failure line to the intersection with the closest distance to the measurement, normed by the overall length of the failure.

$$S^+(F_I) = l^2(S_R^+(F_I), S^+)$$

$S^+(F_I)$ is the distance to the failure line.

$$= \left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - s_i^+\right)^2\right)^{\frac{1}{2}}$$

$S(F_I) > S_R^+(F_I)$:

$$\gamma_s\left(S(F_I), S^+\right) = 1 - \min\left(1, 1 - \frac{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}} + \left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - s_i^+\right)^2\right)^{\frac{1}{2}}, 1 + \frac{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}} - \left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - s_i^+\right)^2\right)^{\frac{1}{2}}\right)$$

$S(F_I) < S_R^+(F_I)$:

$$\gamma_s\left(S(F_I), S^+\right) = 1 - \min\left(1, 1 - \frac{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}} + \left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - s_i^+\right)^2\right)^{\frac{1}{2}}, 1 + \frac{\left(\sum_{i=1}^{n}\left(s_i(F_I)\right)^2\right)^{\frac{1}{2}}}{\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)\right)^2\right)^{\frac{1}{2}}} - \left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I) - s_i^+\right)^2\right)^{\frac{1}{2}}\right)$$

The distance is calculated using some vector algebra. Known quantities include the measurement, here denoted as point S^+, and the symptom vector in the vector space, denoted temporarily as $\vec{a} + t \cdot \vec{s}(F_I)$. The task is to find the shortest distance from S^+ to the line $\vec{a} + t \cdot \vec{s}(F_I)$. For this

purpose, first the intersection **S,** of the shortest line with $\vec{a} + t \cdot \vec{s}(F_I)$ will be found. $\vec{s}_R(F,)$ can be expressed as

$$\vec{s}_R(F_I) = \vec{a} + t_0 \cdot \vec{s}(F_I). \qquad \text{Equation (2.1.2-1)}$$

Because $\vec{s}_R(F,) - \vec{s}^+$ and $\vec{s}(F_I)$ are perpendicular, the scalar product

$$\left(\vec{s}_R(F_I) - \vec{s}^+\right) \cdot \vec{s}(F_I) = \left|\left(\vec{s}_R(F_I) - \vec{s}^+\right)\right| \cdot \left|\vec{s}(F_I)\right| \cdot \cos 90° = 0$$
$$\Rightarrow \left(\vec{s}_R(F_I) - \vec{s}^+\right) \cdot \vec{s}(F_I) = 0$$
$$\Rightarrow \vec{s}_R(F_I) \cdot \vec{s}(F_I) - \vec{s}^+ \cdot \vec{s}(F_I) = 0$$
$$\Rightarrow \vec{s}_R(F_I) \cdot \vec{s}(F_I) = \vec{s}^+ \cdot \vec{s}(F_I). \qquad \text{Equation (2.1.2-2)}$$

Multiplying Equation (2.1.2-1) with $\vec{s}(F_I)$

$$\vec{s}_R(F_I) \cdot \vec{s}(F_I) = \vec{a} \cdot \vec{s}(F_I) + t_0 \cdot \vec{s}(F_I)^2. \qquad \text{Equation (2.1.2-3)}$$

Furthermore, from Equation (2.1.2-2) and (2.1.2-3)

$$\vec{s}^+ \cdot \vec{s}(F_I) = \vec{a} \cdot \vec{s}(F_I) + t_0 \cdot \vec{s}(F_I)^2$$
$$\Rightarrow t_0 = \frac{\left(\vec{s}^+ - \vec{a}\right) \cdot \vec{s}(F_I)}{\left|\vec{s}(F_I)\right|^2} \qquad \text{Equation (2.1.2-4)}$$

but if

$$\vec{a} = \vec{O} \qquad \text{Equation (2.1.2-5)}$$

with Equations (2.1.2-4) and (2.1.2-5)

$$t_0 = \frac{\vec{s}^+ \cdot \vec{s}(F_I)}{\left|\vec{s}(F_I)\right|^2} = I^2\left(S_R^+(F_I), S^+\right)$$
$$= \frac{\vec{s}_1^+ \cdot \vec{s}_1(F_I) + \vec{s}_2^+ \cdot \vec{s}_2(F_I) + \ldots + \vec{s}_n^+ \cdot \vec{s}_n(F_I)}{\vec{s}_1(F_I) \cdot \vec{s}_1(F_I) + \vec{s}_2(F_I) \cdot \vec{s}_2(F_I) + \ldots + \vec{s}_n(F_I) \cdot \vec{s}_n(F_I)}. \qquad \text{Equation (2.1.2-6)}$$

Equation (2.1.2-6) and (2.1.2-1) result in S_R

$$S_R = \frac{\left(\vec{s}^+ - \vec{a}\right) \cdot \vec{s}(F_I)}{\left|\vec{s}(F_I)\right|^2} \cdot \vec{s}(F_I). \qquad \text{Equation (2.1.2-7)}$$

The shortest distance is then calculated as

$$S^+(F_I) = \left|\overline{S_R S^+}\right| = \sqrt{\left(s_1^+ - s_{R_1}\right)^2 + \left(s_2^+ - s_{R_2}\right)^2 + \ldots + \left(s_n^+ - s_{R_n}\right)^2}. \qquad \text{Equation (2.1.2-8)}$$

This measure introduced is similar to the dissemblance measure as well. It satisfies the requirements of symmetry and anti-reflexivity but not of co-transitivity. We refer to this measure as a measure of closeness.

The truth assignment for the soft failures is a little more complex than for the crisp case. This results mainly from the fact that the term $S_n(F_I)$ in the closeness measure is the ratio of the soft failure to the crisp failure. If both symptom and fault are approaching zero, the truth still depends on the locations of observation and failure relative to each other. Therefore, the assignment of truth to this case will be somewhere between zero and one. It is thus desirable to model failures with distinctive symptoms to avoid ambiguous results. All other cases behave much like the truth assignments for abductive reasoning. The results are summarized in Table 2.1.2-1 which also shows the implication operator which is used with a threshold value.

$S(F_I)$	S^+	$S(F_I) \rightarrow S^+$	$\gamma_s(S(F_I), S^+)$
$\lim_{S(F_I)\rightarrow 0}$	$\lim_{S^+\rightarrow 0}$	1	$0<g_S<1$
$\lim_{S(F_I)\rightarrow 0}$	$\lim_{S^+\rightarrow 1}$	1	$\gamma_s \rightarrow 0$
$\lim_{S(F_I)\rightarrow 1}$	$\lim_{S^+\rightarrow 0}$	0	$\gamma_s \rightarrow 0$
$\lim_{S(F_I)\rightarrow 1}$	$\lim_{S^+\rightarrow 1}$	1	$\gamma_s \rightarrow 1$

Table 2.1.2-1: Truth table for implication and closeness measure

The distribution around the modeled failure exists for the soft failure as well. In contrast to the crisp case, however, the model extends to the entire range for a failure between zero and one. For ease of computation, this model can be assumed to be a straight line, although it could be of any other shape as well. In the latter case, the computation for distance would have to be adjusted accordingly. Fig. 2.1.2-2 shows the closeness measure for the soft failure for 1-dimensional symptoms and straight failure line. As before, the fault was assumed to be at s=0.7. The diagnosis for the failure is zero where no symptom is observed and increases to one at the modeled fault, after which it decreases again.

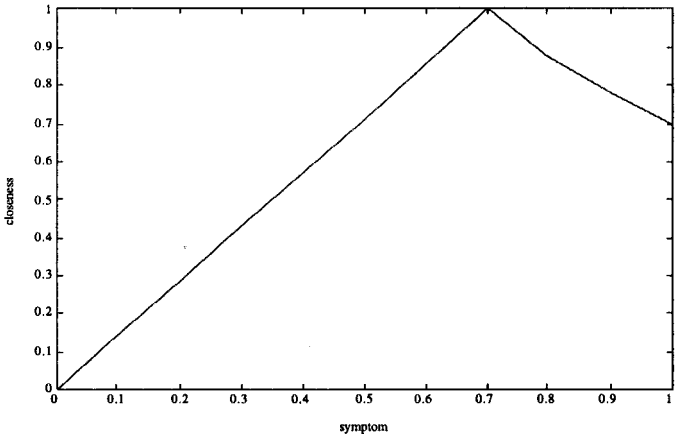


Fig. 2.1.2-2: Distribution for soft failure for 1-dimensional symptoms

Fig. 2.1.2-3 shows a similar situation for soft failures, however for 2-dimensional symptoms. The fault was modeled at $s = \begin{bmatrix} 0.7 \\ 0.7 \end{bmatrix}$. As can be seen, the diagnosis is zero for the case where no symptoms are observed and increases to one at the modeled failure, after which it decreases again. From the mesh displayed and the contours under the mesh, it can be seen that the closeness measure for soft failures radiates in a more rectangular fashion. In contrast to the crisp failures, the

soft failure has a different degree of failure at each point in the space. The degree to which the failure is diagnosed is displayed in Fig. 2.1.2-4. Because the shortest distance to the line between the origin and the crisp failure is used as the model, the degree is the same in the perpendicular direction of the model.

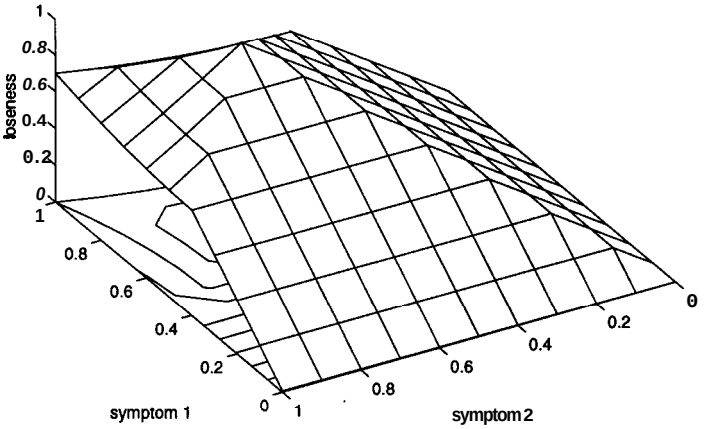


Fig. 2.1.2-3: Distribution for soft failure for 2-dimensional symptoms for g_S

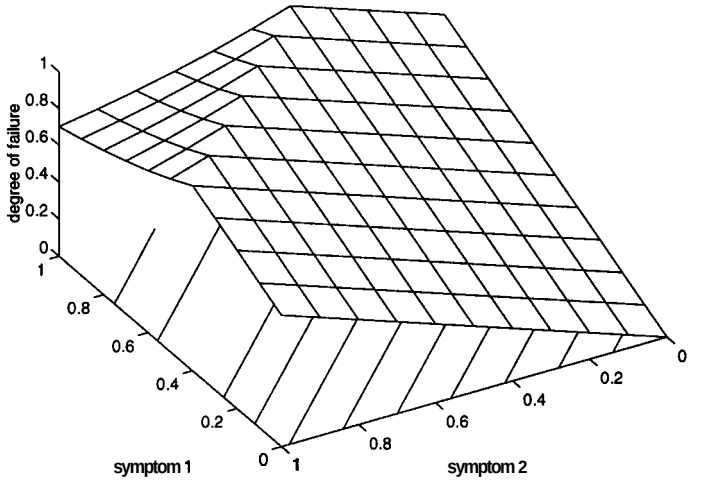


Fig. 2.1.2-4: Degree to which failure is diagnosed for 2-dimensional symptoms for g_S

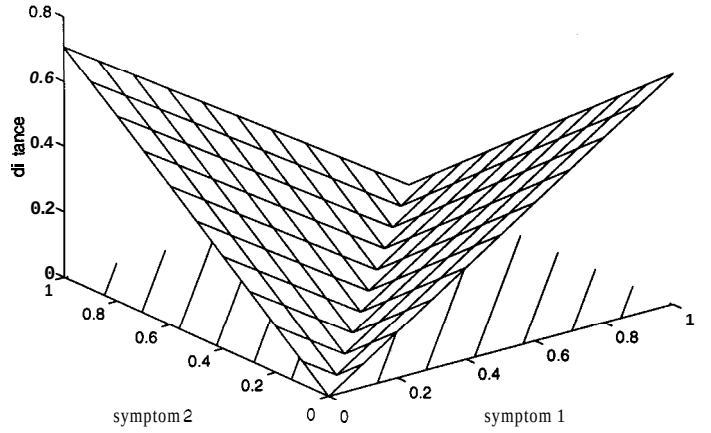


Fig. 2.1.2-5: Distance from observation to failure line for 2-dimensional symptoms for g_S

The closeness measure g_S is a product of the degree to which the failure is diagnosed and the distance of the observation to the failure line. Fig. 2.1.2-4 shows the degree to which the failure is diagnosed with g_S and Fig. 2.1.2-5 shows the distance of the observation to the failure line. For some applications it may be undesirable to have the distribution of the diagnosis as displayed in Fig. 2.1.2-3. Rather, a distribution around the line which decreases with increasing distance from that line seems more practical. The problem lies in the parameter which is the normed distance to the overall distance. **A** more desirable distribution can be found in the diagnosis of the hard failure which was displayed in Fig. 2.1.2-5. Replacing the normed distance from the intersection of the shortest distance with the failure line to the origin with the distance to the crisp failure renders the following relation

$$\gamma_s^*\left(S(F_I),S^+\right)=1-\min\left(1,1-S_d(F_I)+S^+(F_I)\right)$$

where

$S_d(F_I)=\sqrt{\sum_{i=1}^{n}\left(s_i(F_I)-s_i^+\right)^2}$ is the distance to symptom set for failure F_I

S^+ are the observed symptoms

$$S^+(F_I)=l^2\left(S_R^+(F_I),S^+\right)$$

$S^+(F_I)$ is the distance to the failure line for failure combination FI.

$$=\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F,)-s_i^+\right)^2\right)^{\frac{1}{2}}$$

$$\gamma_s^*\left(S(F_I),S^+\right)=1-\min\left(1,1-\left(\sum_{i=1}^{n}\left(s_i(F_I)-s_i^+\right)^2\right)^{\frac{1}{2}}+\left(\sum_{i=1}^{n}\left(s_{R_i}^+(F_I)-s_i^+\right)^2\right)^{\frac{1}{2}}\right).$$

Fig. 2.1.2-6 shows which components are used for the aggregation of evidence for γ_s^*. The two quantities of interest are **S,** and S^+.

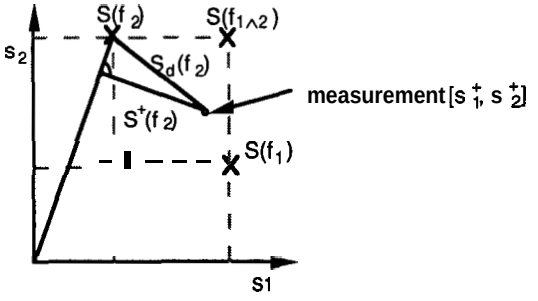


Fig. 2.1.2-6: Aggregation of evidence in the symptom-failure space for fuzzy faults and symptoms using γ_s^*

Fig. 2.1.2-7 shows the diagnosis with this modified closeness measure. It now has a much smoother distribution and decreases with increasing distance from the modeled failure line. The contours are radially decreasing with distance from the modeled failure.

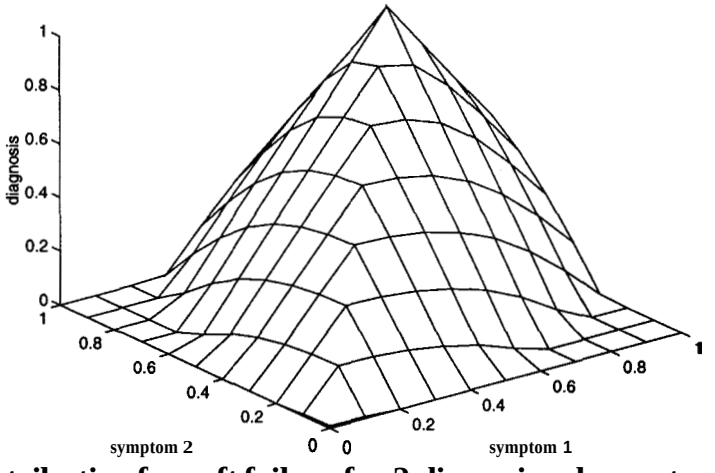


Fig. 2.1.2-7: Distribution for soft failure for 2-dimensional symptoms with closeness measure

Note that γ_s^* is not a dissemblance measure because it does not meet symmetry requirements. $\gamma_s^*\left(S_d(F_I), S^+\right)$ behaves favorably with the desired abductive truth assignments because it avoids the ratio problems which the measure $\gamma_s\left(S(F_I), S^+\right)$ exhibited. Measure $\gamma_s^*\left(S_d(F_I), S^+\right)$ correctly assigns the degree of truth of $\gamma_s^* = 1$ where the antecedent and consequent coincide and it assigns $\gamma_s^* = 0$ otherwise. Table 2.1.2-2 summarizes these results.

$S(F_I)$	S^+	$S(F_I) \rightarrow S^+$	$\gamma_s^*\left(S_d(F_I), S^+\right)$
0	0	1	1
0	1	1	0
1	0	0	0
1	1	1	1

Table 2.1.2-2: Truth table for implication and closeness measure

2.1.3 Example

Some simple cases will be used to further illustrate how the algorithm works. The assumptions are that the failure line is straight and that the @-faultis at the origin.

Because symptoms are allowed to occur to some degree (as opposed to crisply only), the chosen approach needs to calculate the fuzzy closeness measure for all fault combinations and rank them afterwards. Assuming crisp cases as displayed in Fig. 2.1.1-1 and

$$S(f_1[1]) = \begin{bmatrix} 0.8 \\ 0.4 \end{bmatrix}$$

$$S(f_2[1]) = \begin{bmatrix} 0.3 \\ 0.9 \end{bmatrix}$$

$$S(f_{1\wedge 2}[1]) = \begin{bmatrix} 0.8 \\ 0.9 \end{bmatrix}$$

$$S(f_0[1]) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

where

$S(F_0[1])$ denotes the @-faultcase.

The measurement is assumed to be

$$S^+ = \begin{bmatrix} 0.7 \\ 0.6 \end{bmatrix}$$

The distance to the crisp faults is then calculated as

$$S^+(f_1) = 0.22$$
$$S^+(f_2) = 0.50$$
$$S^+(f_{1\wedge 2}) = 0.32$$
$$S^+(f_0) = 0.92.$$

The closeness measure is calculated as

$$\gamma_c(S(f_1), S^+) = 0.78$$
$$\gamma_c(S(f_2), S^+) = 0.5$$
$$\gamma_c(S(f_{1\wedge 2}), S^+) = 0.68$$
$$\gamma_c(S(f_0), S^+) = 0.08.$$

Accordingly, fault f_1 would be chosen because it has the highest closeness which in this case is entirely determined by the distance of the observation to the symptom for the failure.

Assuming soft failures similar to the scenario displayed in Fig. 2.1.2-1, use the same symptom set

$$S(f_1[1]) = \begin{bmatrix} 0.8 \\ 0.4 \end{bmatrix}$$

$$S(f_2[1]) = \begin{bmatrix} 0.3 \\ 0.9 \end{bmatrix}$$

$$S(f_{1\wedge 2}[1]) = \begin{bmatrix} 0.8 \\ 0.9 \end{bmatrix}$$

$$S(f_0[1]) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

and the measurement

$$S^+ = \begin{bmatrix} 0.7 \\ 0.6 \end{bmatrix}.$$

The shortest distances from S^+ are calculated **as**

$$S^+(f_1) = 0.22$$
$$S^+(f_2) = 0.47$$
$$S^+(f_{1\wedge 2}) = 0.12$$
$$S^+(f_0) = 0.92$$

with

$$S_n(f_1) = 1$$
$$S_n(f_2) = 0.83$$
$$S_n(f_{1\wedge 2}) = 0.76$$
$$S_n(f_0) = 1.$$

The closeness measures are calculated **as**

$$\gamma_s(S(F_1), S^+) = 0.78$$
$$\gamma_s(S(F_2), S^+) = 0.36$$
$$\gamma_s(S(F_{1\wedge 2}), S^+) = 0.63$$
$$\gamma_s(S(F_0), S^+) = 0.08.$$

In this case failure f_1 would have been chosen as well because it has the highest closeness measure. It no longer represents just a measure of the distance from the observation to the symptom. Rather, the failure itself has some weight and is therefore considered. Failure f_1 is diagnosed to occur to degree "1" here. The closeness measure to the @-"fault", i.e., the case when no fault occurs undergoes special treatment. By default, the strength of the fault is set to "1" while the strength of the symptom is computed as for all other faults. In other words, the @-fault is considered as a hard case and the equivalence measure is determined alone by the distance of the observation to the symptom (which is 0 **as** well).

Next, the limit cases are investigated. When the observation approaches Ø, i.e., $\lim_{s\to 0} S^+$, the distance from the observation to the origin is very small and the closeness measure tends to 1 for the Ø-fault case while all the other cases approach 0. This means that the case "no faults" will be diagnosed correctly. When the observation is equal to a fault, then this fault will be diagnosed with the highest possible measure of 1. If the observation is exactly between two faults, say

$\mathbf{S'} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$, with exactly opposing symptoms $S(f_1[1]) = \begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}$ and $S(f_2[1]) = \begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}$ then the closeness measure for both faults is naturally the same. However, the closeness measure for the combined fault $S(F_{1\wedge 2}[1]) = \begin{bmatrix} 0.8 \\ 0.8 \end{bmatrix}$ is larger which indicates the strong possibility of a combined fault which also makes sense.

In the case of the observation exactly half way between a fault and the origin, then the closeness measure for that fault is 0.5 because the distance from the observation to the failure line is zero and only the ratio of the location of the observation to the length of the symptom determines the closeness measure. Recall that 0.5 is interpreted as "do not know" on the scale from 0 to 1. The Ø-fault is diagnosed slightly more likely in that case because only the length of the observation is of relevance. **A** circle around the origin with radius 0.5 determines where the @-faulthas value 0.5. For a fault with crisp symptoms and observations half way between the origin and the symptom set or that fault, the fault and the Ø-fault are equally likely. For all other faults and observations in between, the system will favor the @-fault somewhat. This may be undesired for some applications. These results for g_S are summarized in Table 2.1.3-1.

$S(F_1[1])$	$S(F_2[1])$	$S(F_{1\wedge 2}[1])$	S^+	$\gamma_s(S(F_1),S^+)$	$\gamma_s(S(F_2),S^+)$	$\gamma_s(S(F_{1\wedge 2}),S^+)$	$\gamma_s(S(F_0),S^+)$
$\begin{bmatrix} 0.8 \\ 0.4 \end{bmatrix}$	$\begin{bmatrix} 0.3 \\ 0.9 \end{bmatrix}$	$\begin{bmatrix} 0.8 \\ 0.9 \end{bmatrix}$	$\begin{bmatrix} 0.7 \\ 0.6 \end{bmatrix}$	0.776	0.359	0.634	0.078
- // -	- // -	- // -	$\begin{bmatrix} 0 \\ 0 \end{bmatrix}$	0	0	0	1
- // -	- // -	- // -	$\begin{bmatrix} 0.8 \\ 0.4 \end{bmatrix}$	1	0.034	0.358	0.106
$\begin{bmatrix} 0.8 \\ 0.2 \end{bmatrix}$	$\begin{bmatrix} 0.2 \\ 0.8 \end{bmatrix}$	$\begin{bmatrix} 0.8 \\ 0.8 \end{bmatrix}$	$\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$	0.526	0.371	0.625	0.293
$\begin{bmatrix} 1 \\ 0 \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \end{bmatrix}$	$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$	-//-	0	0	0	0.293

Table 2.1.3-1: Summary of results for diagnosis using g_S

The same tests were carried out for γ_s^*. The results can be seen in Table 2.1.3-2.

$S(F_1[1])$	$S(F_2[1])$	$S(F_{1\wedge 2}[1])$	S^+	$\gamma_s^*(S_d(F_1),S^+)$	$\gamma_s^*(S_d(F_2),S^+)$	$\gamma_s^*(S_d(F_{1\wedge 2}),S^+)$
$\begin{bmatrix}0.8\\0.4\end{bmatrix}$	$\begin{bmatrix}0.3\\0.9\end{bmatrix}$	$\begin{bmatrix}0.8\\0.9\end{bmatrix}$	$\begin{bmatrix}0.7\\0.6\end{bmatrix}$	0.553	0.026	0.559
- // -	- // -	- // -	$\begin{bmatrix}0\\0\end{bmatrix}$	0.106	0.051	0.209
- // -	- // -	- // -	$\begin{bmatrix}0.8\\0.4\end{bmatrix}$	0.212	0	0.168
$\begin{bmatrix}0.8\\0.2\end{bmatrix}$	$\begin{bmatrix}0.2\\0.8\end{bmatrix}$	$\begin{bmatrix}0.8\\0.8\end{bmatrix}$	$\begin{bmatrix}0.5\\0.5\end{bmatrix}$		0.212	0.576
$\begin{bmatrix}1\\0\end{bmatrix}$	$\begin{bmatrix}0\\1\end{bmatrix}$	$\begin{bmatrix}1\\1\end{bmatrix}$	-//-	0	0	0

Table 2.1.3-2: Summary of results for diagnosis using γ_s^*

The following graphs (Figs. 2.1.3-1, 2.1.3-2, and 2.1.3-3) show the maximum fault profile for three faults (excluding the Ø-fault) in a two-dimensional symptom space. Although faults are usually modeled with more distinctive symptoms, the graphs give a good idea of how the diagnosis operates. The faults were modeled at $s_1 = \begin{bmatrix}0.8\\0.2\end{bmatrix}$, $s_2 = \begin{bmatrix}0.3\\0.7\end{bmatrix}$, and $s_3 = \begin{bmatrix}0.9\\0.9\end{bmatrix}$. Fig. 2.1.3-1 shows the maximum fault profile for crisp g_C. The maximum failure surface is smooth and faults are seen to be centered around their modeled place in space. The valleys between the maxima show where the fault would be diagnosed equally likely for either of two faults. Fig. 2.1.3-2 shows the maximum failure profile for soft g_S. The contours are closer indicating steeper gradients on the surface. Finally, Fig. 2.1.3-3, the plot for γ_s^*, has the steepest gradients, which allows better classification but makes the diagnosis also more susceptible to modeling errors.

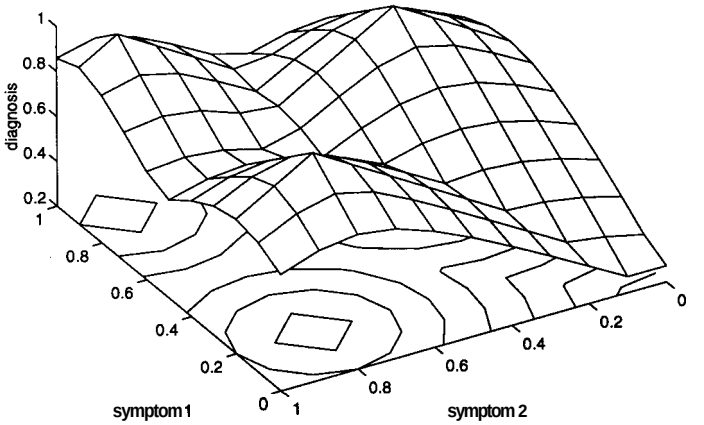


Fig. 2.1.3-1: Modeling of three faults with two symptoms using crisp g_C

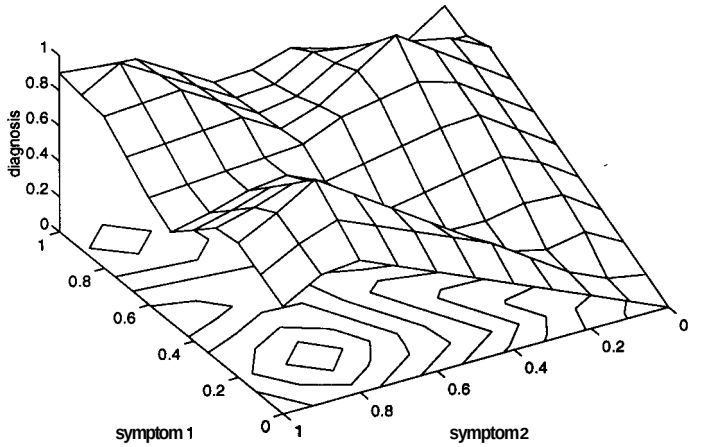


Fig. 2.1.3-2: Modeling of three faults with two symptoms using g_S

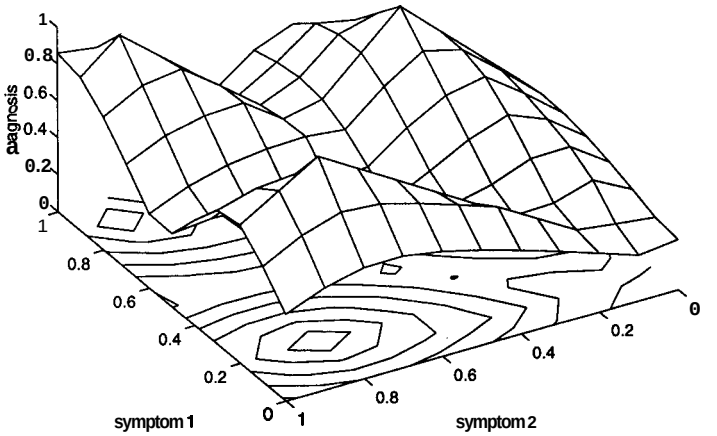


Fig. 2.1.3-3: Modeling of three faults with two symptoms using γ_s^*

2.1.4 Summary and Conclusions

The approach proposed in this chapter offers a solution which allows for the diagnosis of crisp and partial failures using closeness measures γ_c, y,, and γ_s^*. The calculation of the closeness measure uses distance measures from the observed symptom set to the symptom set for a failure combination and – in the case of soft failures – distance measures to the failure line. Two closeness measures are provided for soft failures which allows to determine to which degree a failure occurs. The first calculates the distance to the failure line **as** a measure for the degree of the symptom and the distance from origin to intersection with shortest distance, normed by the overall length of the failure line, as a measure for the strength of the failure. The second closeness measure, γ_s^*, also takes the distance from observed symptom to failure line **as** the measure for the strength of the symptom but uses the distance from observed symptom to symptom set as the measure for the strength of the rule.

The degree to which the failure occurs is given by the distance from the origin to the intersection point with the shortest distance line. Advantages of this approach are that solutions are always given, overcoming shortcomings of previous fuzzy approaches. Diagnosis is achieved by monitoring symptoms and evaluating to which degree they occur. Insufficiencies of threshold driven diagnosis are eliminated because the approach avoids assumptions of failure independence and of relative frequency of disorder occurrence. Links in the causal network are represented as causal strengths for failure-symptom relations similar to the Bayesian approach.

This approach is, in a computational sense, not very expensive. Additional expert knowledge about the behavior of multiple fault-symptom relations can be incorporated into the system model which may result in the placement of combined faults at locations other than the maxima of their symptoms. This would occur when the failures (partially) cancel their symptoms or when the straight line model for the failure behavior is known to be incorrect.

2.2 Fuzzy Diagnosis in IVHS

This section shows how fuzzy diagnosis is used in the context of sensor and actuator malfunctions in automated vehicles of the Intelligent Vehicle Highway System (IVHS). Sensor validation and sensor fusion provide means to remedy some of the detrimental effects of uncertain readings as shown in Goebel (1996). Noise and aberrant readings can be filtered out to some degree, using information from partially redundant sensors, sensor characteristics, and expected readings (Goebel and Agogino, 1996). In either case it is important to know how the sensors and actuators behave to allow monitoring of their performance and perform diagnosis to permit remedial action in case of malfunction. This is an important task in safety preservation of the IVHS system.

Since platoons travel largely independently, failures of components within individual vehicles must be dealt with by considering the effects of remedial maneuvers on the other members in the platoon. Ideally, a failure can be accommodated within the vehicle itself without disturbing the operation of the platoon. In either case, diagnosis lies at the heart of establishing a safe system. Only when a problem can be observed and identified it can be dealt with. It is equally important to trace the source of a failure to avoid the creation of other faults or reoccurrence of the same fault.

Diagnosis of sensors and actuators may also enable prediction of failure when failure modes are known. In this case, an emergency maneuver may be avoided because more time allows for preparation to switch to the degraded mode of operation (Lygeros et al., 1995) in which indirect systems are used for control, if possible. Moreover, it is critical to determine the degree to which a failure has occurred. If, for example, an actuator exhibits a malfunction such as a bias, it is

important to know the magnitude of the malfunction. This will allow for better compensation to the problem.

The theoretical groundwork for diagnosis was laid in Section 2.1. The first step in applying the method described in Section 2.1 to **IVHS,** faults and symptoms are assigned real meaning, e.g., for faults f_n and symptoms s_m:

f_1	quantization failure
f_2	out of range failure
f_3	obstruction of sensor
f_4	fog/dirt
s_1	reading in quantization range
s_2	difference between estimate and measurement large
s_3	change of measurements large
s_4	previous measurement is an outlier
s_5	measurement same order of magnitude as previous one
s_6	measurements have large variance
s_7	measurement shows large negative change

Table 2.2-1: Assignment of faults and symptoms

The causal network resulting from connecting the faults with the symptoms is displayed in Fig. 2.2- 1.

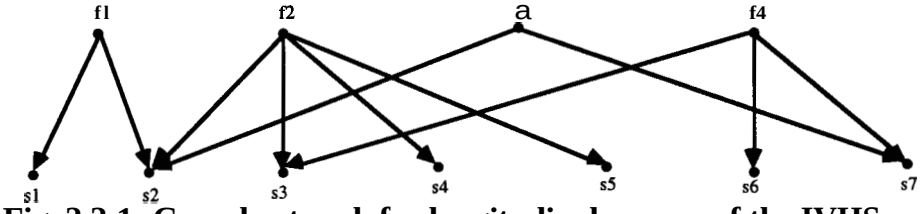


Fig. 2.2-1: Causal network for longitudinal sensors of the IVHS

In linguistic form, the rules are as follows:

• Quantization Failure:
IF quantization failure
THEN estimate is in quantization range
AND difference between estimate and measurement is large

• Out-of-Range Failure:
IF out-of-range failure
THEN change of measurement is positive very large
OR
IF out-of-range failure
THEN previous measurement is an outlier
AND current measurement has the same order of magnitude

• Sensor obstructed:
IF sensor is obstructed
THEN change of measurements is negative large
OR

IF sensor if obstructed
THEN previous measurement is obstructed
AND current measurement has the same order of magnitude.

Fuzzification of the symptoms is performed according to Fig. **2.1.2-2** (which shows fuzzification only for the symptoms of the quantification failure).

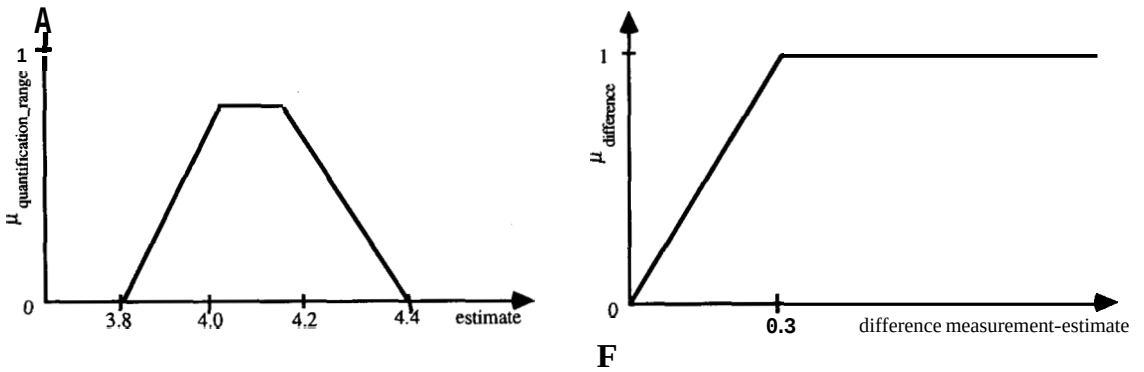


Fig. 2.2-2: Fuzzification of symptoms

Tuning of the membership functions is performed through evaluation of test data. These were obtained from platooning experiments where the follower car performed split and join maneuvers as well as straight following. Inputs to the system were the two longitudinal sensors radar and sonar sensor with sampling rate of 20 ms. Each sensor exhibited characteristic behavior in specific situations. Sensor models were built to allow simulation of the observed behavior as shown in Goebel (1996). To capture sensor failure, experiments under sub-optimal conditions were carried out as well (Bellm, 1995). These conditions include rain, **fog,** vibration, and various obstructions of the sensors. Failure modes were then developed via failure mode and effect analysis **(FMEA)** using results from these experiments which allowed modeling of various cause-effect relations between sensor failure and observable characteristics (Villanueva, 1996).

f_1	f_2	f_3	f_4
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

s_1	s_2	s_3	s_4	s_5	s_6	s_7	
0.9	0.3	0	0	0	0	0	f_1
0	0.9	0.2	0.3	0.3	0	0	f_2
0	0.4	0	0	0	0	0.6	f_3
0	0	0.3	0	0	0.7	0.2	f_4

Table 2.2-2: Modeling of one fault

s 1	s 2	s 3	s 4	s 5	s 6	s 7	f_1	f_2	f_3	f_4
0	0	0	0	0	0	0	0	0	0	0
0	0	.3	0	0	.7	.2	0	0	0	1
0	.4	0	0	0	0	.6	0	0	1	0
0	.4	.3	0	0	.7	.6	0	0	1	1
0	.9	.2	.3	.3	0	0	0	1	0	0
0	.9	.3	.3	.3	.7	.2	0	1	0	1
0	.9	.2	.3	.3	0	.6	0	1	1	0
0	.9	.3	.3	.3	.7	.6	0	1	1	1
.9	.3	0	0	0	0	0	1	0	0	0
.9	.3	.3	0	0	.7	.2	1	0	0	1
.9	.4	0	0	0	0	.6	1	0	1	0
.9	.4	.3	0	0	.7	.6	1	0	1	1
.9	.9	.2	.3	.3	0	0	1	1	0	0
.9	.9	.3	.3	.3	.7	.2	1	1	0	1
.9	.9	.2	.3	.3	0	.6	1	1	1	0
.9	.9	.3	.3	.3	.7	.6	1	1	1	1

Table 2.2-3: Modeling of several concurrent faults

Table 2.2-2 shows results for each single fault and Table 2.2-3 shows how several faults are modeled by using the maximum strength of each symptom.

The fuzzy closeness measure was assigned using the Lukasiewicz operator as outlined in Section 2.1. Symptoms were generated deliberately and in a random fashion. Some of the results for the faults and symptoms described above are summarized in Table 2.2-4.

Symptoms							Faults				gc
0.	0.	0.	0.	0.	0.	0.	0	0	0	0	1.
.1	.1	.1	.1	.1	.1	.1	0	0	0	0	.9
.2	.2	.2	.2	.2	.2	.2	0	0	0	0	.8
.3	.3	.3	.3	.3	.3	.3	0	1	0	1	.8
.5	.5	.5	.5	.5	.5	.5	1	1	1	1	.757
1.	1.	1.	1.	1.	1.	1.	1	1	1	1	.571
.1	.2	.3	.4	.5	.6	.7	0	1	1	1	.814
.7	.6	.5	.4	.3	.2	.1	1	1	0	0	.829
.1	.9	.3	.1	.6	.1	.1	0	1	0	0	.871

Table 2.2-4: Symptom observations, fault diagnosis and equivalence measure

When all symptoms are zero, no faults are diagnosed with a closeness measure of 1. If all symptoms are measured a little bit (0.1), still no fault **is** diagnosed, albeit with a smaller closeness measure. If all symptoms are measured at full strength (1), the fault combination "all faults" is diagnosed. Random combination of faults are able to diagnose failures in between the extrema. Under normal operation, the fault combination (0 0 0 0) is the most common. The fault combination (1 1 1 1) is not equal to "1" because it depends on the distance measured from the observations to the symptom set for that failure. This scheme allows us to use fault monitors with which a progression of failures for critical components can be visualized. Fig. 2.2-3 shows a

linear increase in the sensor readings with several modeled failures, in addition to the fault monitors which correctly diagnose two faults.

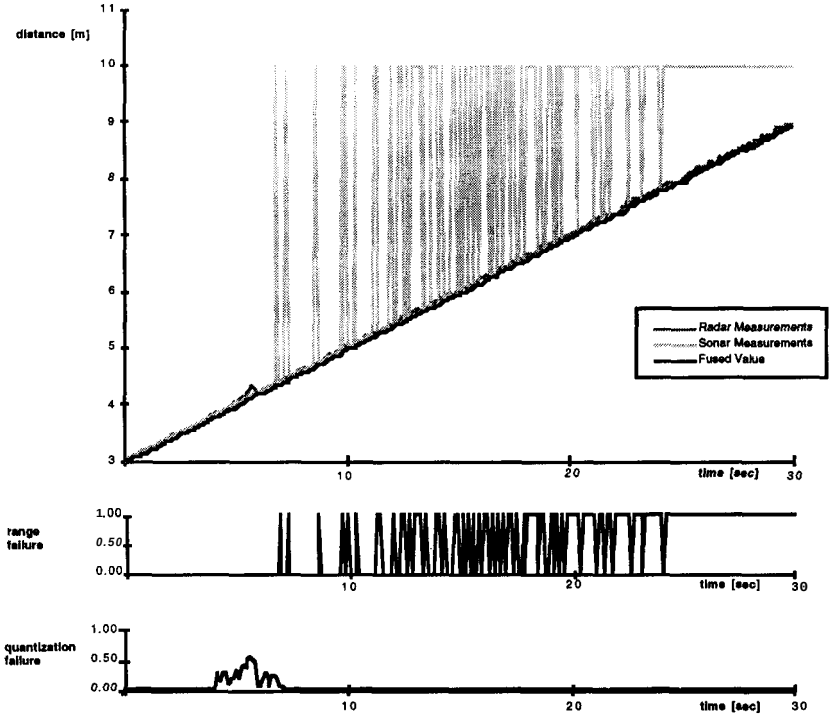


Fig. 2.2-3: Failure monitors for sensor failure

Conclusions

Fuzzy diagnosis through the use of g_C, g_S, or γ_s^* is a fast and accurate way to trace failures and their source, (if modeled) from observed symptoms. For on-line diagnosis in automated highway systems, its fast computation makes it possible for failures to be quickly detected and followed by timely reaction. Thus, the safety of the overall system is potentially improved. The results of the simulations show the method is mathematically sound and gives results which make intuitive sense. If symptoms are observed with very little strength, no fault should be diagnosed. And if all symptoms are observed at full strength, all faults should be diagnosed. The particular diagnosis will depend on the modeling criteria. Under normal operation, the fault monitors will not show anything, which is desirable. Trends of failure may show up slowly and can be detected, allowing prediction of failure in the future.

In contrast to the Bayesian approach, this approach avoids assumptions of failure independence and of relative frequency of disorder occurrence. Similarity between the two methods occurs in that links in the causal network represent causal strengths for failure-symptom relations. The potentially large number of possible covers for all possible combinations of symptoms is reduced by a ranking scheme which favors the solution with the smallest complexity.

Additional expert knowledge about the behavior of multiple fault-symptom relations may exist. This demands a symptom combination differing from the default model, ***so*** that it must be used to ensure proper working of the system. This may occur when two faults (partially) cancel their symptoms.

3. Sensor Validation and Fusion Using Bayesian Techniques

This chapter shows how diagnosis is performed using vector dynamic belief networks. After the theory is introduced in Section 3.1, applications are described in Section 3.3.

3.1 Methodology for Vector Dynamic Belief Network

This section shows the development of vector Gaussian continuous networks. These are an extension of continuous Gaussian networks (Pearl, 1988; Shachter 1987), which are directed acyclic graphs that encode probabilistic relationships between variables. Vector Gaussian continuous networks consist of composite nodes representing multivariables, that take continuous values. These vector or composite nodes can represent correlations between parents, as opposed to conventional univariate nodes. Rules for inference in these networks based on both message propagation and topology transformation are derived. The domain of application of these networks includes monitoring and estimation problems, where old algorithms such as Kalman filtering along with new algorithms can be represented and derived using these networks and rules for inference.

Introduction

Practical systems consist of variables that acquire continuous values in an operating range, e.g., the temperature of a particular component being monitored or the pressure in a chamber. Therefore, to represent these variables by means of a network structure, the nodes which represent these variables should be able to take continuous values. Pearl (1988) has developed a comprehensive scheme for inference in continuous Gaussian networks. This scheme, however is based on the assumption that the parents of each variable are uncorrelated. This assumption breaks down when the underlying network contains loops, when two or more nodes possess both common descendants and common ancestors. We present a formalism in which vector or composite nodes are used to represent dynamic systems, where parents nodes may be correlated.

These local vector nodes which are multivariate Gaussian variables correspond to ***clustering:,*** a method used to handle multiply connected networks. The links between these nodes are characterized by cross-correlation matrices. In this section, we will develop rules for inference in these compound networks. This approach combines the desirable features of message propagation including local computation, autonomy, and low storage requirements, with those of matrix techniques in the presence of multiply connected networks, or precise updating. For this, we first define and develop the vector Gaussian belief network.

We will mainly be concerned with developing rules for inference in these networks. For this task we will use both the method of message propagation (Pearl, 1988) and the method of arc reversal and topology transformation (Olmsted, 1984, Shachter, 1986, Rege and Agogino, 1988). These methods lead to the development of algorithms that can be implemented in a decentralized (parallel using multiple processors) or a centralized (single processor) architecture.

Like previous work (Pearl, 1988, Shachter and Kenley, 1989, Geiger and Heckerman, 1994) in continuous-valued variables we require that the variables have Gaussian density functions. The most compelling reason for accepting a white Gaussian assumption is that it makes the mathematics tractable. Recently, methods have been developed for implementing continuous Bayesian networks using sums of weighted Gaussians (Driver and Morrell, 1995a, Driver and Morrell, 1995b), which can approximate any distribution.

Belief Networks and Influence Diagrams

A belief network is a directed acyclic graph representing random variables. A belief network with decision nodes is known as an influence diagram, which is a network representation of probabilistic inference and decision analysis models. Each node in the network represents a variable which could be either a random variable, constant, decisions or objectives. The links represent causal influences among the random variables and the information available at the time of the decisions. Each variable may be discrete in which case it assumes an arbitrary number of mutually exclusive and exhaustive values, or it may be continuous. An absence of an arrow between two nodes represents conditional independence between the variables.

Each node has a conditional probability table that quantifies the effects that the parents have on the node. The parents of a node are all those nodes that have arrows pointing to it. Each node without a parent requires a prior probability for each state. Deterministic relationships between variables is a special case that is handled by setting each conditional probability to either 0 or 1. The joint probability distribution of the variables represented in the belief network can be calculated from the information in the network. Let, $X_1,..,X$, be the random variables represented by the belief network. Then the probability of a conjunction of a particular value of each variable is given by

$$P(X_1 = x_1 \wedge .. \wedge X_n = x_n) = P(x_1,..,x_n) = \prod_{i=1}^{n} P(x_i | Parents(X_i))$$

At this point it is helpful to review the Bayesian interpretation **of** probability. One prevalent notion of probability is that it is a measure of the frequency with which an event occurs. An event is a state of some part of our world in some time interval in the past, present, or future. This interpretation of probability as a frequency in a series **of** repeated experiments is traditionally referred to as the ***objective*** or ***frequentist*** interpretation. **A** different notion of probability is the ***degree*** of ***belief*** held by a person that the event will occur. This interpretation of probability is called ***subjective*** or ***Bayesian*** interpretation. Frequentist interpretation is a special case of the Bayesian interpretation. Thus in the Bayesian interpretation a probability or belief always depends on the state of knowledge of the person who provides the probability (Heckerman, 1995).

Each entry in the joint distribution is represented by the product of the appropriate elements of the conditional probability tables (CPTs) in the belief network. Each node in the belief network is conditionally independent of its parents' predecessors given its parents --- given $Parents(X_i) \subseteq \{x_{i-1},..,x_1\}$, then the specification of the joint is equivalent to

$$P(X_i | X_{i-1},..,X_1) = P(X_i | Parents(X_i)).$$

A belief network can handle a large number of pieces of evidence without the exponential growth in the conditional probability values due to its ***sparse*** structure. In a locally structured or a sparse system, each subcomponent interacts directly with only a bounded number of other components, regardless of the total number of components.

Belief networks can be singly or multiply connected. An acyclic graph is singly connected if there is at most one chain (or undirected path) between each pair of variables. Networks with undirected cycles are multiply connected. In a probabilistic inference system evidence comes in one or more nodes, and one needs to compute the posterior probability distribution for other ***query*** nodes.

There are a number of inference algorithms for a simply connected network. There are three basic classes of algorithms for evaluating multiply connected networks, each with its own area of applicability. Clustering methods, conditioning methods and stochastic simulation. For more details on belief Networks interested readers are directed to Pearl (1988, 1995) and Russell and Norvig (1995) .

Over the past decade, Bayesian networks have become a tool of great versatility and power, and have become the most common representation scheme for probabilistic knowledge (Shachter, 1990; Shafer and Pearl, 1990). This representation has been useful for modeling many real world problems including diagnosis of medical patients (Heckerman et al., 1992), forecasting (Gu et al., 1994), automated vision (Levitt et al., 1990), semiconductor manufacturing control (Nadi et al., 1991), supervisory control of robotic manipulator (Ramamurthi and Agogino, 1993), monitoring and diagnosis of manufacturing processes (Agogino et al., 1988), information retrieval (Turtle and Croft, 1991), etc.

3.2 Continuous (Gaussian) Belief Networks

Consider a domain $\vec{x}$, of n continuous variables $x_1,,,,x_n$. The joint probability density function for $\vec{x}$ is a multivariate nonsingular normal distribution

$$\Pr(\vec{x}) = N(x;\bar{x},P) \stackrel{\Delta}{=} |2\pi P|^{-1/2} \exp\left(-\frac{1}{2}(x-\bar{x})^T P^{-1}(x-\bar{x})\right)$$

where $N(.)$ denotes the normal pdf with argument **x,** and

$$\mathrm{X} = E[x]$$

$$\boldsymbol{P} = E\left[(x-\bar{x})^T(x-\bar{x})\right]$$

are respectively, the mean and covariance matrix of the vector ***X.*** The inverse of the covariance matrix P^{-1} is also known as the precision or the information matrix. Note that to avoid indicating the dimension of the vector ***X*** the determinant in the above equation has been written with the factor 2π inside it. An important property of Gaussian random variables is that they stay Gaussian under linear transformation.

The joint distribution of the random variables can also be written as a product of conditional distributions each being an independent normal distribution, namely

$$\Pr(x) = \prod_{i=1}^{n} \Pr\left(x_i | x_1,..,x_{i-1}\right)$$

$$\Pr\left(x_i | x_1,..,x_{i-1}\right) = N\left(x_i, m_i + \sum_{j=1}^{i-1} b_{ij}\left(x_j - m_j\right), \sigma_i^2\right)$$

where m_i is the unconditional mean of x_i, σ_i^2 is the conditional variance of x_i given values for $\mathbf{x,},..,x_{i-1}$, and b_{ij} is a linear coefficient reflecting the strength of the relationship between x_i and x_j (e.g., DeGroot (1970). Hence, one can interpret a multivariate normal distribution as a belief

network, where there is an arc from x_j to x_i whenever $b_{ij} \neq 0, j < i$ (Geiger and Heckerman, 1994). This special form of belief network is commonly known as the Gaussian belief network, the name having been adopted from Shachter and Kenley (1989) who first described Gaussian influence diagrams.

The Gaussian-influence diagram representation of a multivariate normal distribution is better suited to model elicitation and understanding than is the standard representation (Shachter and Kenley, 1989). For a user to access a Gaussian belief network the following need to be specified: (1) the unconditional mean of each variable $x_i(m_i)$, **(2)** the relative importance of each parent x_j in determining the values of its child x_i (b_{ij}), and **(3)** a conditional variance for x_i given that its parents are fixed (σ_i^2) (Geiger and Heckerman, 1994).

Pearl (1988) has developed an encoding scheme for representing continuous variables in a belief network. The developed encoding scheme is based on the following assumptions:' (1) all interaction between variables are linear; **(2)** the sources of uncertainty are normally distributed and are uncorrelated; and **(3)** the belief network is singly connected[2].

Pearl (1988) considers a hierarchical system of continuous random variables like the one shown in Fig. **3.2-1**. Each variable X has a set of parent variables $U_1, U_2, .., U_n$ and **a** set of children variables $Y_1, Y_2, .. Y_m$. The relation between X and its parents is given by the linear equation

$$X = b_1 U_1 + b_2 U_2 + \cdots + b_n U_n + w_x$$

where $b_1, b_2, .., b_n$ are constant coefficients representing the relative contribution made by each of the U variables to the determination of the dependent variable X and w_X is a noise term summarizing other factors affecting X. Variable w_X is assumed to be normally distributed with a zero mean, and uncorrelated with any other noise variable.

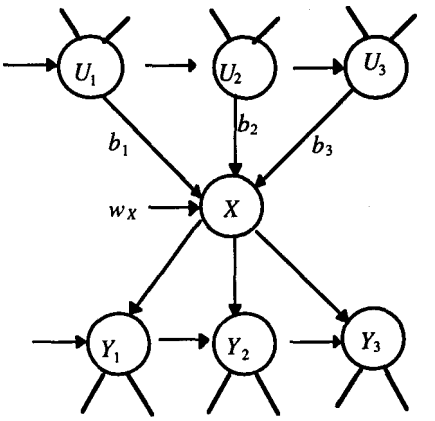


Fig. 3.2-1: A fragment of a singly-connected network showing the relationships between a continuous variable X, its parent variables

[1]The first two assumptions are the same **as** those made by Shachter and Kenley, 1989.
[2]**A** network in which no more than one path exists between two nodes.

Given the network topology, the link coefficients (b's), the variances (σ_{w_X}) of the noise terms, and the means and variance of the root variables (nodes with no parents), Pearl **(1988)** has developed a distributed scheme for updating the means and variances of every variable in the network to account for evidential data ***e,*** a set of variables whose values have been determined precisely. The update can also be done in a manner similar to that in Shachter and Kenley (1989). However, Pearl places an additional restriction that the computation be conducted in a ***distributed*** fashion, as though each variable were managed by a separate and remote processor communicating only with processors that are adjacent to it in the network, i.e., the update is performed in a decentralized (parallel) manner.

The equation relating the variable of interest to its parents replaces the conditional probability tables that are required in the case the variables are discrete. In addition, due to the assumption that the variables are normally distributed the complete distribution can be specified with the help of just two parameters, the mean and the variance.

We extend these continuous Gaussian networks to its vector form. Fig. **3.2-2** shows a generic form of a vector Gaussian network. Here, all the variables represented by the nodes (U_i, X, Y_i) are vectors, for e.g., $X \equiv [x_1, \cdots, x_n]^T$ where x_i are Gaussian random variables. The arc between the variables represent the following relationship

$$X = \sum_i F_i \cdot U_i + v; \quad Y_i = H_i X + w_i$$

where U_i and Y_i are n_{U_i} and n_{Y_i} dimension vector, F_i is a $n_n \mathbf{x} n_{U_i}$ matrix, H_i is a n_{Y_i} xn, matrix v is a n_n vector, $Q = E[w_X w_X^T]$, the covariance matrix for the noise term represents the correlation between the parent variable. It can be easily shown that in the case of multiple parent nodes $[U_1, U_2]^T$ with corresponding state matrices F_1, F_2 respectively, one can obtain an equivalent matrix

$$F_{equivalent} = \begin{bmatrix} F_1 & 0 \\ 0 & F_2 \end{bmatrix}$$

which express the augmented state vector $[X_1, X_2]^T$. This procedure is known as clustering of the nodes. One may however want to keep the nodes separate due to which we will work with a network that has multiple parents. We will in general cluster the variables into nodes in which no evidence comes in at any of the nodes, nodes with different covariance matrices, etc. Here, each node can be considered as made up of another Gaussian belief network where the inter-relationship between the variables is given by the matrix F.

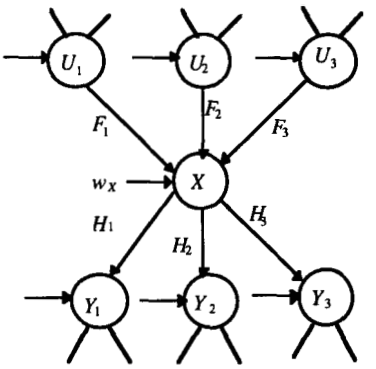


Fig. 3.2-2: Generic form of a vector Gaussian belief network

Vector Rules for Normal Distributions

Rule 1

$$N(x;P,\bar{x}) = |2\pi P|^{-1/2}\exp\left(-\frac{1}{2}(x-\bar{x})^T P^{-1}(x-\bar{x})\right)$$

Rule 2

$$N(x;P,\bar{x}) = N(\bar{x},P,x)$$

Rule 3

$$N\left(y = Ax + B;P_y,\bar{y}\right) = a\cdot N\left(x;P_x = \left[A^T P_y^{-1}A\right]^{-1},P_x\left[A^T P_y^{-1}(\bar{y}-B)\right]\right)$$

Rule 4

$$N\left(x;P_1,\bar{x}_1\right)\cdot N\left(x;P_2,\bar{x}_2\right) = a\cdot N\left(x;\left[P_1^{-1}+P_2^{-1}\right]^{-1},\left[P_1^{-1}+P_2^{-1}\right]^{-1}\left[P_1^{-1}\bar{x}_1+P_2^{-1}\bar{x}_2\right]\right)$$

where the constant a is given by

$$a = N\left(\bar{x}_1,P_1+P_2,\bar{x}_2\right)$$

$$N\left(x;P_1,\bar{x}_1\right)\cdot N\left(x;P_2,\bar{x}_2\right) = a\cdot N\left(x;\left[P_1^{-1}+P_2^{-1}\right]^{-1},\left[P_1^{-1}+P_2^{-1}\right]^{-1}\left[P_1^{-1}\bar{x}_1+P_2^{-1}\bar{x}_2\right]\right)$$

Rule 5

$$\prod_I N\left(x;P_i,\bar{x}_i\right) = a\cdot N\left(x;\left[\sum_i P_i^{-1}\right]^{-1},\left[\sum_i P_i^{-1}\right]^{-1}\left[\sum_i P_i^{-1}\bar{x}_i\right]\right)$$

here again a is a constant, The exact form of this constant is not important as it cancels out during the probability update process. The form of this constant for the scalar case is given in Driver and Morrell (1995b).

Rule 6

$$\prod_i N\left(Ax;P_i,\bar{y}_i\right) = a\cdot N\left(x;\left[\sum_i A^T P_i^{-1}A\right]^{-1},\left[\sum_i A^T P_i^{-1}A\right]^{-1}\left[\sum_i A^T P_i^{-1}\bar{y}_i\right]\right)$$

Rule 7

$$\int N(y;P_1,\bar{x}_1)N(y;P_2,x)dy = N(x;P_1+P_2,\bar{x}_1)$$

Rule 8

$$\int_y N(x;P_x,y)N(y;P_y,\bar{y})dy = N[x;P_x+P_y,\bar{y}]$$

Rule 9

$$\int_{u_1}\cdots\int_{u_n}\prod_{i=1}^{n} N(u_i,P_i,\bar{u}_i)\cdot N\left(\sum_j B_j u_j,Q,x\right)du_1\cdots du_n = N\left(x;Q+\sum_j B_j P_j B_j^T,\sum_j B_j\bar{u}_j\right)$$

Inference using Message Propagation: Decentralized Approach

Consider a typical fragment of a singly connected network **(Fig.** 3.2-2), consisting of an arbitrary node X, the set of all X's parents, $U=\{U_1,U_2,\cdots,U_n\}$, and the set of all $X's$ children, $Y=\{Y_1,Y_2,\cdots,Y_m\}$. Let **e** be the total evidence obtained, $\mathbf{e}_X^-$ be the evidence connected to X through its children **(Y),** and $\mathbf{e}_X^+$ be the evidence connected to X through its parents (U). Readers are referred to Pearl **(1988)** where the rules for propagation are derived for the scalar case. To derive the inference rules for vector Gaussian networks, we will follow a procedure very similar to Pearl's for the scalar case.

Belief Update

To compute the belief of **X,** i.e., *BEL(x)*, we divide the evidence **e** into two components, $\mathbf{e}_x^+$ and $\mathbf{e}_x^-$ representing data in the sub networks above **X** and below **X,** respectively.

We consider the general belief network structure shown in Fig. 3.2-2. Here, the links correspond to

$$x=\sum_i F_i\cdot u_i+v$$
$$y_i=H_i x+w_i$$

We calculate the belief for node X, as follows

$$BEL(x)=f(x|\mathbf{e}_x^+,\mathbf{e}_x^-)=\alpha\cdot f(x|\mathbf{e}_x^+)f(\mathbf{e}_x^-|x)=\alpha\cdot\pi(x)\cdot\lambda(x)$$
$$\pi(x)=f(x|\mathbf{e}_x^+)=\int_{U_1}\cdots\int_{U_n} f(x|\mathbf{e}_x^+,u_1,\cdots,u_n)f(u_1,\cdots,u_n|\mathbf{e}_x^+)du_1\cdots du_n$$
$$=\int_{U_1}\cdots\int_{U_n} f(x|u_1,\cdots,u_n)\prod_{i=1}^{n} f(u_i|e_i^+)du_1\cdots du_n$$
$$=\int_{U_1}\cdots\int_{U_n} N\left(x;Q,\sum_{i=1}^{n}B_i u_i\right)\prod_{i=1}^{n} N(u_i;P_i^+,\bar{u}_i^+)du_1\cdots du_n$$

$$\pi(x) = N\left(x; \sum_{i=1}^{n} B_i P_{u_i} B_i^T + Q, \sum_{i=1}^{n} B_i \bar{u}_i^+\right)$$
$$= N[x; P_\pi, \bar{x}_\pi]$$

The variance P_π can be interpreted as sum of the uncertainty in each of the u_i ($\sum_{i=1}^{n} B_i P_{u_i} B_i^T$) and the uncertainty in the relationship between x and u_i (Q).

Similarly, we now compute $\lambda(x)$. Let Φ denote the set of child nodes of X, let $\Omega = \{j \in \Phi | \mathbf{e}_j^- \neq \mathbf{O}\}$, and let m be the number of elements in Ω. Next, we relabel the child nodes so that nodes Y_1 through Y_m correspond to the nodes with $\mathbf{e}_j^- \neq \mathbf{O}$. If m = 0 then $\lambda(x) \triangleq 1$. If m = 1 then $\lambda(x) \triangleq \lambda_{Y_m}(x)$. For $m \geq 2$, we compute $\lambda(x)$ as follows:

$$\lambda(x) = f(\mathbf{e}_x^-|x) = f(\mathbf{e}_1^-, \mathbf{e}_2^-, .., \mathbf{e}_m^-|x) = \prod_j f(\mathbf{e}_j^-|x) = \prod_j \lambda_j(x)$$
$$\lambda(x) = \prod_j N(H_j x; R_j, \bar{y}_j)$$
$$= a \cdot N\left(x; \left[\sum_j H_j^T R_j^{-1} H_j\right]^{-1}, \left[\sum_j H_j^T R_j^{-1} H_j\right]^{-1} \left[\sum_j H_j^T R_j^{-1} \bar{y}_j\right]\right)$$
$$= a \cdot N(x; P_\lambda, \bar{x}_\lambda)$$

where again a is some constant the exact form of which is not important. **As** we will soon see it cancels out during the belief update process.

Note:

$$P_\lambda^{-1} = \sum_j H_j^T R_j^{-1} H_j \text{ and } P_\lambda^{-1} \bar{x}_\lambda = \sum_j H_j^T R_j^{-1} \bar{y}_j$$

To update x we do not require $\left[\sum_j H_j^T R_j^{-1} H_j\right]^{-1}$ to exist. P_λ^{-1}, the inverse of the covariance matrix is called the information matrix. We also define a transformed state vector $z_\lambda \triangleq P_\lambda^{-1} x_\lambda$ and $\therefore \bar{z}, = P_\lambda^{-1} \bar{x}_\lambda$. It is important to note that P_λ^{-1} is really the covariance of the information state vector z, $P_\lambda^{-1} x_\lambda$. Hence, $N(x; P_\lambda, \bar{x}_\lambda) = a, \cdot N(z; P_\lambda^{-1}, \bar{z})$, where again a, is some constant.

If $\mathbf{e}_j^- = 0$ then $\lambda(x) \triangleq 1$. Combining these two results, we obtain:

$$
\begin{aligned}
BEL(x) &= f\left(x \middle| e_x^+, e_x^-\right) \\
&= \frac{f\left(e_x^- \middle| x, e_x^+\right) \cdot f\left(x \middle| e_x^+\right)}{\int_x f\left(e_x^- \middle| x, e_x^+\right) \cdot f\left(x \middle| e_x^+\right) \cdot dx} \\
&= \frac{a \cdot N\left(x; P_\pi, \bar{x}_\pi\right) N\left(x; P_\lambda, \bar{x}_\lambda\right)}{a \cdot \int_x N\left(x; P_\pi, \bar{x}_\pi\right) N\left(x; P_\lambda, \bar{x}_\lambda\right) dx} \\
&= \frac{N\left(x; \left[P_\pi^{-1} + P_\lambda^{-1}\right]^{-1}, \left[P_\pi^{-1} + P_\lambda^{-1}\right]^{-1}\left[P_\pi^{-1}\bar{x}_\pi + P_\lambda^{-1}\bar{x}_\lambda\right]\right) \cdot N\left(\bar{x}_\pi, P_\pi + P_\lambda, \bar{x}_\lambda\right)}{N\left(\bar{x}_\pi, P_\pi + P_\lambda, \bar{x}_\lambda\right)} \\
&= N\left(x; \left[P_\pi^{-1} + P_\lambda^{-1}\right]^{-1}, \left[P_\pi^{-1} + P_\lambda^{-1}\right]^{-1}\left[P_\pi^{-1}\bar{x}_\pi + P_\lambda^{-1}\bar{x}_\lambda\right]\right) \\
&= N\left(x; P_\pi - P_\pi\left[P_\pi + P_\lambda\right]^{-1} P_\pi, \bar{x}_\pi + P_\pi\left[P_\pi + P_\lambda\right]^{-1}\left(\bar{x}_\lambda - \bar{x}_\pi\right)\right)
\end{aligned}
$$

As can be seen all constants associated with $\lambda(x)$ and $\pi(x)$ cancel out. Hence, the exact form of these constants is not important. Therefore, in the remaining part of this section, we will neglect the constants. Also, for $\lambda(x)$ it is easier to work with the information state vector ***z*** and the information matrix P_λ^{-1}.

During implementation, we begin with the leaf nodes. Here, λ messages are sent to the parent nodes and this propagation stops at the root nodes. Then, starting from the root nodes, the node calculates its belief and sends π messages to each of its children. The propagation stops on reaching a leaf node, i.e., a node with no child. This propagation scheme is guaranteed to stop **as** the networks are acyclic, are singly-connected, and have a finite number of nodes.
To prescribe how the influence of new information will spread through the network, we need to specify how a typical node, say X, will compute its outgoing messages $\lambda_X(u_i)$, i $=1,..,n$, and $\pi_{Y_j}(x), j = 1,..,m$, from the incoming messages λ_{Y_j} **(x),** $j = 1,..,m$ and $\pi_X(u_i)$, i $= 1,..,n$.

Top Down Propagation: Message to Children

Consider the message π_{Y_j} **(x),** which node X sends to its jth child Y_j (j=1,2,..,m), we note that it is conditioned on all data except a subset $\mathbf{e}_j^-$ of variables that connect to ***X*** via Y_j. Therefore,

$$
\begin{aligned}
\pi_{Y_j}(x) &= f\left(x \middle| \mathbf{e} - \mathbf{e}_j^-\right) = BEL\left(x \middle| \mathbf{e}_j^- = \phi\right) \\
&= N\left(x; P_{Y_j}^+, \bar{x}_{Y_j}^+\right) \\
\pi_{Y_j}\left(x_j\right) &= N\left(y_j; H_j P_{Y_j}^+ H_j^T + R_i, H_j^T \bar{x}_{Y_j}^+\right)
\end{aligned}
$$

So, $\pi_{Y_j}(x)$ can be computed by the method of the last section with the assumption that $\lambda_Y(x) = 1$. Hence,

$$P^+_{Y_j} = \left[P_\pi^{-1} + \sum_{k \neq j} H^T R_k^{-1} H \right]$$

$$= P_\pi - P_\pi \left[P_\pi + \sum_{k \# j} H^T R_k^{-1} H \right]^{-1} P_\pi$$

$$\bar{x}^+_{Y_j} = \left[P_\pi^{-1} + \sum_{k \# j} H^T R_k^{-1} H \right] \left[P_\pi^{-1} \bar{x}_\pi + \sum_{k+j} H^T R_k^{-1} \bar{y}_k \right]$$

$$= \bar{x}_\pi + \left[\sum_{k \neq j} H^T R_k^{-1} H \right] \left[P_\pi + \sum_{k \neq j} H^T R_k^{-1} H \right] \left(\sum_{k \neq j} H^T R_k^{-1} \bar{y}_k - \bar{x}_\pi \right).$$

Bottom Up Propagation: Message to Parents

Consider the message $\lambda_X(u_i)$, which node X sends to its ith parent U_i. We divide the evidence $\mathbf{e}$ into its disjoint components $\mathbf{e}_i^+$, i=1,..,n and $\mathbf{e}_j^-$, j=1,..,m, and condition $\lambda_x(u_i)$ on all parents of **X.** For notational convenience we temporarily denote U_i by **U** and b_i by b, and let the other parents be indexed by k, ranging from I to some n:

$$\lambda_x(u) = f(\mathbf{e} - \mathbf{e}_U^+ | u)$$

$$= \int_{u_1} \cdots \int_{u_n} \int_x f(e_1^+, .., e_n^+, e_1^-, .., e_m^- | u_1, .., u_n, x, u) \cdot f(u_1, .., u_n, x | u) dx \cdot du_1 \cdots du_n$$

Consider the first distribution in the integrand:

$$f(e_1^+, .., e_n^+, e_1^-, .., e_m^- | u_1, .., u_n, x, u) = f(e_1^-, .., e_m^- | x) \cdot f(e_1^+, .., e_n^+ | u_1, .., u_n, x, u)$$

$$= \prod_j \lambda_{Y_j}(x) \cdot \prod_k f(\mathbf{e}_k^+ | u_k)$$

$$= \lambda(x) \prod_k \frac{f(u_k | \mathbf{e}_k^+) f(\mathbf{e}_k^+)}{\int_{\mathbf{e}_k^+} f(u_k | \mathbf{e}_k^+) f(\mathbf{e}_k^+) d\mathbf{e}_k^+}$$

Next, consider the second distribution in the integral:

$$f(u_1, .., u_n, x | u) = f(x | u, u_1, .., u_n) f(u_1, .., u_n)$$

$$= f(x | u, u_1, .., u_n) \prod_k f(u_k)$$

$$= f(x | u, u_1, .., u_n) \prod_k \int_{\mathbf{e}_k^+} f(u_k | \mathbf{e}_k^+) f(\mathbf{e}_k^+) d\mathbf{e}_k^+$$

$$f\left(e_1^+,..,e_n^+,e_1^-,..,e_m^-|u_1,..,u_n,x,u\right)=f\left(e_1^-,..,e_m^-|x\right)\cdot f\left(e_1^+,..,e_n^+|u_1,..,u_n,x,u\right)$$
$$=\prod_j \lambda_{Y_j}(x)\cdot\prod_k f\left(\mathbf{e}_k^+|u_k\right)$$
$$=\lambda(x)\prod_k \frac{f\left(u_k|\mathbf{e}_k^+\right)f\left(\mathbf{e}_k^+\right)}{\int\limits_{\mathbf{e}_k^+} f\left(u_k|\mathbf{e}_k^+\right)f\left(\mathbf{e}_k^+\right)d\mathbf{e}_k^+}$$

Substituting

$$f\left(u_1,..,u_n,x|u\right)=f\left(x|u,u_1,..,u_n\right)f\left(u_1,..,u_n\right)$$
$$=f\left(x|u,u_1,..,u_n\right)\prod_k f\left(u_k\right)$$
$$=f\left(x|u,u_1,..,u_n\right)\prod_k \int\limits_{\mathbf{e}_k^+} f\left(u_k|\mathbf{e}_k^+\right)f\left(\mathbf{e}_k^+\right)d\mathbf{e}_k^+$$

and

$$\lambda_x(u)=\int\limits_{u_1}..\int\limits_{u_n}\int\limits_x \lambda(x)\prod_k f\left(u_k|\mathbf{e}_k^+\right)f\left(\mathbf{e}_k^+\right)f\left(x|u,u_1,..,u_n\right)dx\cdot du_1\cdot\cdot du_n$$
$$=\int\limits_{u_1}..\int\limits_{u_n}\int\limits_x \lambda(x)\prod_k \pi_X(u_k)f\left(x|u,u_1,..,u_n\right)dx\cdot du_1\cdot\cdot du_n$$

where:

$$\lambda(x)=N\left(x;P_\lambda,\bar{x}_\lambda\right)=a\cdot N\left(z;P_\lambda^{-1},\bar{z}\right),\qquad \pi_X\left(u_k\right)=N\left(u_k;P_{u_k}^+,\bar{u}_k\right)$$

$$f\left(x|u,u_1,..,u_n\right)=N\left(x;Q,Bu+\sum_k B_k u_k\right)$$

Using the properties for vector normal distributions:

$$\lambda_x(u)=\int\limits_{u_1}..\int\limits_{u_n}\int\limits_x N\left(x;P_\lambda,\bar{x}_\lambda\right)\prod_k N\left(u_k;P_{u_k}^+,\bar{u}_k\right)N\left(x;Q,Bu+\sum_k B_k u_k\right)dx\cdot du_1\cdot\cdot du_n$$

Integrate with respect to **x:**

$$=\int\limits_{u_1}..\int\limits_{u_n}\prod_k N\left(u_k;P_{u_k}^+,\bar{u}_k\right)N\left(\bar{x}_\lambda;P_\lambda+Q,Bu+\sum_k B_k u_k\right)du_1\cdot\cdot du_n$$

$$\lambda_x(u)=N\left(Bu;P_\lambda+Q+\sum_k B_k P_{u_k}^+ B_k^T,\bar{x}_\lambda-\sum_k B_k\bar{u}_k\right)$$

$$=N\left(u;P_x(u)=\left[B^T\left(P_\lambda+Q+\sum_k B_k P_{u_k}^+ B_k^T\right)^{-1}B\right]^{-1},\boldsymbol{P_x(u)}\left[B^T\left(P_\lambda+Q+\sum_k B_k P_{u_k}^+ B_k^T\right)^{-1}\left(\bar{x}_\lambda-\sum_k B_k\bar{u}_k\right)\right]\right)$$

$$= N\left(P_x^{-1}(u)\cdot u; P_x^{-1}(u) = B^T\left(P_\lambda + Q + \sum_k B_k P_{u_k}^+ B_k^T \right)^{-1} B, \left(P_\lambda + Q + \sum_k B_k P_{u_k}^+ B_k^T \right)^{-1} \left(\bar{x}_\lambda - \sum_k B_k \bar{u}_k \right) \right)$$

Therefore, for the ith parent, U_i we have:

$$\lambda_x(u_i) = N\left(u_i; P_X^-(u_i), \bar{x}_x^-(u_i)\right)$$

$$N\left(u_i; \left[B_i^T \left(P_\lambda + Q + \sum_{k\neq i} B_k P_{u_k}^+ B_k^T \right)^{-1} B_i \right]^{-1}, P_X^-(u_i) \left[B_i^T \left(P_\lambda + Q + \sum_{k\neq i} B_k P_{u_k}^+ B_k^T \right)^{-1} \left(\bar{x}_\lambda - \sum_{k\neq i} B_k \bar{u}_k \right) \right] \right)$$

where

$$P_\lambda = \left[\sum_j H_j^T R_j^{-1} H_j \right]^{-1}; \quad P_\lambda^{-1} = \sum_j H_j^T R_j^{-1} H_j$$

$$\bar{x}_\lambda = \left[\sum_j H_j^T R_j^{-1} H_j \right]^{-1} \left[\sum_j H_j^T R_j^{-1} \bar{y}_j \right]; \quad P_\lambda^{-1} \bar{x}_\lambda = \sum_j H_j^T R_j^{-1} \bar{y}_j$$

Predictive Estimation: No Evidence in Children Nodes

If $\lambda(x) = 1$ i.e., there is no evidence from the children we have

$$\lambda_x(u) = \int_{u_1} \cdots \int_{u_n} \int_x \prod_k N\left(u_k; P_{u_k}^+, \bar{u}_k\right) N\left(x; Q, B\bar{u} + \sum_k B_k \bar{u}_k \right) dx \cdot du_1 \cdots du_n$$

$$= \int_{u_1} \cdots \int_{u_n} \prod_k N\left(u_k; P_{u_k}^+, \bar{u}_k\right) du_1 \cdots du_n$$

$$= 1$$

which implies that evidence gathered at a node does not affect any of its spouses until their common child node obtains evidence. This reflects the d-separation condition and matches our intuition regarding multiple causes.

Alternate Forms for Message to Parents

We can simplify the belief update process for the parent nodes u_i.

$$P_\pi(u_i) = N\left(u_i, P_{u_i}, \bar{u}_i\right) = N\left(B_i u_i; B_i P_{u_i} B_i^T, B_i \bar{u}_i\right)$$

$$P_\lambda(u_i) = N\left(B_i u_i; P_\lambda + Q + \sum_{k\neq i} B_k P_{u_k} B_u^T, \bar{x}_\lambda - \sum_{k\neq i} B_k \bar{u}_k \right)$$

Combining the above two equations to update the belief in u_i :

$$N\left(B_i u_i; B_i P_{u_i}^{new} B_i^T, B_i \bar{u}_i^{new}\right)$$

$$B_i P_{u_i}^{new} B_i^T = \left(B_i P_{u_i} B_i^T\right) - \left(B_i P_{u_i} B_i^T\right)\left[P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1}\left(B_i P_{u_i} B_i^T\right)$$

$$B_i \bar{u}_i^{new} = B_i \bar{u}_i + \left(B_i P_{u_i} B_i^T\right)\left[P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1}\left(\bar{x}_\lambda - \sum_k B_k \bar{u}_k\right)$$

Therefore,

$$N\left(u_i; P_{u_i}^{new}, \bar{u}_i^{new}\right)$$

$$= N\left(u_i; P_{u_i} - P_{u_i} B_i^T\left[P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right] B_i P_{u_i}, \bar{u}_i + P_{u_i} B_i^T\left[P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1}\left(\bar{x}_\lambda - \sum_k B_k \bar{u}_k\right)\right)$$

when the above formula is used we require $P_\lambda = \left[\sum_j H_j^T R_j^{-1} H_j\right]^{-1}$ to exist.

We can remove this requirement by noting

$$P_\lambda^{-1} = \sum_{j=1}^{m} H_j^T R_j^{-1} H_j \triangleq H^T R^{-1} H$$

$$H = \left[H_1^T, \cdot\cdot, H_m^T\right]^T \qquad \mathbf{R} = blockdiag\{R_1, \cdot\cdot, R_m\}$$

$$\therefore R = H P_\lambda H^T \quad P_\lambda^{-1} = H^T\left[H P_\lambda H^T\right]^{-1} H$$

where m is the number of children nodes of **X** through which there is evidence. But

$$\left[P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1} = H^T\left[H\left(Q + \sum_k B_k P_{u_k} B_u^T\right)H^T + R\right]^{-1} H$$

For a proof of the above identity note the following:

$$H^T\left[H\left(Q + \sum_k B_k P_{u_k} B_u^T\right)H^T + R\right]^{-1} H = H^T\left[H\left(Q + \sum_k B_k P_{u_k} B_u^T + P_\lambda\right)H^T\right]^{-1} H$$

$$= \left(P_\lambda + Q + \sum_k B_k P_{u_k} B_u^T\right)^{-1}$$

Further,

$y = \left[y_1^T, \cdot\cdot, y_m^T\right]^T$ and $\bar{y} = \left[\bar{y}_1^T, \cdot\cdot, \bar{y}_m^T\right]^T$

Using this,

$$P_\lambda^{-1}\bar{x}_\lambda = \sum_{j=1}^{m} H^T R_j^{-1}\bar{y}_j = H^T R^{-1}\bar{y}$$

$$H^T R^{-1} H\bar{x}_\lambda = H^T R^{-1}\bar{y}$$

$$\therefore H\bar{x}_\lambda = \bar{y}$$

$$N\left(u_i; P_{u_i}^{new}, \bar{u}_i^{new}\right) = N\left(\begin{array}{l} u_i; P_{u_i} - P_{u_i} B_i^T H^T \left[H\left(Q + \sum_k B_k P_{u_k} B_u^T\right) H^T + R \right]^{-1} H B_i P_{u_i}, \\ \bar{u}_i + P_{u_i} B_i^T H^T \left[H\left(Q + \sum_k B_k P_{u_k} B_u^T\right) H^T + R \right]^{-1} \left(\bar{y} - H\sum_k B_k \bar{u}_k\right) \end{array}\right|$$

is an alternate form for updating the beliefs. The above alternative form is particularly useful in deriving the decentralized form of the Kalman filter.

Boundary Conditions

The boundary conditions for vector Gaussian continuous networks are established as follows

1. If **X** is a root node (a node with no parents) that has not been instantiated, then we set $\pi(x)$ equal to the prior density function f(x).

2. If **X** is a leaf node (a node with no children) that has not been instantiated, then we set $\lambda(\mathbf{x}) = 1$. This implies that $Bel(\mathbf{x}) = \pi(\mathbf{x})$.

3. If **X** is an evidence node, say $X = \mathrm{X}$, then we set $\lambda(x) = \delta(x - \mathrm{X}) = N(x; 0, \bar{x})$ regardless of the incoming λ-messages. This implies that $Bel(x) = N(x; 0, \bar{x})$ as expected. Furthermore, for each j, $\pi_{Y_j}(\mathbf{x}) = N(x; 0, \bar{x})$ is the message that node **X** sends to its children (in this case each child gets the same message).

4. To interpret the updated process note that if there was evidence at *x* then the new value would have been:

$$N\left(u_i; P_{u_i} - P_{u_i} B_i^T \left[Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1} B_i P_{u_i}, \bar{u}_i + P_{u_i} B_i^T \left[Q + \sum_k B_k P_{u_k} B_u^T\right]^{-1} \left(\bar{x}_{evidence} - \sum_k B_k \bar{u}_k\right)\right)$$

So, the new update can be interpreted that there is new evidence at x (i.e., x is $\bar{x}_\lambda$), however there is uncertainty associated with this new evidence given by P_λ.

5. *An* alternate form of the message sent by X's children is

$$P_\lambda^{-1} = H^T R^{-1} H \quad P_\lambda^{-1}\bar{x}_\lambda = H^T R^{-1}\bar{y}.$$

6. The covariance matrices are normally positive definite and symmetric. Due to which all the required inverses exist.

Example: Inference in Vector Continuous Probabilistic Networks

To illustrate the inference process we consider a simple example motivated from a chemical process. The example has been suitably modified to illustrate the various inference processes.

Three reactants (the concentration of two of which are correlated) combine to form two products. The concentration needs to be estimated. There are two sensor nodes, Y_1 and Y_2, that measure a combination of these reactants. The process is represented by the following vector Gaussian network.

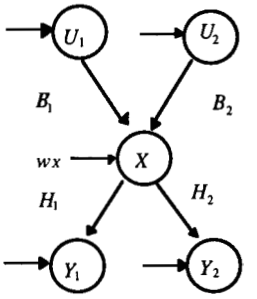


Fig. 3.2-3: The vector belief network used in the example.

$$x_3 = B_1 x_1 + B_2 x_2 + w_3$$
$$x_4 = H_1 x_3 + w_4$$
$$x_5 = H_2 x_3 + w_5$$

where

$$\mathbf{B}_1 = \begin{bmatrix} 1 & 2 \\ 1 & 0 \end{bmatrix} \quad \mathbf{B}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix} \quad \mathbf{H}_1 = [1 \quad 1] \quad \mathbf{H}_2 = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

For the root nodes the a priori beliefs in the two nodes x_1 and x_2 is given by $x_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$; $x_2 = [1]$, with variances $P_1 = \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix}$; $P_2 = [1]$, respectively. These numbers have been chosen to illustrate some of the issues in carrying out inference in these networks.

The noise distribution is given by

$$E[w_3 w_3^T] = Q = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$$
$$E[w_4 w_4^T] = R_1 = [1]$$
$$E[w_5 w_5^T] = R_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad 3$$
$$E[w_i w_j^T] = \Theta \quad i \neq j$$

Decentralized Method for Predictive Estimate Without Evidence

1. For x_3

$$\pi(x_3) = N(x_3; B_1P_1B_1^T + B_2P_2B_2^T + Q, B_1\bar{x}_1 + B_2\bar{x}_2)$$

$$\lambda(x_3) = 1$$

$$Bel(x_3) = N\left(x_3; P_3 = \begin{bmatrix} 30 & 9 \\ 9 & 6 \end{bmatrix}, \bar{x}_3 = \begin{bmatrix} 3 \\ 2 \end{bmatrix}\right)$$

2. For x_4

$$\pi(x_4) = N(x_4; H_1P_3H_1^T + R_1, H_1\bar{x}_3)$$

$$\lambda(x_4) = 1$$

$$Bel(x_4) = N(x_4; 55, 5)$$

3. For x_5

$$\pi(x_5) = N(x_5; H_2P_3H_2^T + R_2, H_2\bar{x}_3)$$

$$\lambda(x_5) = 1$$

$$Bel(x_5) = N\left(x_5; \begin{bmatrix} 31 & 39 \\ 39 & 55 \end{bmatrix}, \begin{bmatrix} 3 \\ 5 \end{bmatrix}\right)$$

Centralized Method for Predictive Estimate Without Evidence

Here, the rule for node propagation and absorption is used to update x_3, x_4 and x_5. The results are the same as for the decentralized case. For example the variance for x_3 is

$$E\left[(x_3 - \bar{x}_3)(x_3 - \bar{x}_3)^T\right] = E\left[(B_1\tilde{x}_1 + B_2\tilde{x}_2 + w_3)(B_1\tilde{x}_1 + B_2\tilde{x}_2 + w_3)_T\right]$$
$$= B_1P_1B_1^T + B_2P_2B_2^T + Q$$

Decentralized Method for Diagnostic Estimate with Evidence

In this case the evidence is

$$x_5 = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

1. For x_3

$$\pi(x_3) = N\left(x_3; \begin{bmatrix} 30 & 9 \\ 9 & 6 \end{bmatrix}, \begin{bmatrix} 3 \\ 2 \end{bmatrix}\right)$$

Note that there is no evidence at the child node x_4, due to which we have $m=1$

$$\lambda(x_3) = N\left(x_3;\left[H_2^T R_2^{-1} H_2\right]^{-1},\left[H_2^T R_2^{-1} H_2\right]^{-1}\left[H_2^T R_2^{-1}\bar{x}_5\right]\right) = N\left(z_3; H_2^T R_2^{-1} H_2, H_2^T R_2^{-1}\bar{x}_5\right)$$

$$Bel(x_3) = N\left(x_3;\left[P_\pi^{-1} + H_2^T R_2^{-1} H_2\right]^{-1},\left[P_\pi^{-1} + H_2^T R_2^{-1} H_2\right]^{-1}\left[P_\pi^{-1}\bar{x}_\pi + H_2^T R_2^{-1}\bar{x}_5\right]\right)$$

$$= N\left(x_3;\begin{bmatrix} 0.7011 & -0.4891 \\ -0.4891 & 1.1087 \end{bmatrix},\begin{bmatrix} 4.4022 \\ 1.0217 \end{bmatrix}\right)$$

2. For x_4

$$\pi(x_4) = N\left(x_4; H_1\left[P_\pi^{-1} + H_2^T R_2^{-1} H_2\right]^{-1} H_1^T + R_1, H_1\left[P_\pi^{-1} + H_2^T R_2^{-1} H_2\right]^{-1}\left[P_\pi^{-1}\bar{x}_\pi + H_2 R_2^{-1}\bar{x}_5\right]\right)$$

$$= N\left(x_4; H_1 P_3 H_1^T + R_1, H_1\bar{x}_3\right)$$

$$\lambda(x_4) = 1$$

$$Bel(x_4) = N(x_4; 1.8315, 5.4239)$$

3. For x,

$$\pi(x_1) = N\left(B_1 x_1; B_1 \cdot \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} \cdot B_1^T, B_1 \cdot \begin{bmatrix} 1 \\ 0 \end{bmatrix}\right)$$

$$\lambda(x_1) = N\left(B_1 x_1;\left[\left[H_2^T R_2^{-1} H_2\right]^{-1} + Q + B_2 \cdot 1 \cdot B_2^T\right]^{-1},\left[H_2^T R_2^{-1} H_2\right]^{-1}\left[H_2^T R_2^{-1}\bar{x}_5\right] - B_2 \cdot 1\right)$$

$$Bel(x_1) = N\left(x_1;\begin{bmatrix} 1.8261 & -0.1957 \\ -0.1957 & 1.0924 \end{bmatrix},\begin{bmatrix} 0.3478 \\ 1.1413 \end{bmatrix}\right)$$

Similarly,

4. For $N(x_2; 0.8315, 0.9239)$

b. Evidence at Leaf Node $x_4 = 10$

Note that in this case $\left[H_1^T R_1 H_1\right]^{-1} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}^{-1}$ doesn't exist.

For updating x_3 we use $P_\lambda^{-1} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ $P_\lambda^{-1}\bar{x}_\lambda = \begin{bmatrix} 10 \\ 10 \end{bmatrix}$

$$Bel(x_3) = N\left(x_3;\begin{bmatrix} 2.3455 & -1.6364 \\ -1.6364 & 1.9091 \end{bmatrix},\begin{bmatrix} 6.5455 \\ 3.3636 \end{bmatrix}\right)$$

$$Bel(x_5) = N\left(x_5;\begin{bmatrix} 3.3455 & 0.7091 \\ 0.7091 & 1.9818 \end{bmatrix},\begin{bmatrix} 6.5455 \\ \mathbf{9.9091} \end{bmatrix}\right)$$

$$Bel(x_1) = N\left(x_1;\begin{bmatrix} 2.1818 & -0.8182 \\ -0.8182 & \mathbf{2.1818} \end{bmatrix},\begin{bmatrix} 1.9091 \\ \mathbf{0.9091} \end{bmatrix}\right)$$

$$Bel(x_2) = N(x_2; 0.8364, 1.2727)$$

Centralized Method for Diagnostic Estimate with Evidence

$$x_5 = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

For x_3
Node x_4 is removed from the graph since it is a barren node.
Node x_1, x_2 are propagated into x_3. This corresponds to

$$N(x_3, P_3, \bar{x}_3) = N(x_3; B_1 P_1 B_1^T + B_2 P_2 B_2^T + Q, B_1 \bar{x}_1 + B_2 \bar{x}_2)$$

Lastly, the arc from x_3 to x_5 is reversed this corresponds to updating the belief for x_3

$$E\left[(x_3 - \bar{x}_3)(x_5 - \bar{x}_5)^T\right] = E\left[(x_3 - \bar{x}_3)\left(H_2(x_3 - \bar{x}_3) + w_5\right)^T\right] = P_3 H_2^T$$

$$E\left[(x_5 - \bar{x}_5)(x_5 - \bar{x}_5)^T\right] = E\left[\left(H_2(x_3 - \bar{x}_3) + w_5\right)\left(H_2(x_3 - \bar{x}_3) + w_5\right)^T\right] = H_2 P_3 H_2^T + R_2$$

Therefore,

$$\textit{Update Mean: } \bar{x}_3^{new} = \bar{x}_3 + P_3 H_2^T \left[H_2 P_3 H_2^T + R_2\right]^{-1} \left(\begin{bmatrix} 5 \\ 5 \end{bmatrix} - H_2 \bar{x}_3 \right)$$

$$\textit{Variance: } P_3^{new} = P_3 - P_3 H_2^T \left[H_2 P_3 H_2^T + R_2\right]^{-1} H_2 P_3$$

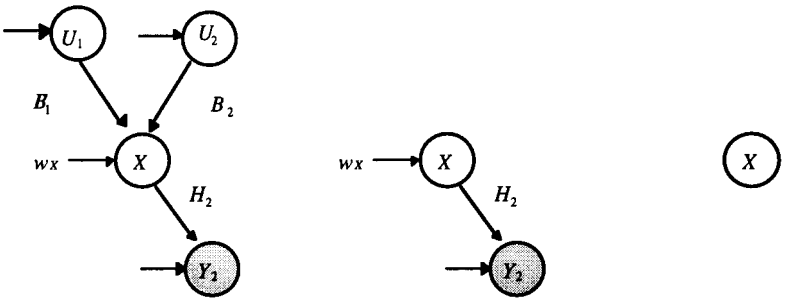


Fig. 3.2-4: The topology transformation to update x

For x_4
The new belief in x_3 is propagated to update x_4

$$N(x_4; P_4^{new}, \bar{x}_4^{new}) = N(x_4; H_2 P_3^{new} H_2^T + R_2, H_2 \bar{x}_3^{new})$$

For x,
Barren node x_4 is removed.
Node x_3 is absorbed. This corresponds to

$$x_5 = H_2 x_3 + w_5 = H_2(B_1 x_1 + B_2 x_2 + w_3) + w_5$$

Node x_2 is propagated. This just means that we will use the a priori mean value for x_2

$$x_5 = H_2\left(B_1 x_1 + B_2 \bar{x}_2 + w_3\right) + w_5$$

The arc from x_1 to x_5 is reversed and node x_5 is propagated

$$E\left[(x_1 - \bar{x}_1)(x_5 - \bar{x}_5)^T\right] = E\left[(x_1 - \bar{x}_1)\left(H_2\left(B_1(x_1 - \bar{x}_1) + B_2(x_2 - \bar{x}_2) + w_3\right) + w_5\right)^T\right] = P_1 B_1^T H_2^T$$

$$E\left[(x_5 - \bar{x}_5)(x_5 - \bar{x}_5)^T\right] = E\left[\left(H_2\left(B_1(x_1 - \bar{x}_1) + B_2(x_2 - \bar{x}_2) + w_3\right) + w_5\right)\left(H_2\left(B_1(x_1 - \bar{x}_1) + B_2(x_2 - \bar{x}_2) + w_3\right) + w_5\right)^T\right]$$

$$= H_2\left[B_1 P_1 B_1^T + B_2 P_2 B_2^T + Q\right] H_2^T + R_2$$

Hence, the mean and the variance is updated by

$$\textbf{\textit{Mean:}}\ \bar{x}_1^{new} = \bar{x}_1 + P_1 B_1^T H_2^T \left[H_2\left[B_1 P_1 B_1^T + B_2 P_2 B_2^T + Q\right] H_2^T + R_2\right]^{-1} \left(\begin{bmatrix} 5 \\ 5 \end{bmatrix} - H_2\left(B_1 \bar{x}_1 + B_2 \bar{x}_2\right)\right)$$

$$\textbf{\textit{Variance:}}\ P_1^{new} = P_1 - P_1 B_1^T H_2^T \left[H_2\left[B_1 P_1 B_1^T + B_2 P_2 B_2^T + Q\right] H_2^T + R_2\right]^{-1} H_2 B_1 P_1$$

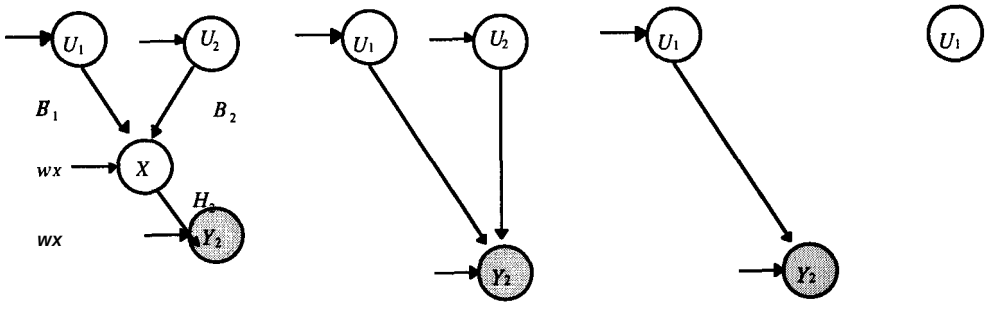


Fig. 3.2-5: The topology transformation to update

The process to update x_2 is similar to that for updating x_1.

Decentralized Method for Predictive and Diagnostic Estimation

$$x_2 = 0,\ x_4 = 10 \text{ and } x_5 = \begin{bmatrix} 5 \\ 5 \end{bmatrix}$$

1. Update for x_3

$$\pi(x_3) = N\left(x_3; \begin{bmatrix} 26 & 7 \\ 7 & 5 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

$$\lambda(x_3) = N\left(x_3; \left[H_1^T R_1^{-1} H_1 + H_2^T R_2^{-1} H_2\right]^{-1}, \left[H_1^T R_1^{-1} H_1 + H_2^T R_2^{-1} H_2\right]^{-1}\left[H_1^T R_2^{-1} \bar{x}_4 + H_2^T R_2^{-1} \bar{x}_5\right]\right)$$

$$\lambda(x_3) = N\left(x_3; \begin{bmatrix} 1 & -1 \\ -1 & 1.5 \end{bmatrix}, \begin{bmatrix} 5 \\ 2.5 \end{bmatrix}\right)$$

$$Bel(x_3) = N\left(x_3; \begin{bmatrix} 0.6738 & -0.5556 \\ -0.5556 & 0.8889 \end{bmatrix}, \begin{bmatrix} 4.9964 \\ 2.4444 \end{bmatrix}\right)$$

$$P_\lambda^{-1} = \begin{bmatrix} 0.7333 & 0.6667 \\ 0.6667 & 1.3333 \end{bmatrix}; \quad P_\lambda^{-1}\bar{x}_\lambda = \begin{bmatrix} 2.6671 \\ 3.3333 \end{bmatrix}$$

$$Bel(x_1) = N\left(x_1; \begin{bmatrix} 1.2903 & -0.4839 \\ -0.4839 & 0.8065 \end{bmatrix}, \begin{bmatrix} 2.2043 \\ 1.21511) \end{bmatrix}\right)$$

2. Update for x_1
Here we have

$$\bar{y} = [10 \quad 5 \quad 5]^T \quad H = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix}^T \quad R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Bel(x_1) = N\left(\begin{matrix} x_1; P_1 - P_1 B_1^T H^T \left[H\left(Q + B_1 P_1 B_1^T\right)H^T + R\right]^{-1} H B_1 P_1, \\ \bar{x}_1 + P_1 B_1^T H^T \left[H\left(Q + B_1 P_1 B_1^T\right)H^T + R\right]^{-1}\left(\bar{y} - H\left(B_1\bar{x}_1 + B_2 \cdot 0\right)\right) \end{matrix}\right|$$

$$Bel(x_1) = N\left(x_1; \begin{bmatrix} 1.2903 & -0.4839 \\ -0.4839 & 0.8065 \end{bmatrix}, \begin{bmatrix} 2.2043 \\ 1.21511) \end{bmatrix}\right)$$

Centralized Method for Predictive and Diagnostic Estimation

1. Update for x_3
First the nodes x_1 and x_2 are propagated, this corresponds to obtaining the prior probability distribution for x_3. The belief for x_3 is

$$N(x_3, P_3, \bar{x}_3) = N\left(x_3; B_1 P_1 B_1^T + Q, B_1\bar{x}_1 + B_2 \cdot 0\right)$$

$$N(x_3, P_3, \bar{x}_3) = N\left(x_3; \begin{bmatrix} 26 & 7 \\ 7 & 5 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

Next the arc from x_3 to x_4 is reversed.

$$Update\ Mean: \ \bar{x}_3^{new} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + P_3 H_1^T \left[H_1 P_3 H_1^T + R_1\right]^{-1}\left(10 - H_1\begin{bmatrix} 1 \\ 1 \end{bmatrix}\right)$$

$$Variance: \ P_3^{new} = P_3 - P_3 H_1^T \left[H_1 P_3 H_1^T + R_1\right]^{-1} H_1 P_3$$

$$P_3^{new} = \begin{bmatrix} & \end{bmatrix}; \quad \bar{x}_3 = \begin{bmatrix} 6.7391 \\ 3.0870 \end{bmatrix}$$

Similarly, the arc from x_3 to x_5 is reversed.

$$N(x_3, P_3, \bar{x}_3) = N\left(x_3; \begin{bmatrix} 0.6738 & -0.5556 \\ -0.5556 & 0.8889 \end{bmatrix}, \begin{bmatrix} 4.9964 \\ 2.4444 \end{bmatrix}\right)$$

Fig. 3.2-6 shows the topology transformation for this case.

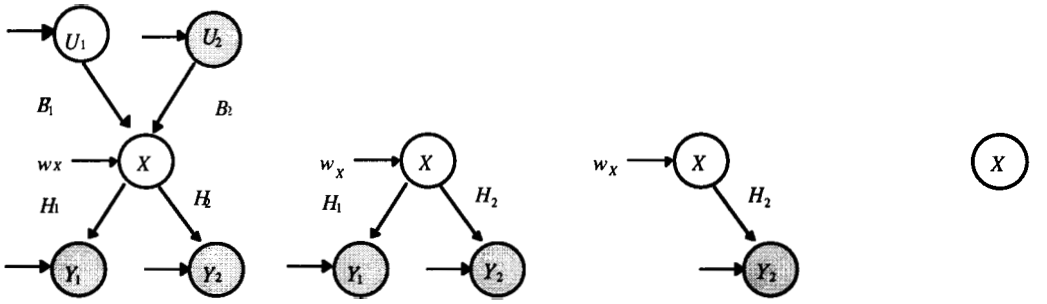


Fig. 3.2-6: The topology transformation for x

For updating the belief in x_1 we

1. Propagate the evidence node x_2, i.e., substitute the evidence value for x_2.

2. Absorb node x_3, i.e.,

$$y_1 = H_1 B_1 x_1 + H_1 B_2 \bar{x}_2 + w_4' \qquad R_4' = R_4 + H_1 Q H_1^T$$

$$y_2 = H_2 B_1 x_1 + H_2 B_2 \bar{x}_2 + w_5' \qquad R_5' = R_5 + H_2 Q H_2^T$$

3. Combine

$$y = \begin{bmatrix} H_1 \\ H_2 \end{bmatrix} \left(B_1 x_1 + B_2 \bar{x}_2 \right) + \begin{bmatrix} w_4' \\ w_5' \end{bmatrix}$$

4. Reverse the arc

The various topology transformation of the belief network is shown in Fig. **3.2-7,** where the shaded node implies that the node is an evidence node.

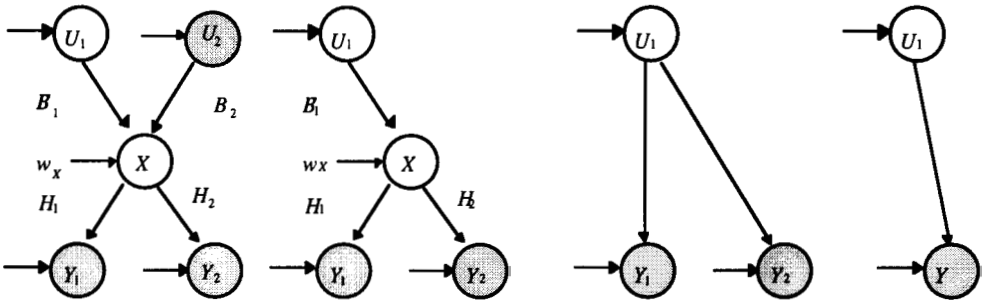


Fig. 3.2-7: The topology transformation for y

Summary

In this section, we have extended Gaussian networks to its vector form, where a node represents a multivariate Gaussian distribution. These networks can represent the uncertainty inherent in the estimation process for dynamical systems. We have developed rules for inference in these networks using both the method of message propagation and topology transformation. The two different methods for inference lead to development of algorithms which can be implemented with either multi-processors (in a parallel or a decentralized manner) or with a single processor (centralized manner). Next, we will apply these inference rules to develop algorithms for monitoring and diagnosis of complex systems. Using the network structure and the rules for

inference developed here, important algorithms such as a Kalman filter, probabilistic data association filter, interacting multiple model algorithm can be represented and derived.

3.3 Methodology for Intelligent Sensor Measurement Validation, Fusion, and Sensor Fault Detection for Generic Processes

In monitoring and diagnostic systems, it is very important to validate sensor data in order to distinguish between a sensor failure and a system failure. A comprehensive methodology is developed in this section for monitoring complex systems. This methodology validates the sensor data, associates a degree of validity with each measurement, isolates faulty sensors, estimates the actual values despite faulty measurements, and detects incipient sensor failures. The methodology consists of four steps: redundancy creation, state prediction, sensor measurement validation and fusion, and fault detection through residue change detection. Through these four steps we use the information that can be obtained by looking at information from a sensor individually, information from the sensor as part of a group of sensors, and the immediate history of the process that is being monitored. The advantage of this methodology is that it can detect multiple sensor failures, abrupt as well as incipient. It can also detect subtle sensor failures such as drift in calibration and degradation of the sensor. Incipient faults can be forecasted by using probabilistic reasoning.

The approach presented here differs from the one above in that here we generate and monitor residue signals. Changes in the properties of these residue signals is a symptom of an abnormal condition in the sensor. The corresponding VDBN is shown in Fig. 3.3-1 and is equivalent to a Kalman filter with multiple estimates. The algorithms in above assumed that the faulty state could be modeled. Then the process of fault detection corresponds to determining if the system has changed from a normal to the modeled faulty state. In this section, we are interested in any deviation of the system from its normal state.

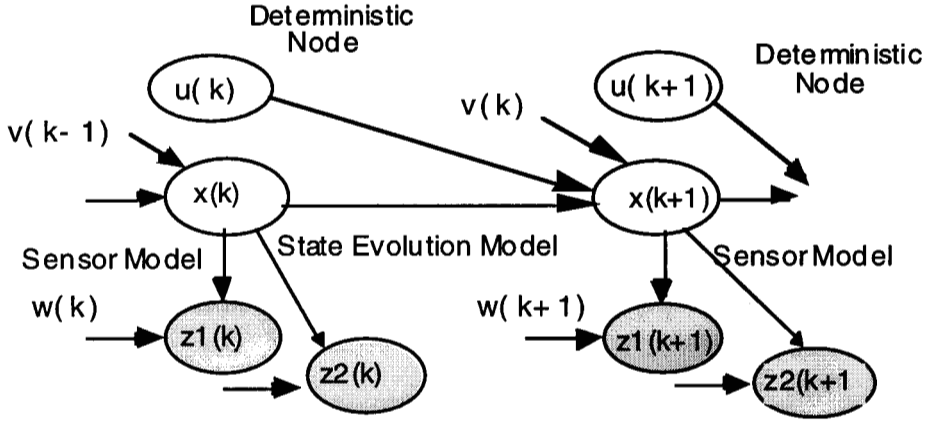


Fig. 3.3-1: The VDBN for the change detection process. This is equivalent to a Kalman filter with multiple estimates

In this section, we also illustrate how probabilistic reasoning can be used for change detection and for forecasting potential incipient failures. We describe methods for constructing, refining, estimating the conditional probabilities between variables (both analytically and learning from data), and performing inference with this representation of temporal probabilistic knowledge. As explained above, dynamic probabilistic networks (DPN) are a species of belief network designed to model stochastic temporal processes. DPN representation extends static belief-network models to more general dynamic forecasting models. We apply the method developed above for on-line learning of the temporal probabilities to adapt the temporal CPT entries which helps the system to adapt itself to changes in its environment. By monitoring the residue using DPNs we are able to predict the degradation of the monitored system (by forecasting) and its eventual failure. The

proposed methodology has a number of advantages over conventional statistical methods for change detection such as thresholds and the SPRT/CUMSUM tests and over fuzzy thresholds.

Problem Statement

In automated vehicles, aircraft, and other complex systems, a large number of sensors are used for monitoring and control. Monitoring helps the operator in performing supervisory control tasks. A monitoring system receives information about the system through measuring devices, i.e., sensors, and makes it available to the operator. The monitoring system usually includes automated diagnosis as a further aid to the operator. The diagnosis system uses sensor readings to assess the state of the system, detect abnormal states, and identify the root cause of the abnormal state in order to advise the operator about corrective actions to prevent significant damage to the system.

The safety, reliability, and performance of complex systems with many sensors are largely dependent on the accuracy and reliability of the sensors. Sensor readings play a key role in assessments of the system state. Where the sensors are less reliable than the systems being monitored, the indication of an abnormal state may be the result of a sensor failure rather than a system failure. Failure to identify the source of the indication of an "abnormal state'' and take appropriate corrective action could result in expensive and unnecessary system shutdowns or, worse still, accidents that endanger both system and system personnel. Thus, it is very important for a monitoring and diagnostic system that is critical to operator decision-making to distinguish between the case where a sensor failure and not a system fault is responsible for the indication of an abnormal state. For that reason, it is essential that there be a methodology that can validate the sensor data, associate a degree of validity with each sensor measurement, isolate faulty sensors, estimate actual values despite faulty measurements, and detect incipient sensor failures.

A number of factors make the process of sensor data validation and sensor failure detection difficult. First and most importantly, sensor failures can be masked by normal system maneuvers or deviations (e.g., in a power plant, changes in the operating conditions due to a load change). Subtle sensor failures such as drift are particularly difficult to detect. Second, the imperfect nature of the sensors adds noise to the sensor readings.

Sensors fail or become faulty for many reasons. An abrupt failure can be caused by a power failure, loose or corroded contacts, or flaws or limitations in the data acquisition and processing system. *An* incipient sensor failure such as a drift in the sensor (e.g., caused by deterioration in the sensing element) is more difficult to detect. Although the problem of incipient failures has received little attention, it is extremely important for sensors that provide critical information to monitoring and diagnostics systems. These failures need to be detected and where possible predicted before they have catastrophic consequences.

This section describes a comprehensive methodology for intelligent sensor data validation, fusion, and sensor failure detection. By combining information from many sources, it is possible to decrease the uncertainty and ambiguity inherent in processing the information from a single sensor source. A large number of sensors measuring many variables can collectively achieve a high level of accuracy and reliability. Our methodology exploits the information that can be obtained by looking at information from a sensor individually, information from the sensor as part of a group of sensors, and the immediate history of the process that is being monitored.

Our methodology consists of four steps: (1) Redundancy Creation generates multiple values for the variable that is being estimated; **(2)** State Prediction uses temporal information about the variable estimate for a specified time window to predict the value of the variable being measured at the next sampling point; **(3)** Sensor Data Validation and Fusion determine whether the information for the sensor can be believed, associating a degree of belief in this measurement, and combining

the various redundant estimates and the predicted value to generate a fused value; and **(4)** Fault detection is carried out by generating residue signals and monitoring them for changes. These changes are symptoms of sensor failures.

Each of these steps can be carried out by using a variety of tools, some of which are shown in Fig. 3.3-1. The choice of tool is not critical but would depend on the user's background and preference. Although the efficacy of our methodology is illustrated here by applying it to data from a gas turbine power plant, this approach is applicable to most complex processes.

Overview

The four steps comprising our methodology are shown in Fig. 3.3-2. First, we create redundancy in the sensor readings. Next we use a time-series state prediction model to predict the expected value for each variable. The sensor readings and the redundant estimates are compared to the values predicted by the state prediction model. We then fuse the validated readings into a fused estimate and detect sensor failures by generating residue signals (i.e., differences between the sensor readings and the fused estimates) and monitoring them for changes in their statistical properties.

A sensor reading validation cycle (steps carried out between two sampling points) consists of predicting the value of the variable being estimated, measuring, creating redundant estimates, validating, fusing multiple estimates, and updating the value of the variable being estimated. Abrupt sensor failures are detected through the validation gate (a region based on the expected distribution); incipient sensor failures are detected by monitoring the sensor residues. The basis of this methodology is the systematic use of direct measurements provided by the sensors, the redundant measurements, and the estimated predicted value from the prediction process. The simultaneous checking of values of each variable with the cross-checking of estimates obtained from values from sensors measuring dissimilar process variables (through redundancy creation as detailed in the following section) and with an adaptive prediction process combined through a Kalman filter (i.e., combining distributions) enables the method to detect multiple sensor failures and detect and estimate bias and calibration errors. Changes in the statistical properties of the sensor residue are used to detect faults in the sensor. Since each sensor has its own residue signal, simultaneous multiple sensor failures can be detected.

Sensor Measurement Validation and Fusion

If the sample points are spaced at regular intervals, the following prediction model can be used

$$x(k+1) = \sum_{i=1}^{n_1} a_i \cdot x(k+1-i) = \left[a_1, .., a_{n_1}\right] \cdot \left[x(k), .., x(k-n_1)\right]^T = \theta^T \cdot \phi(k)$$

A simple, first order adaptive Wiener model, (i.e., a model driven by random noise) is a special case of the model where each of the variables, $i=1,..,n$, is of the type:

$x_i(k+1) = a_i \cdot x_i(k) + w_i(k)$ i = 1, .., n

where $x_i(k+1)$ is the variable being estimated at sampling time $k+1$, $w_i(k)$ is the zero-mean random noise driving the process and a_i is the adaptive parameter.

A number of methods can be employed for parameter estimation, such as estimation based on likelihood, Bayesian estimate, least squares estimate, and the minimum mean-square error estimation (Brown et al., 1992). These four methods are equivalent under the assumptions employed by the Kalman filter. The Kalman filter is a form of optimal estimation (in the statistical sense) characterized by recursive (i.e., incremental) evaluation, an internal model of the dynamics of the system being estimated, and a dynamic weighting of incoming evidence with ongoing expectation that produces estimates of the state of the observed system. The ***apriori*** information to the filter is the system dynamics, the noise property of the system and the measurements that can

be estimated from the historic data. For a review of the Kalman filtering process readers are referred to Alag (1996). We illustrate one cycle of our validation and fusion cycle.

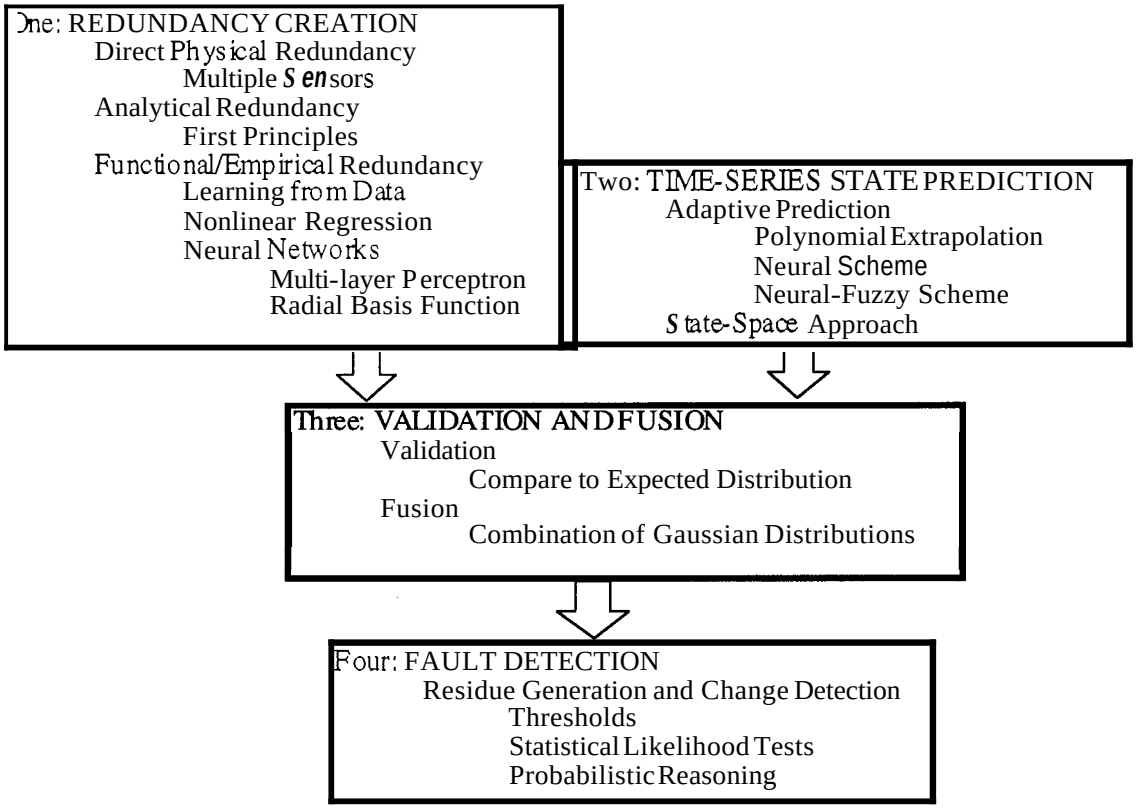


Fig. 3.3-2: Flow chart representation of the four steps of the methodology along with the implementation tools

In our methodology each measurement parameters x is described by

$$x(k+1) = x(k) + u(k) + w(k)$$

$$y(k) = x(k) + v(k)$$

where $w(k)$ is the input noise and $v(k)$ is the measurement noise. The initial state is not known, but the following probabilistic information about $x(0)$, $w(k)$ and $v(k)$ are known. Variables $x(k)$, $w(k)$, and $v(k)$ are independent, random, and Gaussian distributed with

$$E[x(0)] = x_0, \quad E[(x(0) - x_0)(x(0) - x_0)^T] = X_0$$

$$E[w(k)] = 0, \quad E[w(k)w^T(j)] = Q(k)\delta_{kj}$$

$$E[v(k)] = 0, \quad E[v(k)v^T(j)] = R(k)\delta_{kj}$$

$$E[(x(0) - x_0)w^T(k)] = 0, \quad E[(x(0) - x_0)v^T(k)] = 0$$

$$E[w(k)v^T(k)] = 0 \quad \text{for all k and j}$$

where $E[\cdot]$ corresponds to taking the expectation. Q(k) is taken such that $0.5.\Delta x_{max}$ ■$\sqrt{Q}$ ■Δx_{max} where Δx_{max} is the maximum possible change in the variable during the time period.

Initialization

We begin by assuming the measurement variable is normally distributed i.e., $N(x(0); P(0|0) = X, \hat{x}(0|0) = x_0)$

Prediction

We begin at time *k,* where all the sensor readings at time ***k*** have been taken into account. Our estimate for the measurement variable is given by $N(x(k); P(k|k), \hat{x}(k|k))$. Using the state transition model we make a prediction for the state of the variable at the next sampling period[3]. This is given by the following distribution for our estimate

$$N(x(k+1); P(k+1|k), \hat{x}(k+1|k)) = N(x(k+1); P(k|k) + Q, \hat{x}(k|k) + \hat{\theta}^T(k)\phi(k))$$

Fig. **3.3-3** shows the VDBN during the prediction stage.

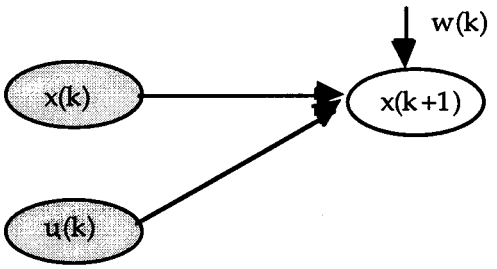


Fig. 3.3-3: The VDBN at the prediction stage

Fig. **3.3-4** shows the process of prediction. Note that the uncertainty increases due to the uncertainty in the system model. This uncertainty is also a function of the time interval between samples.

[3] If a state space model exists, it can be used to make the prediction.

$$Variable(t+\mathrm{A}) = F\left[Variable(t_0), \cdots, Variable(t), \mathrm{A}\right] + \text{Uncertainty}$$

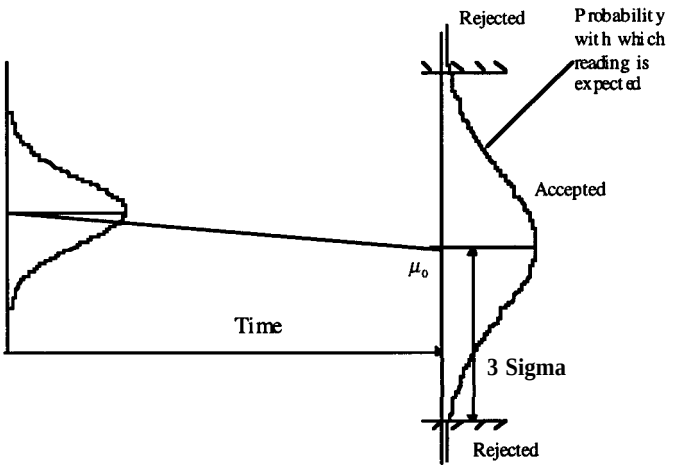


Fig. 3.3-4: The increase in uncertainty during the prediction process

Validation

This is the part of the estimation process where a decision has to be made as to whether the readings from a sensor should be considered for the estimation process. Fig. 3.3-5 shows the corresponding VDBN during the validation process. For this purpose we first calculate the expected distribution for the ith sensor readings. This is given by

$$N\left(y_i(k+1); S_i(k+1), \hat{y}_i(k+1)\right) = N\left(y_i(k+1); P(k+1|k) + R_i, \hat{x}_i(k+1|k)\right)$$

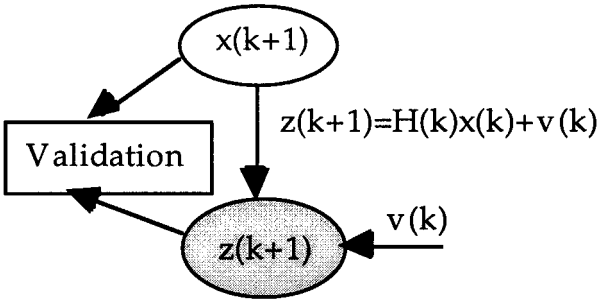


Fig. 3.3-5: The decision-theoretic problem: whether to accept the reading for the fusion process

To validate the sensor readings we use the principle of validation gate (Bar-Shalom and Fortmann, 1988; Kim, 1992; Alag and Agogino, 1995). As shown in Fig. **3.3-6** we define a region where there **is** a high probability that the readings will lie.

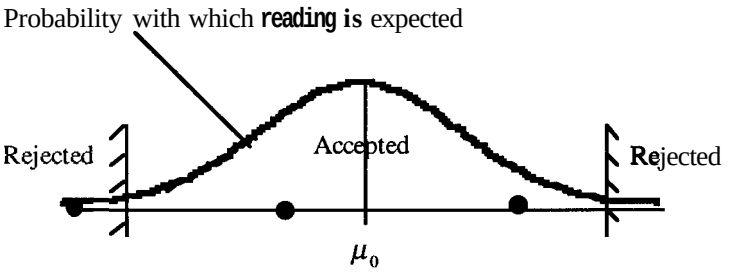


Fig. 3.3-6: Region where the reading is expected as defined by the validation process

$$\text{Let, } v_i \stackrel{\Delta}{=} \left(y_i(k+1) - \hat{y}_i(k+1)\right)$$

Based on this expected distribution a region can be defined in the measurement space where there is a high probability that the measurement will be found (e.g., a **3** sigma bound corresponds to a confidence of 99.8%)

$$\tilde{V}_{k+1}(\gamma) \stackrel{\Delta}{=} \left\{ y_i : \left[y_i(k+1) - \hat{y}(k+1)\right] S^{-1}(k+1) \left[y_i(k+1) - \hat{y}_i(k+1)\right] \le \gamma \right\}.$$

The region $\tilde{V}_{k+1}(\gamma)$ is called the validation region or the validation gate. It is the ellipse (or ellipsoid for multi-dimension state vector) of probability concentration, the region of minimum volume that contains a given probability of mass under the Gaussian assumption. Measurements that lie within the gate are considered valid; those outside are too far from the expected, and are therefore discarded.

Fusion

Readers are referred to Alag (1996) for more details on sensor fusion. **As** stated there in the presence of Gaussian noise the Kalman filter is the optimal filter. There are two main ways by which the validated readings can be fused using a Kalman filter: either sequentially or simultaneously. For linear systems both versions are the same and optimal.

Sequential Update

Fig. **3.3-7** shows the topology transformation during the sequential update process.
Let, m be the number of valid measurements,
Begin with

$$\text{Mean: } E\left[x(k+1 | y_1(k+1))\right] = \hat{x}(k+1|k) + W_1(k+1)\left(y_1(k+1) - \hat{y}_1(k+1|k)\right)$$

$$W_1(k+1) = P(k+1|k) S_1^{-1}(k+1)$$

$$\text{Variance: } P(k+1 | y_1(k+1)) = (1 - W_1(k+1)) \cdot P(k+1|k)$$

Perform the following loop for each sensor. For the ith sensor we have

$$\textit{Mean: } E\left[x(k+1|y_1(k+1),..,y_i(k+1))\right] = \hat{x}(k+1|y_1(k+1),..,y_{i-1}(k+1)) +$$

$$W_i(k+1)\left(y_1(k+1) - \hat{y}_1(k+1|y_1(k+1),..,y_{i-1}(k+1))\right)$$

where

$$\hat{y}_1(k+1|y_1(k+1),..,y_{i-1}(k+1)) = \hat{x}(k+1|y_1(k+1),..,y_{i-1}(k+1))$$

$$W_i(k+1) = P(k+1|y_1(k+1),..,y_{i-1}(k+1))S_i^{-1}(k+1|y_1(k+1),..,y_{i-1}(k+1))$$

$$S_i(k+1|y_1(k+1),..,y_{i-1}(k+1)) = P(k+1|y_1(k+1),..,y_{i-1}(k+1)) + R_i$$

$$\textit{Variance: } P(k+1|y_1(k+1),..,y_i(k+1)) = (1 - W_i(k+1|y_1(k+1),..,y_{i-1}(k+1)))\cdot P(k+1|y_1(k+1),..,y_{i-1}(k+1))$$

Finally,

$$N(x(k+1);P(k+1|k+1),\hat{x}(k+1|k+1))$$

$$= N(x(k+1);P(k+1|y_1(k+1),..,y_m(k+1)),\hat{x}(k+1|y_1(k+1),..,y_m(k+1)))$$

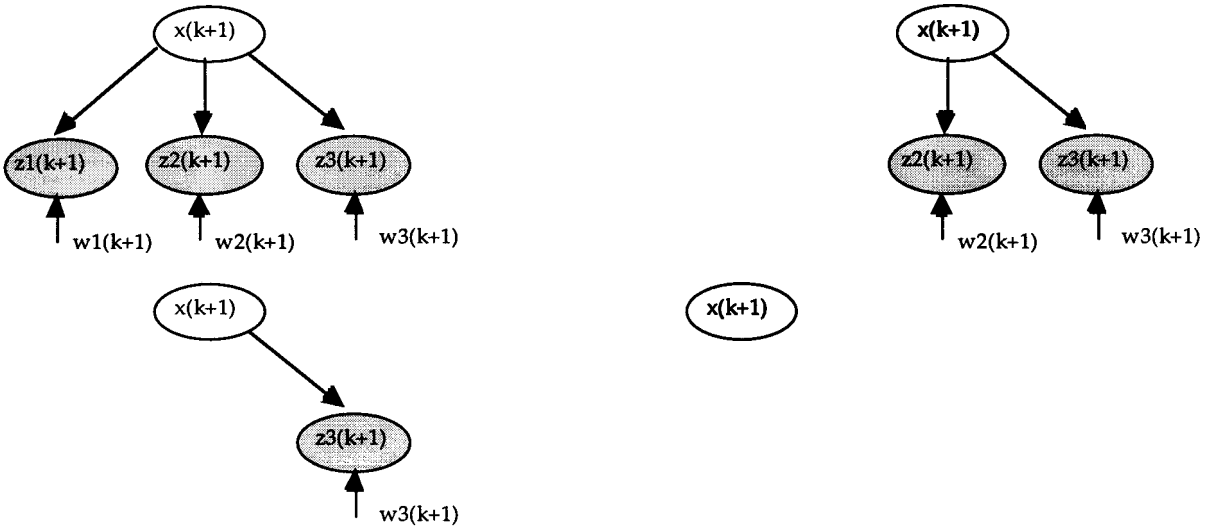


Fig. 3.3-7.: The topology transformation during the sequential fusion process

Simultaneous update using all the evidence at once

$$N(x(k+1);P(k+1|k+1),\hat{x}(k+1|k+1))$$

$$= N\left(x(k+1);\left[P^{-1}(k+1|k) + \sum_{j=1}^{m} R_j^{-1}\right]^{-1}, P^{-1}(k+1|k+1)\left(P^{-1}(k+1|k)\hat{x}(k+1|k) + \sum_{j=1}^{m} R_j^{-1}y(k+1)\right)\right)$$

This completes one cycle of the validation and fusion of the sensor readings.

Fig. **3.3-8** and Fig. **3.3-9** are examples **of** predicted distribution which is combined with the incoming evidence in the form of sensor readings for the single and multiple sensor case, respectively. **As** can be seen the smaller the variance the higher the belief in the reading.

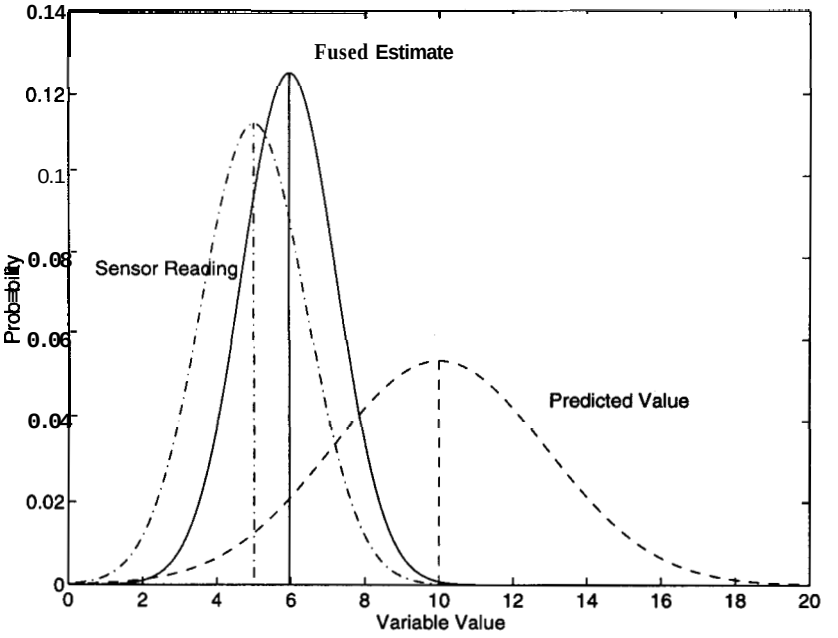


Fig. 3.3-8: Example of the combination of a predicted distribution with the sensor reading.

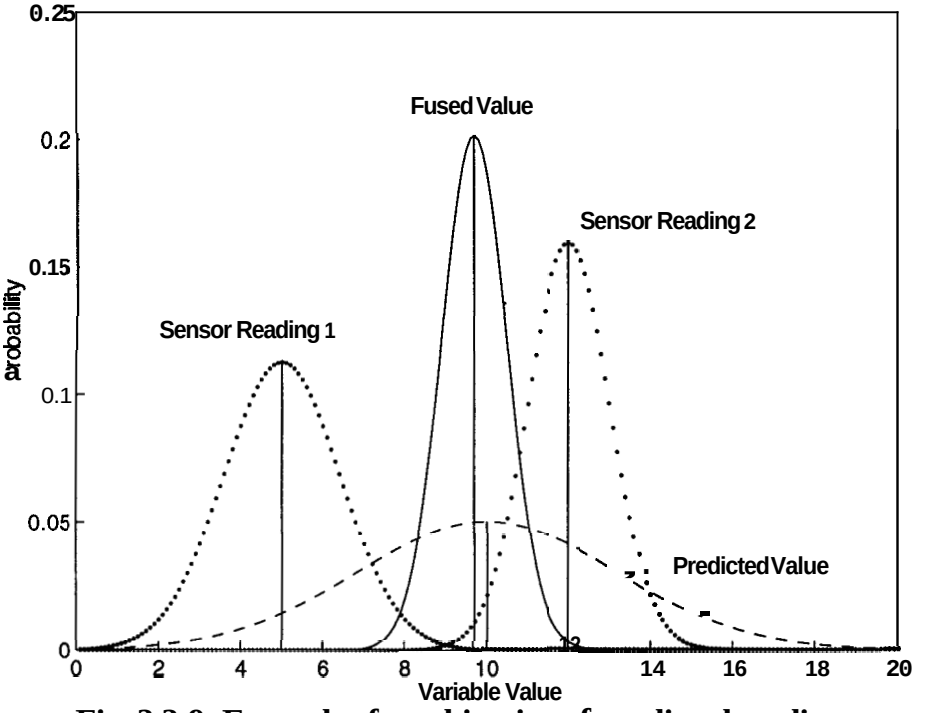


Fig. 3.3-9: Example of combination of predicted reading with multiple sensor distribution

Sensor Confidence

One way to quantify sensor confidence is to calculate the similarity of the sensor reading to the fused distribution for the variable. Normally, sensor confidence measures are given in terms of a

metric (distance) between two probability density functions. The numeric value of sensor confidence represents the closeness between two distributions. We use a scale from *0* to 1, where *0* implies completely independent distributions, while 1 implies identical distributions. A number of distance measures have been suggested, most of which are based on the likelihood ratio. Jefferys' Divergence Measure (Jefferys, 1946) and Bhattacharyya's Coefficient (Bhattacharyya, 1943) are based on the likelihood ratio between two conditional density functions. The Kolmorgov Variational Distance, the Matusita Distance, the Kolmogorov-Smirnov Distance, and the Tie Statistic have also been proposed. We use the Bhattacharyya coefficient because of its computational efficiency and empirical accuracy. As a distance measure, it has the desirable property that it decreases or increases with the probability of error as defined by the Kolmogorov Distance (Kailath, 1967). Using the metric developed in Kim and Agogino (1991), we calculate sensor confidence by the following formula :

$$\text{Sensor Confidence}_i(k) = \sqrt[4]{2 \cdot P(k|k) \cdot R_i} \left(\sqrt{P(k|k) + R_i} \right)^{-1} \exp\left(- \frac{\left(\hat{x}(k|k) - y_i(k) \right)^2}{4 \cdot \left(P(k|k) + R_i \right)} \right)$$

3.4 Sensor Failure Detection

For each variable, the fused estimate for the variable is used to generate residues, i.e., the differences between the fused estimate and the sensor readings. The statistical properties of these residues are then used to detect failed sensors. Since each sensor has its own residue, it is possible to detect multiple simultaneous sensor failures. Ideally the residues for the sensors should have a zero mean and a variance equal to the variance when the sensor is functioning normally. Their deviation from zero is a combined effect of noise and faults. The process of fault isolation, identifies the type of faults that have occurred. The process consists of matching the symptoms to the causes, i.e., matching features from the sensor residues to the faults.

Sensor failures take a number of forms. Most failures manifest themselves as changes in the mean and variance of the sensor residues. A change in the mean of the residue is a symptom of a bias in the sensor signal, while an increase in the variance of the residue is a symptom of degradation of the sensor. The mean and variance of the residue can be calculated recursively. Each effect can occur independent of the other. Some of the failures in which we are interested include spike failure, stuck sensor, biased sensor, noisy sensor, and drifting sensor (incipient failure).

Thresholds

In general, fault detection and isolation (FDI) methods suffer from a fundamental practical limitation in that the system model on which the process is based is never known exactly. The consequence is that the generated residuals will be non-zero even in the absence of faults. As a result thresholds must be used to detect faults. The disadvantage of this approach is that thresholds not only reduce the sensitivity of the system to faults but may also depend on the magnitude and nature of the system disturbances. Normally, fixed thresholds are used. If a decision signal exceeds the threshold, the occurrence of a fault is assumed. If the decision function remains below the threshold, the monitored process is considered fault free. Choosing too low a threshold increases the rate of false alarms, while choosing too high a threshold increases the time to fault detection (Frank, 1990).

In order to increase the robustness of the decision making process, investigators have tried a number of schemes. These include the use of adaptive thresholds (Clark, 1989), i.e., in some way each threshold becomes a function of measurable quantities, and the use of fuzzy logic for decision making (Frank, 1993, Patton, 1994). In the adaptive threshold approach the residual thresholds

are varied according to the control activities of the process. Frank (1994) adapted the threshold to changes of operational conditions by fuzzy variables and fuzzy rules.

Statistical Tests

Statistical tests determine which of two models is active by updating the ratio between the probabilities of each model being correct at each sampling instant. The sequential probability ratio test (SPRT) is based on this principle, as proposed by Wald (1947). Apart from its simplicity and its optimality property among all sequential tests with given error probabilities, it minimizes the average number of data samples required to reach a decision, if the samples are distributed identically and independently. Since SPRT is a test to determine which of the two models is generating all the observed data, rather than whether there is a change in the regime generating the data, it has to be modified to the cumulative sum (CUMSUM) algorithm (Basseville and Nikiforov, 1993; Keresteciogl u, 1993).

In the SPRT test at each sampling instant, the log likelihood ratio L_k of two hypotheses H_0 and H_1, described by two distinct sets of parameters 8, and θ_1, is calculated for as long as $a < L, c \beta$. At the first instant k when the inequality is violated the test is stopped and a decision is made for H_0 if $L, \leq a$ or for H_1 if $L_k \geq \beta$. For the case where the observations form a sequence of independently and identically distributed random variables, the log likelihood ratio can be written as

$$L_k = L_{k-1} + z_k \quad L_0 = 0$$

$$z_i = \ln \frac{f_1(y(i))}{f_0(y(i))}$$

The thresholds a and β are related to the probability of Type I error ε, (i.e., the probability of deciding in favor of H_1 when H_0 is true) and Type II error ε_1 (the probability of rejecting H_1 when H_1 is true). Variables ε, and ε, are also called the false alarm probability and missed alarm probability. a and β are related to the error probabilities by

$$a 2 \ln \frac{\varepsilon_2}{1-\varepsilon,} \qquad \beta \leq \ln \frac{1-\varepsilon_2}{\varepsilon_1}$$

These relations are valid even if the observations are not independently distributed, as long as SPRT terminates with a probability of one. If the SPRT test is being carried out to detect changes in the mean and assuming that the random variable y is normally distributed with variance σ^2, we obtain

$$z_k = \ln \frac{\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(y(k)-\theta_1)^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi\sigma^2}} \cdot \exp\left(-\frac{(y(k)-\theta_0)^2}{2\sigma^2}\right)} = \frac{1}{2\cdot\sigma^2}\left[(\theta_1^2 - \theta_0^2) - 2\cdot y(k)^2(\theta_1 - \theta_0)\right]$$

The SPRT, which is most suitable for testing two hypotheses against each other, can be adapted to detecting changes that occur at an unknown time. This adaptation leads to the CUMSUM test.

Here, the log likelihood ratio is set back to the negative threshold a when it falls below it. In this case the test statistic is computed recursively as

$$L_k = \max[\alpha, L_{k-1} + z_k] .$$

Change Detection and Incipient Fault Prediction Using Probabilistic Networks

Many fault detection and isolation schemes (FDI) involve the generation of residual signals to monitor the performance of the actuators and the sensors. Ideally these residue signals should have a zero mean and a variance equal to when the sensor **is** functioning normally. The deviation of the residues from the ideal is a combined effect of noise and faults. Ideally, faults within the monitored system will affect its residue signal. It is therefore our aim to detect changes in the residue signals through which faults can be detected.

In general, FDI methods suffer from a fundamental practical limitation in that the system model on which the process is based is never known exactly. Consequently, the generated residuals will be non zero even in the absence of faults. Due to this limitation one is forced to use thresholds to distinguish a fault. However, thresholds not only reduce the sensitivity of the system to faults, but also vary with the input signal of the original system and also depend on the magnitude and nature of the system disturbances. Usually, fixed thresholds are used. If a decision signal exceeds the thresholds, the occurrence of a fault is assumed. If on the other hand the decision function remains below the threshold, the monitored process is considered fault free. Choosing the threshold too low increases the rate of false alarms, while choosing it too large increases the time to fault detection (Frank, 1990).

In order to increase the robustness of the decision making process investigators have tried a number of schemes. These include the use of adaptive thresholds (Clark, 1989), i.e., each threshold becomes a function in some way of measurable quantities, and the use of fuzzy logic for decision making (Frank, 1993; Patton, 1994). In the adaptive threshold approach the residual thresholds are varied according to the control activities of the process. An intuitive approach to adapt the threshold to the input was proposed by Clark (1989). The analytical solution to this problem was delivered by Emani-Naeini (1988), and in generalized form by Ding and Frank (1991). Frank (1994) adapts the threshold to changes in operational conditions by fuzzy variables and fuzzy rules. Similar approaches were proposed independently by Sauter et al. (1993) and Schneider and Frank (1993).

We use probability theory (*belief networks)* to represent the uncertainty in the residual processing process. The proposed methodology has a number of advantages over other techniques. This methodology explicitly models time within the process due to which it is capable of forecasting incipient faults. It combines the advantages of using fuzzy thresholds: modeling the effect of maneuvers in the decision making process and of the SPRT/CUMSUM method: using previous history to make a decision. It has been argued that probability theory provides the most complete and consistent framework for dealing with uncertain knowledge (Pearl, 1988; Neapolitan, 1990). The main advantage of belief networks is that once probability values are chosen, their combination in the belief network is rigorous and no additional uncertainties are introduced by inadequate structuring of rules, certainty factor calculi, or ad hoc handling of missing evidence (Rojas-Guzman and Kramer, 1993).

In this section,

- We propose the use of probabilistic reasoning for the management of uncertainty that is inherent in the residue processing process. We illustrate how belief networks can be constructed and refined considering both exogenous and endogenous changes[4].
- We illustrate how conditional probabilities can be estimated analytically and refined by learning from data. We present a new method for learning these CPT entries, which makes use of the special structure of the network. Hence, by using probabilistic networks we can combine expert knowledge (analytical probabilities and structure) with knowledge obtained from learning from data.
- We propose the use of dynamic probabilistic networks to model this stochastic temporal process. By monitoring the residues using DPNs we are able to predict (by using probabilistic projection) the degradation and eventual failure of the system being monitored.

Belief Network Representation and Conditional Probabilities for Each Sensor

The two main properties whose changes we are interested in monitoring are the changes in the mean and variance of the residual signal. **A** change in the mean of the signal is a symptom of a bias in the sensor signal, while an increase in the variance of the sensor signal is a symptom for the degradation of the sensor. Each effect can occur independent of the other. In order to construct a belief network with a correct structure, we need to choose parents for each node such that each node is conditionally independent of its predecessors given its parents. Normally, if one tries to build a diagnostic model with links from symptoms to causes, one has to specify additional dependencies between otherwise independent causes. However, by using a causal model we require far fewer CPT entries which are easy to estimate (Russell and Norvig, 1995).

Using this we build the belief network shown in Fig. 3.4-1. We begin with nodes 'Mean State' and 'Variance State' (numbered 1 and 2, respectively), which correspond to the actual mean and variance of the residual signal. These are to be estimated by processing the residual signal. 'Mean Evidence' and 'Variance Evidence' (nodes 3 and 4, respectively), represent the uncertain information about the mean and variance that is extracted from the residual signal. There are no links between node 1 and node **2** because each can change independent of the other. There, is another node 'Sensor State' (node 5) which summarizes the beliefs in nodes 1 and **2.** During implementation evidence coming in at nodes 3 and 4, is used to update the beliefs at the remaining nodes. Normally, changes in the signal occur over a long period and to reduce the effect of noise a sliding data-window is used (size m). In addition, a distribution over the states is obtained by considering n windows together. For the examples shown later, we have taken $m=12$ and $n=5$. In this example each node has **3** states.

Mean State (X_1) and Mean Evidence (X_3)	Variance State(X_2) and Variance Evidence (X_4)
Negligible $(-0.1, 0.1)$	Negligible(< 0.2)
$Small(-.4 - -.1; .1 - .4)$	$Small(< 0.4 \ \& \ > 0.2)$
$Large(>.4)$	Large(> 0.4)

The sensor is defined to be in the "normal" state when both bias and variance state are in the negligible (also called "normal") state; it is said to be in the "Faulty Marginal" state when either one or both of the "Bias" and "Variance State" are in the "Small" state, else it is said to be in the "Faulty High" state.

[4] ***Exogenous*** changes are those caused by forces external to the system as opposed to ***endogenous*** changes which are generated by forces internal to the system (Hanks et al., 1995).

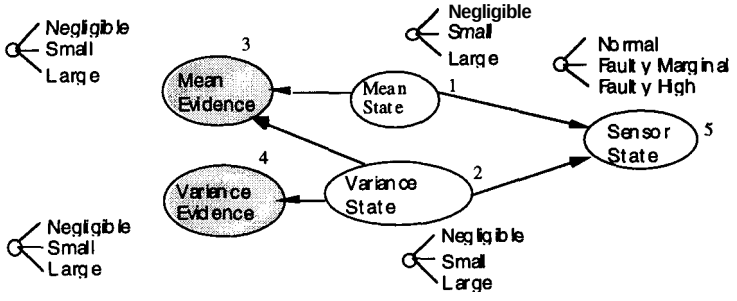


Fig. 3.4-1: The belief network used for fault detection for each of the sensors. The events for each of the states, and numbering of the nodes is also shown.

Next, we need to estimate the conditional probabilities of the various states. From historical data one can estimate the mean μ (ideally zero) and the variance σ^2 of the sensor residue when the sensor is in the normal state. From statistical theory (Kennedy and Neville, 1986), it is known that if n is the size of the window over which observations with mean μ and variance σ^2 is sampled, then the variance for the mean σ_x^2 is given by

$$\sigma_x^2 = \frac{\sigma^2}{n}$$

and the statistic

$$z = \frac{\bar{x} - \mu}{\sigma / \sqrt{n}}$$

where X the sample mean, approaches a normal distribution with mean 0 and variance 1 as n becomes indefinitely large. Using the above formula one can calculate the corresponding conditional probability of the variable X_2 given $X_1 = negligible$ for various values of the "Variance State". CPT tables obtained by this process are shown in Alag (1996).

Similarly, if repeated samples of size n are drawn from a normally distributed population with variance σ^2, the estimated variance s^2, will vary from sample to sample. Its sampling distribution in standardized form $(n-1)s^2/\sigma^2$ is described by the statistic χ^2. The statistic χ^2 is always positive. Again, the conditional probabilities of the state X_1 given the states of X_2 can be calculated by using this distribution. The corresponding conditional probabilities for the states are given later.

The conditional probabilities for the "sensor state" node given the "bias" and "variance state" are obtained by using the definition of the states. It is one whenever, the conditions satisfying the states are fulfilled otherwise it is zero.

Inference Process

During implementation the evidence about each of the sensors comes into the mean evidence" and "variance evidence" states. We need to get the beliefs for the various states for the "mean state' and "variance state". We use the rules for probabilistic inference using the method of topology transformation. We first remove the barren node "Sensor State". We will first update the new beliefs for node 1 followed by node **2.**

Belief Update for Node 1

We make use of the rule for arc reversal to reverse the arc from X_1 to X_2. An arc from state node x to state node y can be reversed, provided x does not have more than one path to y (so as not to

cause a cycle). On reversal of the arc from x to y, node y inherits all the direct predecessors of node x and vice versa (Olmstead, 1984). This corresponds to

$$\Pr(X_2|X_4) = \frac{\Pr(X_4|X_2)\cdot\Pr(X_2)}{\sum_{y\in\Omega_{X_2}}\Pr(X_4|X_2=y)\cdot\Pr(X_2=y)}$$

where Ω_{X_2} is the sample space of X_2, and the probabilities at the previous inference process is taken for $\Pr(X_i)$. The belief at node 4 is propagated to node **2** and node **4** is deleted. This corresponds to

$$\Pr(X_2|\xi_4) = \sum_{\gamma\in\Omega_4}\Pr(X_2|X_4=y)\Pr(X_4=y|\xi_4)$$

Next, node **2** is absorbed into node 3, and node **2** is deleted which corresponds to

$$\Pr(X_3|X_1,\xi_4) = \sum_{y\in\Omega_{X_2}}\Pr(X_3|X_1,X_2=y)\Pr(X_2=y|\xi_4)$$

The arc from node 1 to 3 is reversed, which corresponds to

$$\Pr(X_1|X_3,\xi_4) = \frac{\Pr(X_3|X_1,\xi_4)\Pr(X_1)}{\sum_{y\in X_1}\Pr(X_3|X_1=y,\xi_4)\Pr(X_1=y)}$$

Lastly, the new evidence at **2** is propagated to get the new beliefs in node 1

$$\Pr(X_1|\xi_3,\xi_4) = \sum_{\gamma\in\Omega_{X_3}}\Pr(X_1|X_3=y,\xi_4)\Pr(X_3=y|\xi_4)$$

Belief Update for Node **2**

We first propagate the beliefs at node 1 (where the beliefs in node 1 before it is updated is used) and then delete. This corresponds to

$$\Pr(X_3|X_2) = \sum_{y\in\Omega_{X_1}}\Pr(X_3|X_1=y,X_2)\Pr(X_1=y)$$

Next, we reverse the arc from node **2** to node 4

$$\Pr(X_2|X_4) = \frac{\Pr(X_4|X_2)\cdot\Pr(X_2)}{\sum_{y\in\Omega_{X_2}}\Pr(X_4|X_2=y)\cdot\Pr(X_2=y)}$$

and the arc from **2** to 3 is reversed, which corresponds to

$$\Pr(X_2|X_3,X_4) = \frac{\Pr(X_3|X_2)\cdot\Pr(X_2|X_4)}{\sum_{y\in\Omega_{X_2}}\Pr(X_3|X_2=y)\cdot\Pr(X_2=y|X_4)}$$

Here, we have the property of d-separation: X_1 d-separates X_2 and X_3.

Definition : Direction-dependent separation (d-separation) (pg. 444, Russell and Norvig, 1995): **A** set of nodes ***E*** d-separates two sets of nodes **X** and Y if every undirected path from a node in X to a node in **Y** is ***blocked*** given E. **A** path is blocked given a set of nodes E if there is a node Z on the path for which one of three conditions holds:

i. Z is in E and Z has one-arrow on the path leading in and one arrow out.

ii. Z is in E and Z has both path arrows leading out.

iii. Neither Z nor any descendant of Z is in E, and both path arrows lead in to Z.

Next, the evidence at nodes **3** and **4** is propagated

$$\Pr\left(X_2|\xi_3,\xi_4\right)=\sum_{y\in\Omega_{X_3}}\sum_{z\in\Omega_{X_4}}\Pr\left(X_2|X_3=y,X_4=z\right)\Pr\left(X_4=z|\xi_4\right)\Pr\left(X_3=y|\xi_3\right)$$

Refining the Belief Network

Structure

We can keep making our belief network structure more and more complex depending upon the degree of accuracy and reliability required. We can identify critical variables and add them to our structure. For example, the residual signal should ideally be pure noise. Hence, testing the signal for whiteness can provide us with additional information. Therefore, in this case we would add two more nodes 'Whiteness State' and 'Whiteness Evidence' to the structure. Other exogenous factors, such as the age and type of the sensor, time since last calibrated, common mode failures among sensors, detection of maneuvers in the plant can also be added. Fig. **3.4-2** shows a more complex belief network. To keep matters simple we will use the belief network in Fig. **3.4-1** for the remaining part of this section.

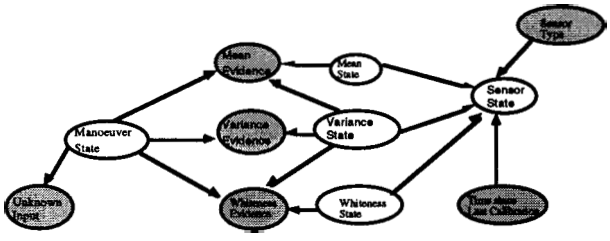


Fig. 3.4-2: A more comprehensive belief network

Conditional Probabilities

The topic of learning in belief networks has received attention only recently. The results that have been developed are new and still evolving, but they have been shown to be remarkably effective in some domains. In the learning process for belief networks we first encode existing knowledge of an expert or experts in the probabilistic network. Next, a database is used to update this knowledge in the form of refining the CPT entries and identifying new distinctions and relationships. Learning in belief networks is similar to that in neural networks, however belief networks have two main advantages. Firstly, it is easy to encode expert knowledge in the form of probabilistic knowledge and use this knowledge to increase the efficiency and accuracy of the learning process. Secondly, the nodes and arcs in a belief network can be easily interpreted as they correspond to recognizable distinctions and causal relationships (Heckerman, 1995; Russell and Norvig, 1995). There are several varieties in which the problem of learning in belief networks comes. The variables in the network can be ***observable*** or hidden. Further, the structure of the network can be known or

unknown. We will be mainly concerned with the case where the structure of the network is known.

Referring to Fig. 3.4-1, we may be interested in refining the CPT entries for the nodes 1 to 4 (node 5 just summarizes the beliefs at node 1 and **2,** and can be removed from the learning process). We can convert this to the simplest case of learning, in which all the nodes are ***Observable,*** i.e., we create a data-set which contains the beliefs in all the nodes. Then since each node is observable, each data case can be pigeonholed into the CPT entries corresponding to the values of the parent variables at each node. The posterior values for the conditional probabilities can then be calculated based on Dirichlet priors (Spiegelhalter et al., 1993; Heckerman, 1995).

However, estimating the beliefs for nodes 1 and **2** involves, using the subjective probabilities of an expert. It is better to just obtain the evidence for nodes 3 and 4, and treat nodes 1 and **2** as unobservable nodes. This case of known structure and hidden variables (in this case nodes 1 and **2** are hidden) is similar to the neural network learning case. Buntine (1994) has suggested the use of generalized network differentiation for learning probabilistic networks with hidden nodes. Neal (1991) derives an expression for the likelihood gradient in sigmoid networks using stochastic simulation, and uses it to show that the Boltzmann machine is a special case of a probabilistic network. Lauritzen (1991) considers the application of the Expectation Maximization (EM) algorithm to solve this case.

More recently, Russell et al. (1995) present an algorithm which they view **as** a special case of EM, where the maximize phase is carried out by a gradient-following method. They assume that each possible setting of the weights is equally likely ***a priori,*** and they use the ***maximum likelihood*** criteria for choosing the weights. In their method to learn the conditional probability table (CPT) entries a local gradient is set up for each CPT entry. To obtain the gradient, an inference algorithm is run separately for each data case. The gradient for changing the weights is the ratio of the probability of obtaining the case representing the CPT entry and the present weight.

To maintain the condition that the CPT entries corresponding to a particular case (an assignment of values of the parents) must sum to 1, the gradient is renormalized to remain on the constraint surface. However, they do not consider the case where a network may contain a root node (node with no parents) which is unobservable[5]. This can be overcome by estimating the ***a priori*** values (subjective probabilities). In case one wants to circumvent the process of estimating the ***a priori*** probabilities one has to do a lot more work. Alag (1996) contains an alternative method that can estimate the ***a priori*** probabilities using the structure of the network. We must however emphasis that this method is useful only for only very small networks.

Comparison with Other Methods

The proposed methodology differs from other methods used for change detection such as thresholds, adaptive thresholds using fuzzy logic, and statistical methods such as SPRT/CUMSUM in that it explicitly models time as one of the variables. Due to which it is capable of forecasting and hence predicting incipient faults before they reach catastrophic limits. Like thresholds and SPRT/CUMSUM we use probability theory to represent uncertainty and for decision making. It uses the advantage of the SPRT/CUMSUM test over thresholds of using the history of the residues in making a decision of a change. Unlike the SPRT/CUMSUM test it can also detect changes in the variance.

---

[5] In their method, ***a priori*** probabilities are required to do probabilistic inference. However, there is no way to change these ***a priori*** probabilities during their learning process for an observable node.

For detecting multiple levels of changes a number of SPRT/CUMSUM tests need to be carried out in parallel. Here, the result of the method is a probability that various change states are true. One can obtain the expected change magnitude by taking the expected value over the states using this probability. It incorporates the advantages of fuzzy-adaptive thresholds in that information about process maneuvers can easily be incorporated in the decision making process by adding nodes to the belief network. In addition no added uncertainty is added during the inference process using the inference rules for a belief network.

Summary

In this chapter, we have developed a comprehensive methodology for intelligent sensor measurement validation, fusion, and sensor fault detection. The methodology is effective in detecting sensor faults since it combines information from a number of sources by looking at information from a sensor individually, information from the sensor as part of a group of sensors, and the immediate history of the variable that is being monitored. A major advantage of this methodology is its ability to detect multiple simultaneous failures. This work addresses the very important problem of distinguishing between a sensor failure and a system failure for complex systems. The proposed four step methodology of redundancy creation, state prediction, validation and fusion, and fault detection can detect subtle sensor failures such as drift in mean and degradation of the sensor over time.

We have also illustrated how probabilistic reasoning can be used to handle the uncertainty inherent in the residue processing process. We have illustrated how to refine and build an appropriate belief network structure and how the conditional probabilities can be estimated both analytically and from learning from data (for which we have presented a new method which is applicable to only a special class of belief networks). We have illustrated how DBNs can be used for forecasting. We have derived a recursive formula for the on-line adaptation of the state evolution model. The combined sensor fault detection methodology is very effective in both detecting and forecasting potential sensor faults.

4. Hazard Analysis
4.1 Introduction

The longitudinal motion controller in the AHS controls maneuvers such as platoon joining and splitting. Sonar and radar sensors are components of the longitudinal motion controller which provide information about the vehicle separation distance between cars or approaching objects. There are certain external conditions which alter the sensor's performance. Research has been conducted to explore environmental condition (e.g., rain, fog, snow) effects on the sensors. The current focus is on analyzing the sensor systems and determining all possible failure modes using Fault Tree Analysis. Knowledge of all possible failure modes and their characteristics can assist in anticipating future failures. Thus, hazardous situations on the highway can be averted. The safety of the Intelligent Vehicle Highway System as a whole will be improved.

The goals of reliability and safety analysis are to reduce the probabilities of failure and the attending human (death, injury, disability), economic (shutdown of highway, loss of vehicles and other equipment), and environmental losses (pollution due to spills from vehicle tanks, burning vehicles). A basic failure event can be an incorrect reading from a sensor. Typical policies to minimize hazards include 1) sensor redundancies; **2)** inspection and maintenance; **3)** protective systems such as bumpers, fences between the lanes; and **4)** alarm displays.

Comparison of Techniques

Method	Characteristics	Advantages	Disadvantages
Preliminary Hazards Analysis	Defines the system hazards and identifies elements for FMEA and fault tree analysis. Overlaps with FMEA and criticality analysis	required first step	none
FMEA	Examines all failure modes of every component. Hardware oriented	Easily understood. Well accepted, standardized approach; non-mathematical,	Examines non-dangerous failures. Time consuming. Often, combinations of failures and human factors not considered
Criticality analysis	Identifies and ranks components for system upgrades	Standardized technique. Easy to apply and understand. non-mathematical	Follows FMEA. Often does not take into account human factors, common cause failures, system interactions
Fault tree analysis	Starts with initiating event and finds the combination of failures which cause it.	Accepted technique. very good for finding failure relationships. Fault oriented: we look for ways system can fail	Large fault trees are difficult tc understand, bear no resemblance to system flow sheet, ad are not mathematically unique. Complex logic involved.
Event tree analysis	Starts with initiating events and examines alternative event sequences.	Can identify *(gross)* effect sequences and alternative consequences of failure.	Fails in case of parallel sequences. Not suitable for detailed analysis
Hazards and operability study	Extended FMEA which includes cause and effect of changes in major plant variables.	Suitable for large chemical plants.	Technique not standardized or well described.
Cause-consequence analysis	Starts at critical event and works forward, using consequence tree; backward, using fault tree.	Extremely flexible. All encompassing. Well documented. Sequential paths clearly shown.	Can become too large very quickly. Similar disadvantages of fault trees

Table 4.1-1: Techniques for hazard analysis

Fault Trees and Decision Tables

Starting from an undesired event, more basic events are found until the basic event is found. Gate symbols connect the event symbols. The former connect events according to their causal relation. There are several types of gate symbols: **AND** gates and OR gates are deterministic while INHIBIT gates express a probabilistic causal relation. Event symbols are described by: 1) the circular basic event, for which the component itself is responsible and the component must be repaired or replace (such as "Valve failure due to wearout"); 2) diamond shaped undeveloped events for which a detailed analysis has not been carried out yet; **3)** rectangular shaped intermediate events; 4) oval conditional events; 5) house shaped events which either do or do not occur.

A cut set is a collection of basic events which – if it occurs – guarantees the top event to occur. **A** path set is a collection of basic events which – if it does not occur – guarantees the top event not to occur. Since there are many cut sets it is desirable to look at general failure modes first. These minimal cut sets are such that if any basic event is removed, the remnants are no longer a cut set. **A** cut set including other sets is not a minimal cut set.

Reliability

The reliability R(t) is expressed **as** the probability of survival to (inclusive) age t and is the number surviving at t divided by the total sample. From this, a failure distribution (or survival distribution = 1-F(t)) can be created which is generally a monotonically increasing function. The curve created through F(t+dt)-F(t) is the failure density f(t). From this the probability for failure between ages t1 and t2 can be obtained by integrating the curve between t1 and t2. The failure rate r(t) is the probability of failure per unit time at age t for the individual in the population r(t)=f(t)/(1-F(t)) which results in the "bathtub" curve. That is, there are a fairly high number of early failures (bum in) followed by a "prime of life" period with lower failure rate (failures occur randomly) which in turn is Followed by a final wearout or bum-out phase with high failure rate.

Incorporating these probability calculations into failure diagrams by attaching probabilities to the transitions between states, one arrives at Markov type influence diagrams as shown in Fig. 4.1-1.

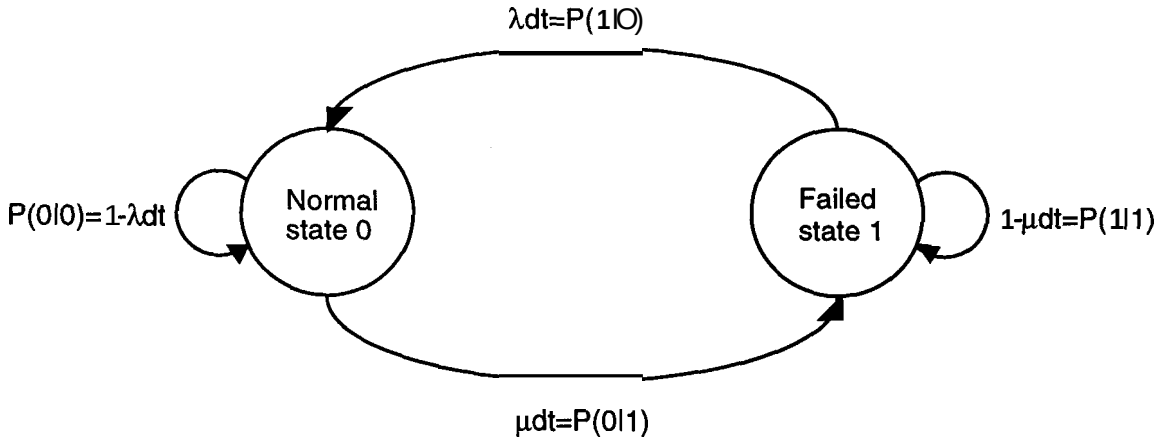


Fig. 4.1-1: Markov transition diagram

where l is the conditional failure intensity, or the probability that the component fails per unit time at time t, given that it is in the normal state at time zero and is normal at time t. Generally $l(t) \neq r(t)$ because the latter assumes the continuation of the normal state to time t, i.e., no failure in the interval [0,t]; they are the same when the component is non-repairable. Variable m is the

conditional repair intensity, or the probability that the component is repaired per unit time at time t, given that it jumped into the normal state at time zero and fails at time t.

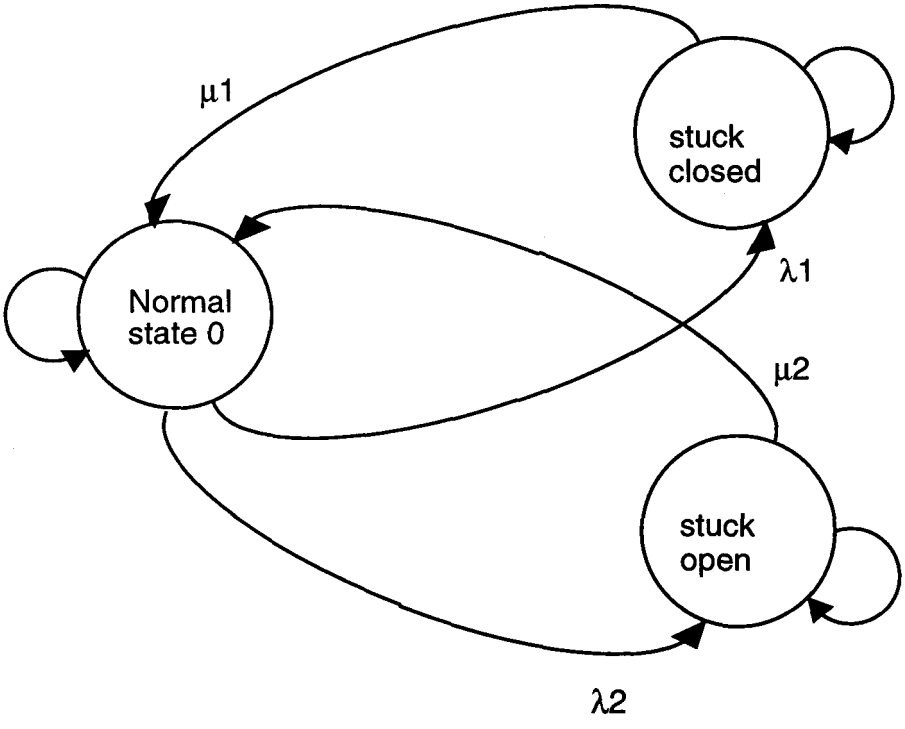


Fig. 4.1-2: Markov transition diagram for the valve

Importance

Importance is essentially a type of sensitivity analysis which describes the contribution of a component or cut set to a particular top event. There are different importance measures such as Birnbaum's Structural Importance (classical sensitivity) and Criticality Importance. The latter considers the fact that it is more difficult to improve the more reliable components than the less reliable components. Criticality Importance is different from the Criticality in the Failure Mode Effect and Criticality Analysis which will be discussed later.

4.2 Hazard Characterization

It is necessary that the Intelligent Vehicle Highway System be at least as safe as the current highway system. Ordinary traffic casualties are 90% associated with some kind of human error (Hitchcock, A., 1992), but in IVHS humans will no longer be in control of their own vehicles. In the IVHS environment where there is such a dependency on sensor performance, **a** sensor component failure can lead to catastrophic situations. It has been hypothesized that in the IVHS the number of accidents will decrease and the number of vehicles involved with accidents will decrease, but the mean number of vehicles involved in an individual accident will increase. Similarly, the number of fatal accidents will decrease and the number of fatalities will decrease, but the mean number of fatalities per fatal accident will increase (Hitchcock, **A.,** 1992).

Hitchcock has completed extensive research in the area of hazard characterization and has defined a hazard as a precursor to a condition in which one further failure could lead to a catastrophe. **A** catastrophe is a high speed collision between platoons where multiple deaths and injuries are likely. In Hitchcock's exhaustive set of hazards that can lead up to catastrophes, he presented the following situations.

A collision is likely to occur:

a) when all platoons involved are under control, automated or manual; in this case vehicles were too close before a final control failure.

> **Hazard 1: A** platoon is separated from one ahead of it, or from a stationary object in its path, by less than platoon spacing.
>
> **Hazard 2: A** vehicle, not under system control is at an unmeasured or unknown distance in front of a platoon.

b) when one platoon is not under control; this will happen if automatic control is switched off before the driver is ready, or not switched on when the driver lets go.

> **Hazard 3: A** vehicle is released to manual control before the driver has given a positive indication that he/she accepts it.
>
> **Hazard 4:** a vehicle is released to manual control at less than manual spacing from the vehicle ahead of it; or at such a relative speed that manual spacing will be realized in less than **2** seconds, or while the brakes are being applied.

c) when the final failure is a failure to brake or to communicate that brakes should be applied.

> **Hazard *5:*** a vehicle under automatic control is in such a condition that if instructed form the infrastructure to brake, it will not do so.

Sensor Function

The sonar and radar sensors are components of the longitudinal distance controller which controls the distance between vehicles in the platoon. The sensors, mounted on the vehicle's grill, measure the distance from a vehicle or an approaching object. Both sensors use the same underlying physical principal to measure the separation distance. They emit a signal which is reflected by a target (target refers to either object or vehicle) within the sensor range; the reflected signal, or echo, is then received by the sensor. The time for the signal to travel to the target and return **as** an echo is recorded. Then the known velocity of the pulse propagation is used to convert the output to distance by a data acquisition system.

Sensor Behavior

Radar and sonar sensors are susceptible to the external conditions under which they operate; conditions such as rain, fog, and humidity have a certain effect on sensor output. The research done by Bellm includes a series of experiments conducted on the longitudinal controllers in sub-optimal environments. Bellm quantitatively characterized the effects of the environment on sensor

output (Table 4.2-1). This information facilitates sensor failure prediction because the symptoms of the failure are known.

	Radar Sensor Output	Sonar Sensor Output
Power Line	constant 2.20m	constant -0.85m
Data Line	constant 4.25m	constant 2.20m
Dirt	mean: 1.5m<x<2.5m	constant Om
Fog	not significantly affected	variance increased by factor of 4
Rain	variance increased by factor of 4	variance increased by facor of 4
		mean increased by 10%
'Plastic	variance increased by 20%	constant Om
Vibration	variance increased by 20%	not significantly affected

Table 4.2-1: Characterization of sonar and radar behavior

4.3 Failure Analysis

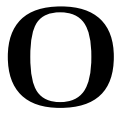
There are several qualitative and quantitative methods established for exploring failure modes and the reasons for which they occur. Fault Tree Analysis provides both the benefits of a qualitative and quantitative analysis through a graphical tool that focuses on a failed state and provides a method for determining the causes of the failed state. After completion of the fault tree, Boolean Algebra can be used to determine the minimum number of component fault combinations that will lead to sensor failure.

The Fault Tree Analysis Method (Vesely, **W.E** et al, **198**1) focuses on failure because a failed state is generally easier to characterize than is a successful state. The basic elements of the Fault Tree method depict the logical interrelationships of basic events that lead to the undesired event (failed state) of the fault tree. There are entities known as "gates" which serve to permit or inhibit the passage of fault logic up the tree. Gates show the relationships of events needed for the occurrence of a "higher" event (outputs to the gate). Gate symbols denote the relationship between the input events and the output event.

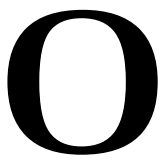
The component fault categories are classified as primary, secondary, and command. A primary fault is any fault of a component that occurs in an environment for which the component is qualified (e.g. defect). A secondary fault is a fault of a component that occurs in an environment for which it has not been qualified. A command fault is a fault where there is proper operation of a component but at the wrong time or in the wrong place. Fault Tree Symbols and their descriptions are illustrated in Figure 4.3-1.

The Fault Tree is complete when all possible occurring events have been exhausted and all the branches terminate with either a basic event or an undeveloped event. Boolean algebra can then be used to quantitatively determine the smallest combination of component failures which lead to system failure.

Primary Events:



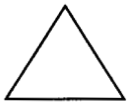
Basic Event: A basic initiating fault requiring no further development.



Undeveloped Event: An event which is not further developed because it is of insufficient consequences or because information is unavailable.

Intermediate Event Symbols:

Intermediate Event: A fault event that occurs because of one or more antecedent causes acting through logic gates.



Transfer in/out: indicates that the tree is either developed further on another page or is the continuation of a tree from a previous page.

Gate Symbols:

OR: Output fault occurs if at least one of the input faults occurs.

AND: Output fault occurs if all of the input faults occur.

Fig. 4.3-1: Basic fault tree symbols (Vesely, W.E, et al, 1981)

Failure analysis was performed on the longitudinal motion controller to predict faults that could later lead to catastrophic failure. Fault Tree Analyses were conducted on sonar and radar sensors. The faults predicted dealt with component failure and operation in sub optimal conditions that could possibly lead to system failure. The results from the fault tree analyses can be used to anticipate further failure and most importantly, hazards. The component faults derived from the trees can be considered symptoms for types of failures that lead to specific hazards.

4.4 Fault Tree Analysis of the Radar Sensor

For the radar sensor the failed state is described as "Radar Sensor output is not close to the true distance." The next step was to determine how to further expand on this failure. Three separate events, connected by the logic gate "OR" re-express the failure: primary fault (defect), secondary fault, and command fault.

The primary fault is defined as a defect internal to the sensor, most likely occurring with the electrical components. The resulting outcome of any electrical component failure is a sensor output equal to the maximum or minimum of that sensor.

The command fault has been defined for this purpose as either a power failure or a data line failure of the longitudinal motion controller. For a data line failure the observable characteristic of the output is approximately constant at 4.25 m and for a power failure the output is around 2.2 m.

The secondary fault can be re-expressed by two events connected by the logic "OR' gate. "Radar Sensor Failure due to environmental clutter" and "Radar Sensor Failure due to range limitations" further describe the intermediate event. "Radar Sensor Failure due to range limitations" is shown as an undeveloped event, where causes of failure cannot be determined due to lack of information of sensor characteristics or observations. The intermediate event of "Radar Sensor Failure due to environmental clutter" can be described by two intermediate events and an undeveloped event. The RSF due to corrosive effects is an attempt to describe the sensor performance near the ocean or any environment where corrosive agents are in the air and can possibly affect the integrity of the electrical components and lead to degradation. This event cannot be described further and is therefore considered an undeveloped event.

The weather and rugged road conditions describe the intermediate event of environmental clutter. They are connected by the logic "OR' gate. In terms of weather conditions, rain, snow and humidity have shown to have adverse affects on sensor performance. Rain can be affect the sensor output in two ways, internally the moisture can cause a power failure and externally the sensor signal can be attenuated by the water drops. Both of these intermediate events can be characterized by basic observations. The moisture can be predicted by a constant output of 2.2m and the attenuation can be predicted by an increase of variance by a factor of four. Snow is the cause of similar results. The moisture from the snow can again internally affect the electrical components and the same prediction can be made (output of 2.2 m). The snow can also reflect the sensor signal in which case the basic observation is that the output mean between 1.5 m and 2.5 m. The humidity in the highway environment may cause internal damage to the sensor in the same manner that the moisture from the rain and snow cause damage.

The road conditions describing radar sensor failure are road clutter, rugged road conditions, and road course. The three events are connected by the logical "OR' gate. The course failure refers to an event on the highway that results in a loss of communication between cars in a platoon, such as a round bend or a pothole in the road where the sensor signal emitted is not aimed at the car ahead of it. The rugged road conditions can result in a vehicle vibration in which case, the observed characteristic in the sensor output would be an increase in variance by 20%. The road clutter can be further described as "Radar Sensor Failure due to debris covering the sensor" and "Radar

Sensor Failure due to debris from the road environment." The two intermediate events are linked by the logical "OR" gate. The debris covering the sensor has an observable effect on the sensor output; the sensor mean is between 1.5 m and 2.5 m. The debris from the road can have any effect on the sensor output and cannot be clearly described and for this reason it has been left as an undeveloped event.

4.5 Fault Tree Analysis of the Sonar Sensor

The sonar sensor fault tree analysis is quite similar. The undesired top event is "Sonar Sensor Failure: the output is not close to the true distance." The three intermediate events, primary (defect), secondary, and command fault further describe the top fault and are logically connected by the "OR" gate.

The primary fault is defined as a defect internal to the sensor, most likely occurring with the electrical components. The resulting outcome of any electrical component failure is a sensor output equal to the maximum or minimum of that sensor.

The secondary fault is associated with a proper operation of a component but in environments for which it was not designed for. The two intermediate events are connected by the logical "OR' gate. Sonar Sensor Failures due to the environment or due to range limitations describe two circumstances for which the sensor was not designed. The observable characteristic of range limitation for the sonar sensor is a default value of 15m when out of range. Similar to the radar sensor, the environmental fault can be further described by weather conditions, road conditions, and corrosive effects. These events are connected by the logical "OR" gate. Corrosive effects again refers to the affects of the electrical components of the sensor in an atmospheric environment where corrosive agents are present. However, further information about the effect on the output of the sensor and thus, this remains an undeveloped event. The weather and road conditions can be defined more completely.

The weather conditions affecting the sonar sensor are rain, snow, humidity, and fog. The moisture in the rain can possibly short out the electrical component in which case the sensor output would show outliers at +2m. Water drops from the rain can also serve to attenuate the signal and the variance in the output would increase by a factor of four and the mean would increase by 10%. Snow has a similar effect. The moisture can affect the sensor internally in which case the output would show outliers at 2m. The snow covering the sensor would result in a constant output of Om. Humidity and fog have the same affect that the moisture has, it may damage the sensor internally and result in an output with outliers at 2 m.

The road conditions which affect the sonar sensor performance are road clutter and course condition. The road clutter can be either debris covering the sensor or debris from the highway environment. The debris covering the sensor would result in a constant output of Om while the characteristics for the sensor signal hitting random debris from the highway environment cannot be determined. The course can cause Sonar Sensor Failure in that a loss of communication would render the output as not the true distance. However, an observable characteristic is not noticed and hence it remains an undeveloped event.

4.6 Linking Faults with Hazards

The completed fault tree provides important information about sensor failure. The faults leading up to the failure have been identified and their correlation to hazards can now be examined. There is not an established method to complete this, however, the fault tree analysis provided a methodical way of thinking that can be similarly applied to each fault.

An example of the flow chart method is shown in Figure 4.6-1. The first event in the flow chart is the fault: "Radar Sensor Signal Attenuated by Rain Drops", the following event is a result from this fault, "Radar Sensor Performance Inhibited", the observable symptom from this fault is, "Radar Sensor Output Increases". As a result of these events the Longitudinal Controller will receive information that is not close to the true measurements which leads to three possible situations, vehicle accelerating too quickly, decelerating too quickly, or not responding quick enough. As a result, the vehicle may be either too close, too far, or at an unknown distance from the vehicle ahead of it. These scenarios point to three possible hazards: 1, 2 and 4.

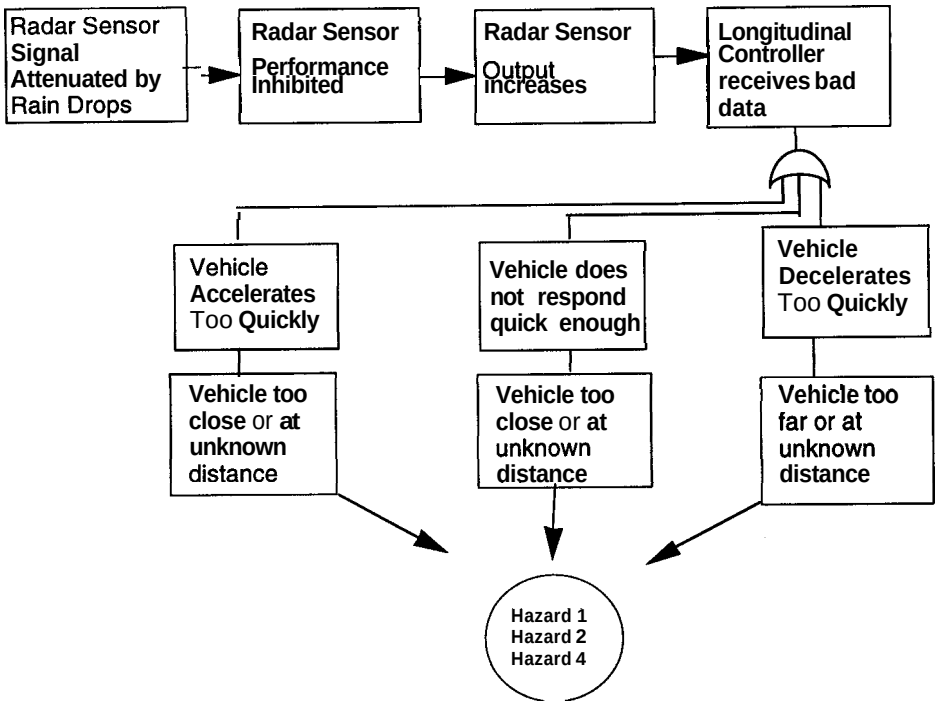


Fig. 4.6-1: Radar sensor hazard flow chart

The same procedure was employed for all the component faults and Tables 4.6-1 and 4.6-2 illustrate the results.

Faults for Radar Sensor	Possible Hazards			
Power Line Down	1	2		4
Data Line Down	1	2		4
Debris Covering Sensor	1	2		4
Debris on Road	1	2		4
Rugged Road Conditions			3	
Course Limitations	1	2		4
Electrical Component Shortage	1	2		4
Attenuation from water drops	1	2		4
Snow reflects sensor signal	1	2		4
Electrical Component failures	1	2		4
Moisture affects performance	1	2		4

Table 4.6-1: Linking radar sensor faults with hazards

Faults for Sonar Sensor	Possible Hazards			
Power Line Down	1	2		4
Data Line Down	1	2		4
Debris Covering Sensor	1	2		4
Debris on Road	1	2		4
Rugged Road Conditions			3	
Course Limitations	1	2		4
Electrical Component Shortage	1	2		4
Attenuation from water drops	1	2		4
Snow reflects sensor signal	1	2		4
Electrical Component failures	1	2		4
Moisture affect performance	1	2		4

Table 4.6-2: Linking sonar sensor faults with hazards

4.7 Conclusions

Through Fault Tree Analysis the component faults that lead up to system failure can be qualitatively predicted. The symptoms of these failures were distinguished by previously completed research The fault characteristics provide useful insight as to what happens with the controller if a failure occurs. This information can be used to predict which hazards are imminent. The results gathered from the Fault Tree Analyses show that the hazard definitions are far too broad to provide useful information in failure analysis. Future work should include redefining the set of existing hazards to allow for use in failure analysis. Similarly, the trees could not be analyzed quantitatively using Boolean algebra because of the large number of undeveloped events and the lack of data corresponding to those events. Although the Fault Trees developed for the radar and sonar sensors are very limited and exclusive in nature, they serve as a springboard to other applications.

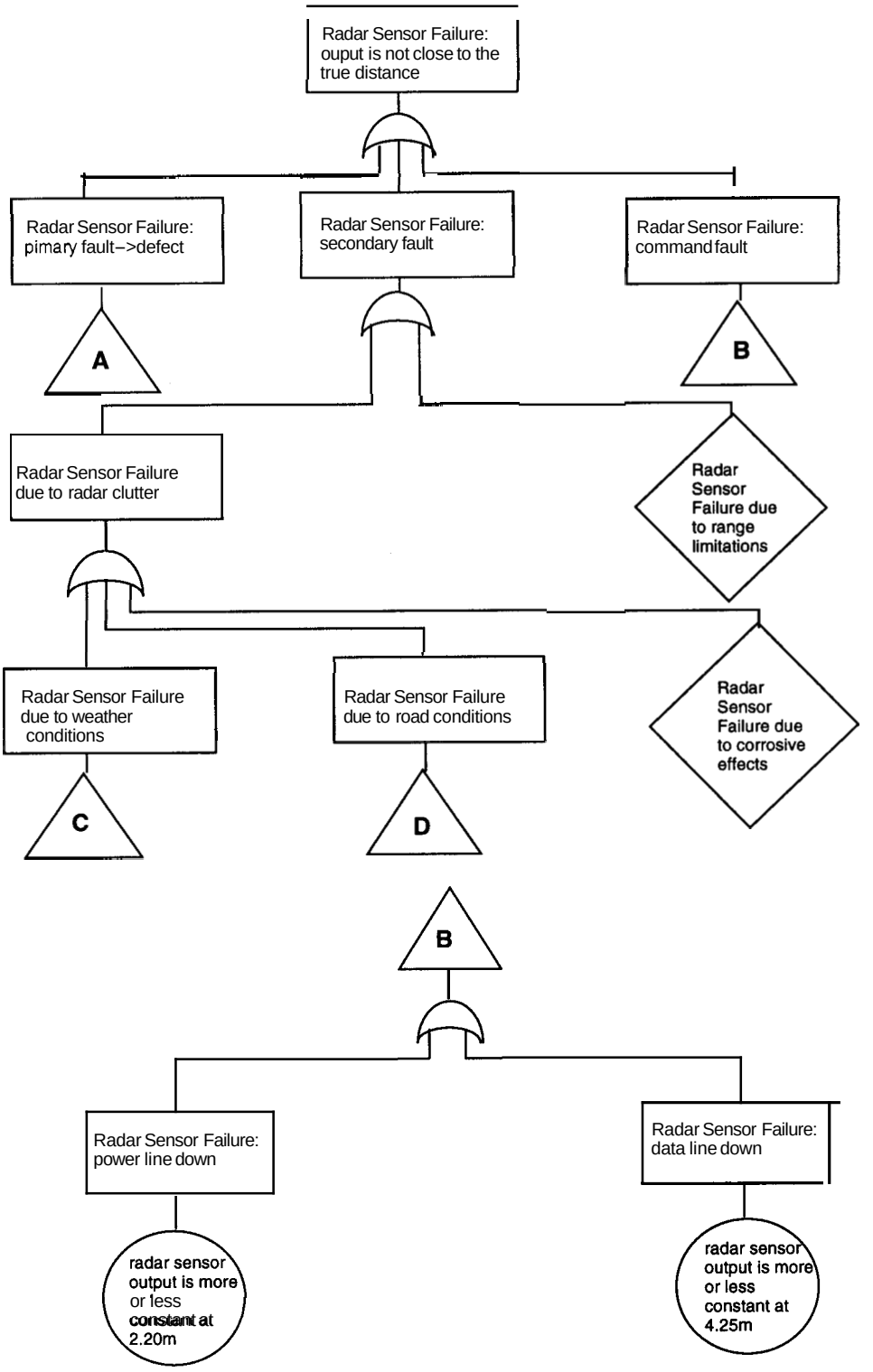


Fig. 4.6-2: Fault tree of radar sensor: Main tree

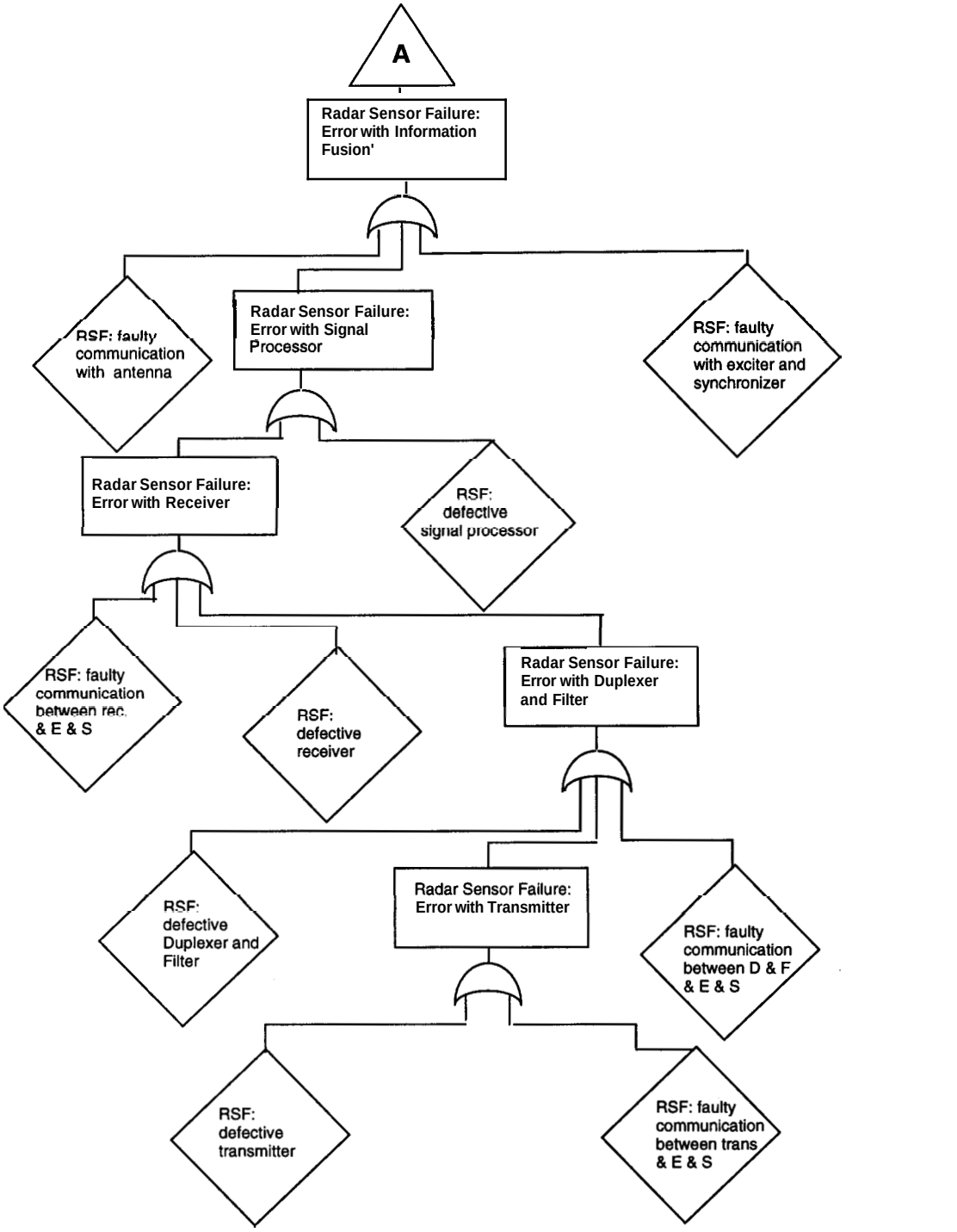


Fig. 4.6-3: Fault tree of radar sensor: Primary fault branch

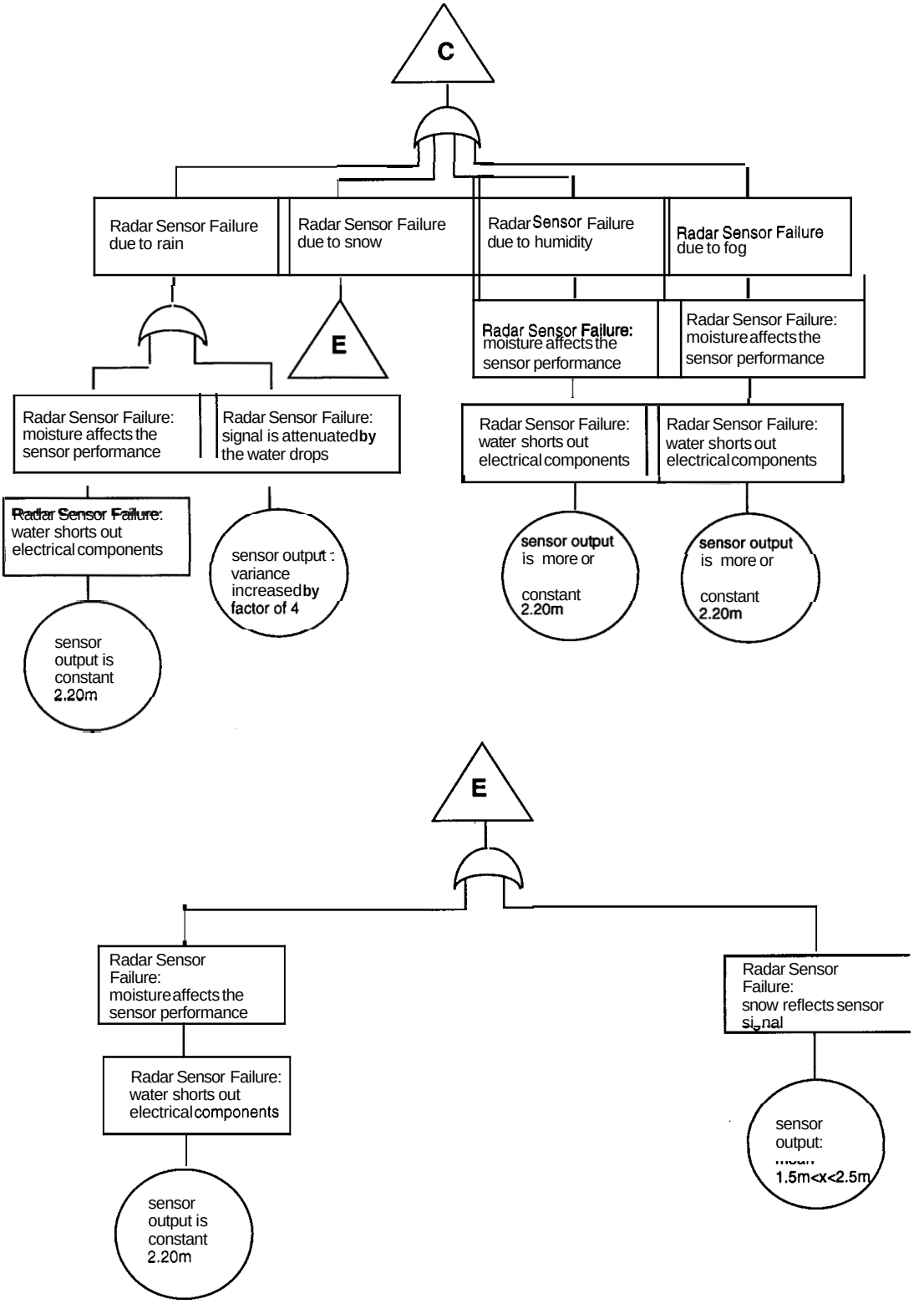


Fig. 4.6-4: Fault tree of radar sensor: Weather condition branch

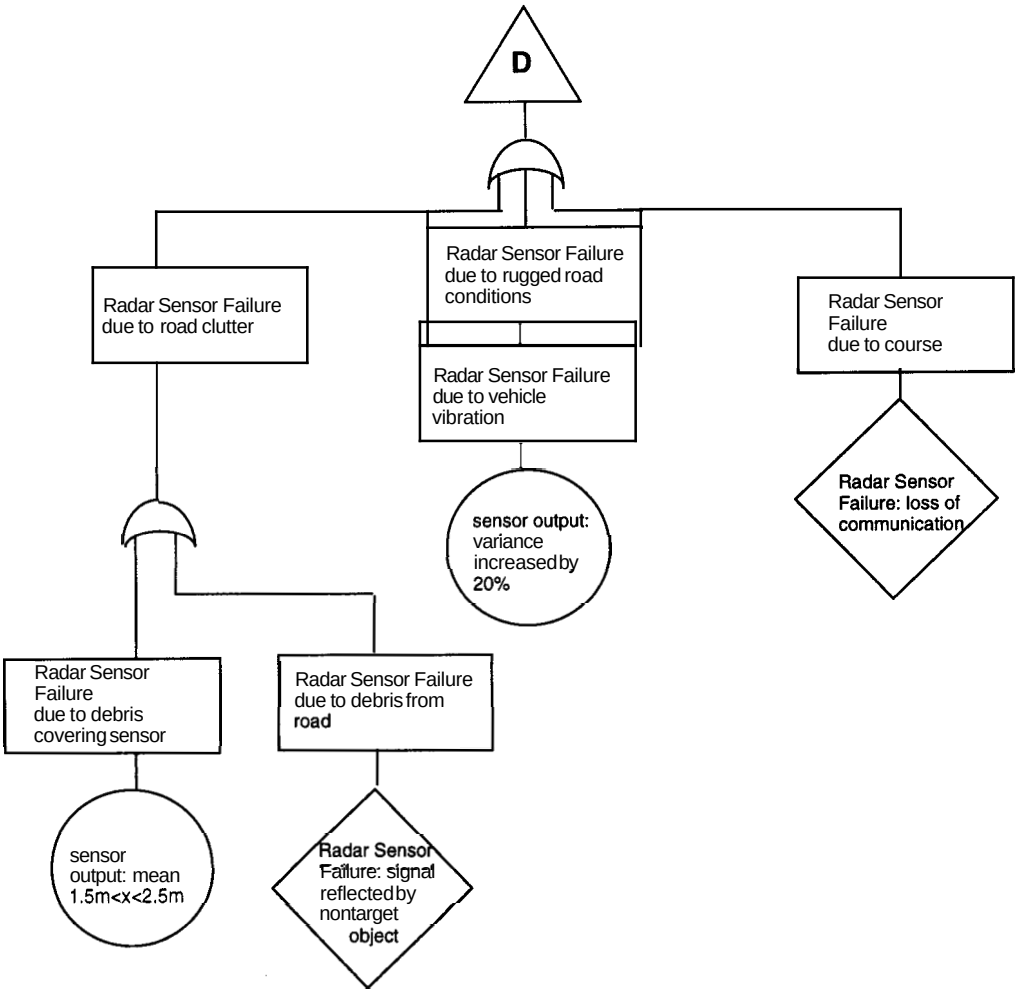


Fig. 4.6-5: Fault tree of radar sensor: Road condition branch

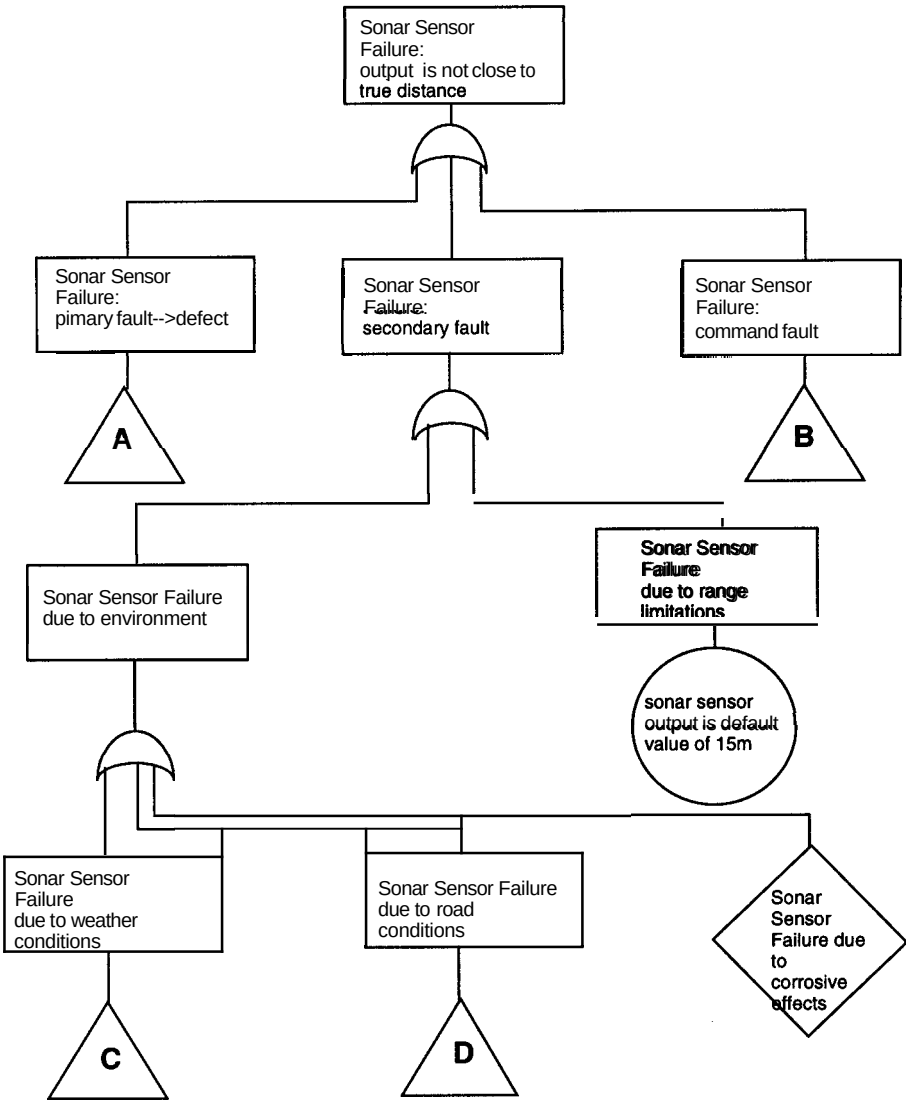


Fig. 4.6-6: Fault tree of sonar sensor: Main branch

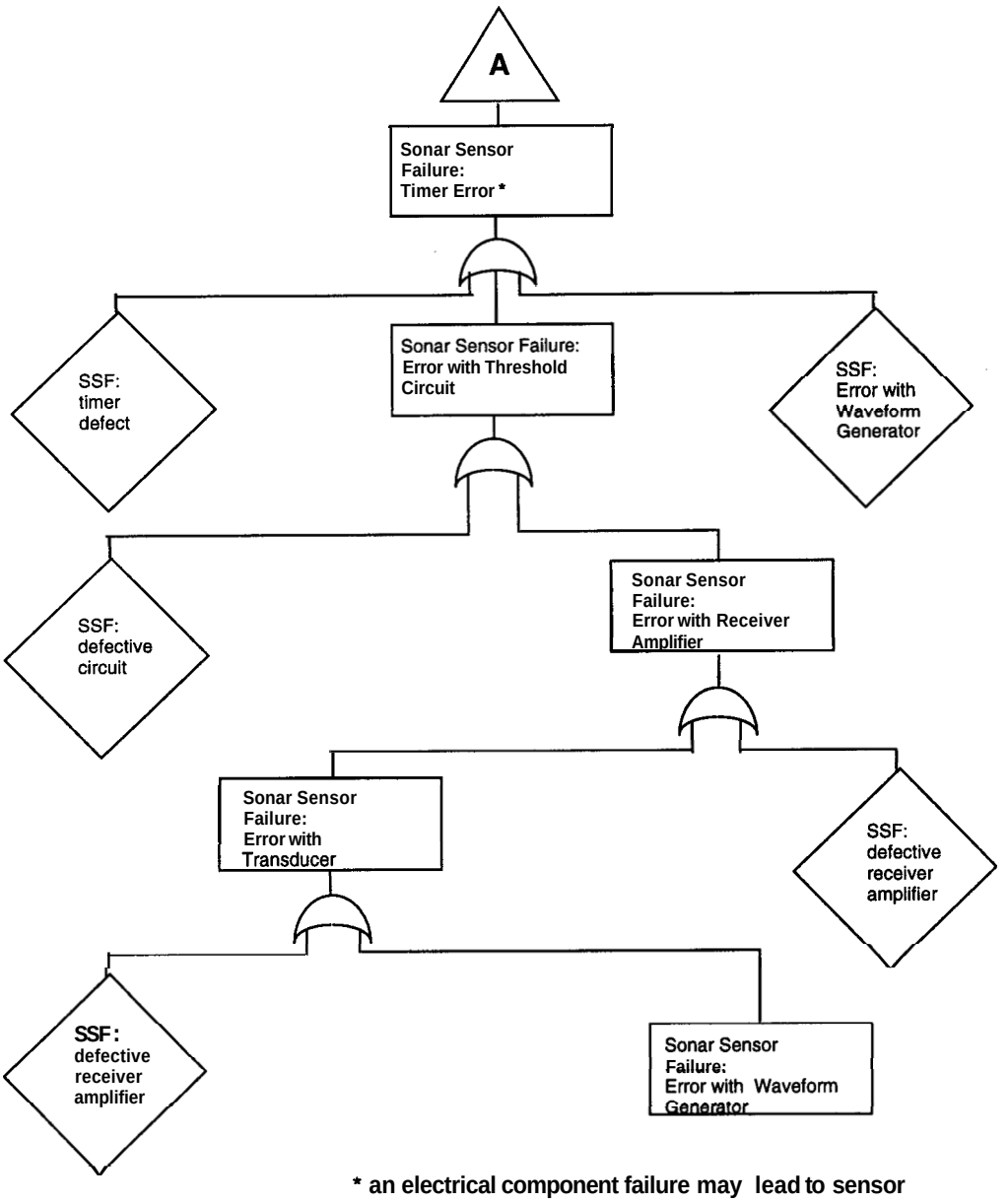


Fig. 4.6-7: Fault tree of sonar sensor: Primary fault branch

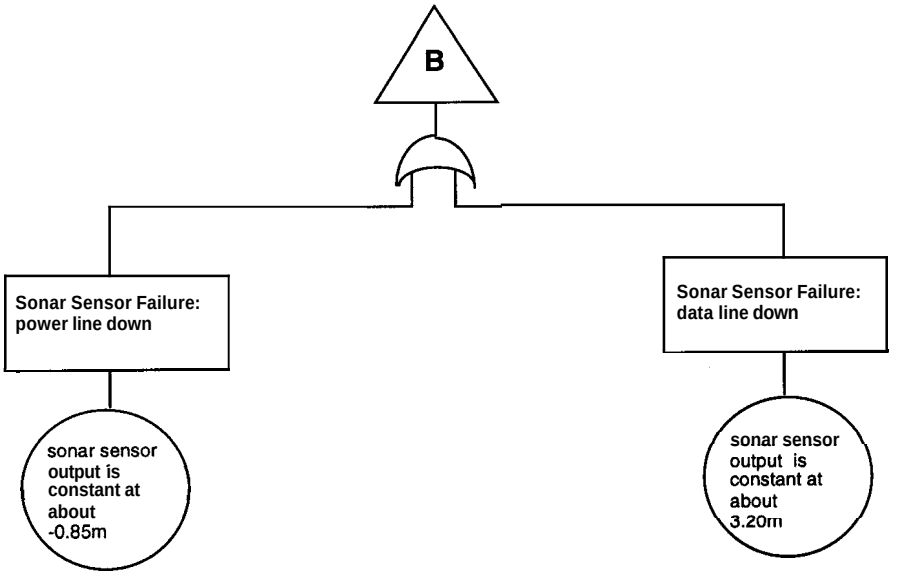


Fig. 4.6-8: Fault tree of sonar sensor: Command fault branch

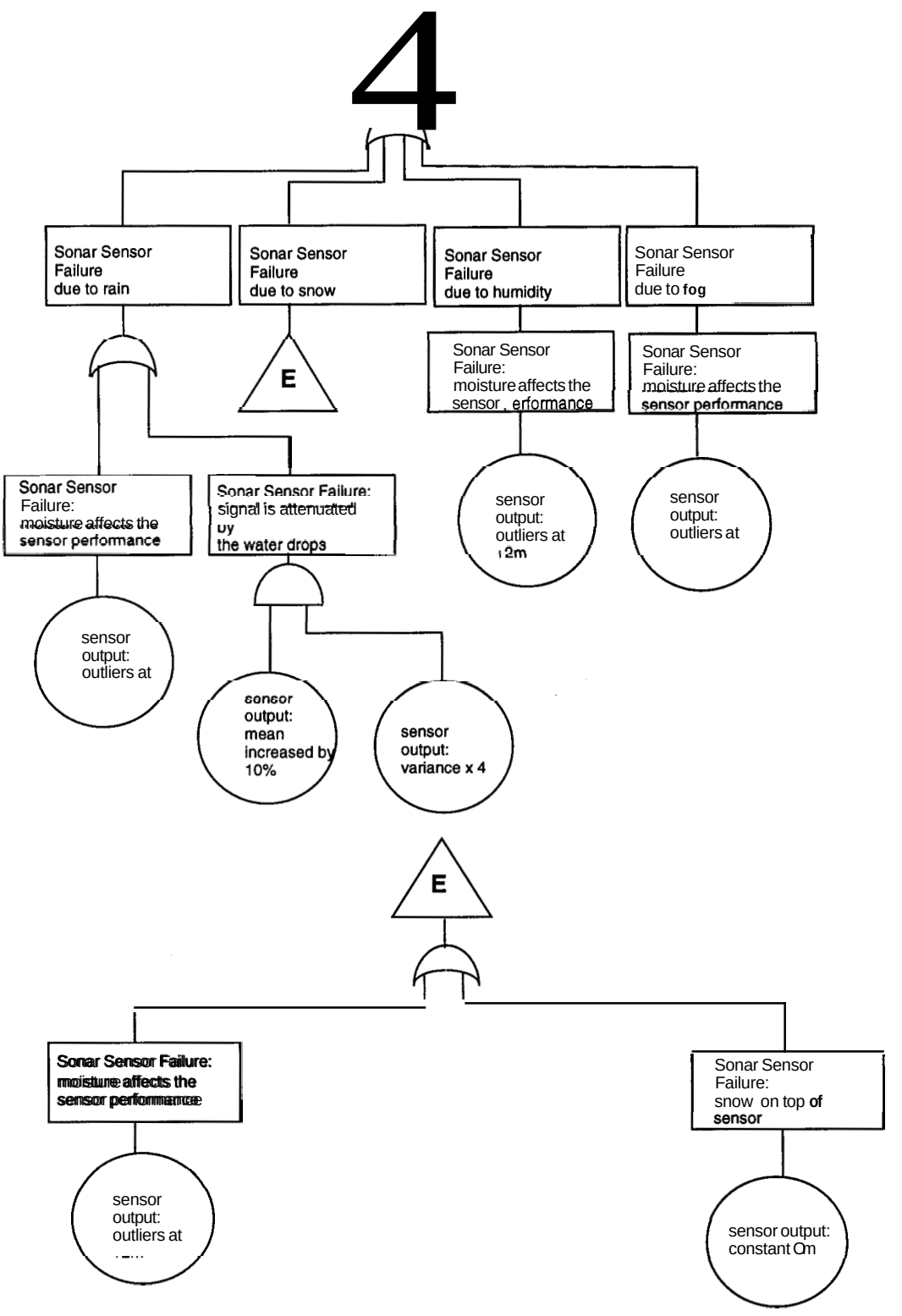


Fig. 4.6-9: Fault tree on sonar sensor: Weather condition branch

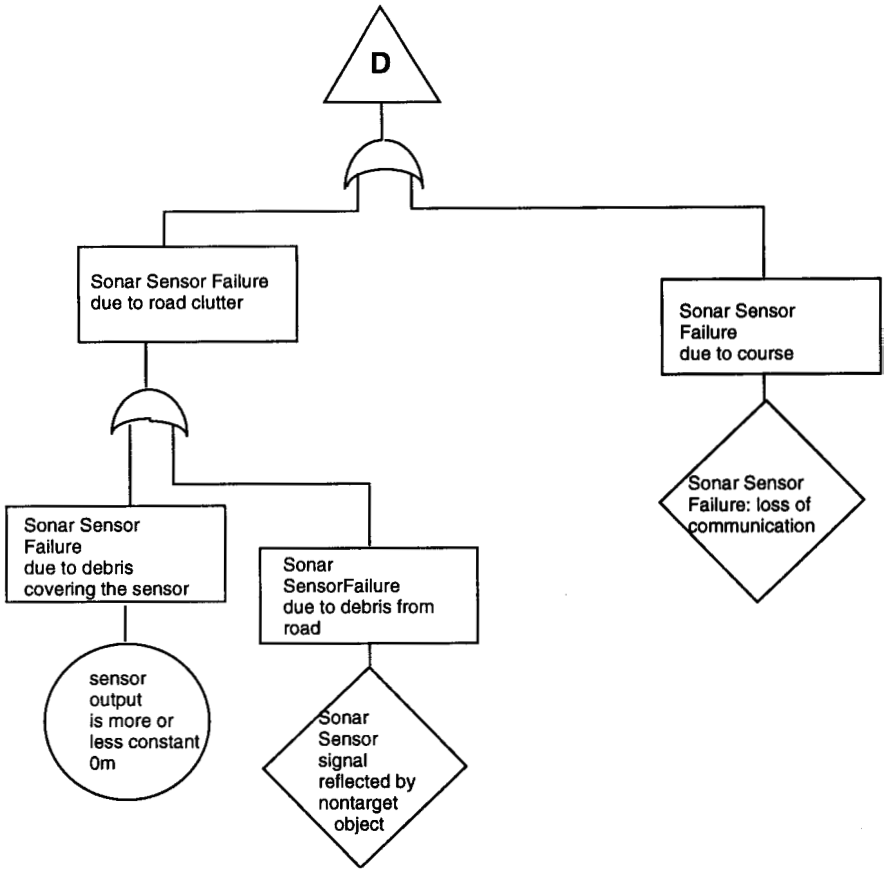


Fig. 4.6-10: Fault tree of sonar sensor: Road condition branch

5. Intelligent Decision Advisor

This chapter outlines an approach to optimal safety decision making under adverse conditions in Advanced Vehicle Control Systems (AVCS), with specific application to the California Partners for Advanced Transit and Highways (PATH) program. The outputs from safety decision making are intended as guidelines for public policy makers to ensure minimal injury and vehicle damage while substantially improving highway capacity. The Intelligent Decision Advisor (IDA) is introduced and its components are described.

5.1 Overview

Automobiles in the AVCS envisioned by the PATH program rely on sophisticated communication and multi-sensory capabilities in order to carry out the main goals of increasing highway capacity while improving passenger safety. Dangerous conditions can arise when sensors and/or environmental variables (vehicle or roadway) are less than ideal (Agogino, et al., 1995; Hitchcock, 1992). Therefore, Advanced Vehicle Control Systems must be able to determine when such conditions are likely to be present and then decide which action (e.g. full stop, lane change (Godbole, et al., 1995; Lygeros, et al., 1995) will cause the minimal expected passenger injury and, as a secondary goal, minimal expected vehicle damage. At the same time, highway capacity maximization is desirable. This research divides this optimization process into the modules shown in Figure 5.1-1: Hazard Diagnosis, Trajectory Prediction, Impact Evaluation, and Injury/Damage Minimization.

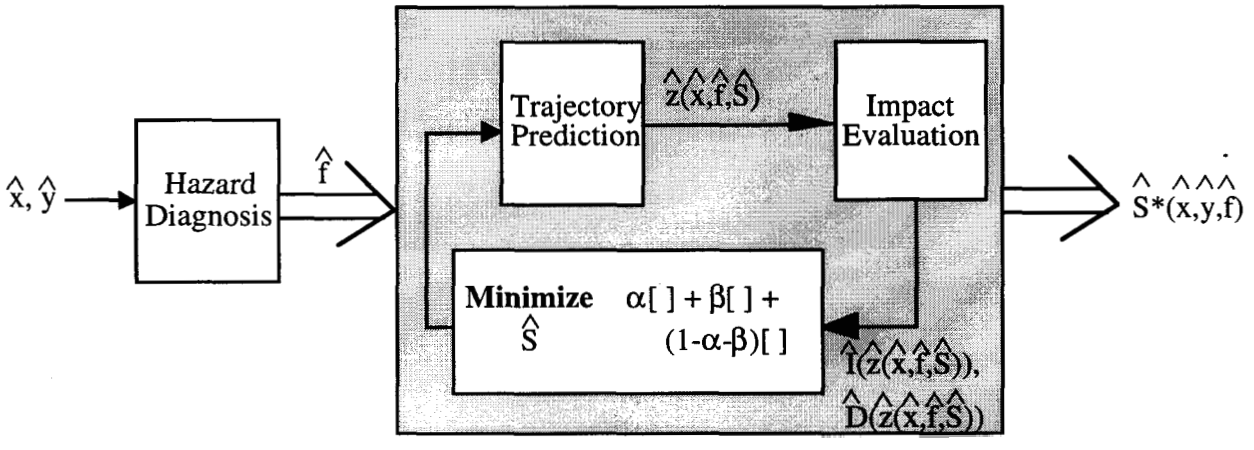


Fig. 5.1-1: IDA overview

5.2 Initial Conditions and Sensor Readings

Suppose that two vehicles, $j = 1, 2$, are traveling down an AVCS capable highway. The highway is assumed to be straight with the flow of traffic moving along the Y axis and lateral movements measured along the X axis. Vehicle heading angles (EDC, 1989) are described by ψ. See Figure 5.2-1 for a pictorial representation of these coordinates. The vehicles are exposed to a variety of situations which may or may not result in an accident. Neither, one, or both vehicles may then incur injury and damage, depending on the course of action chosen by each vehicle.

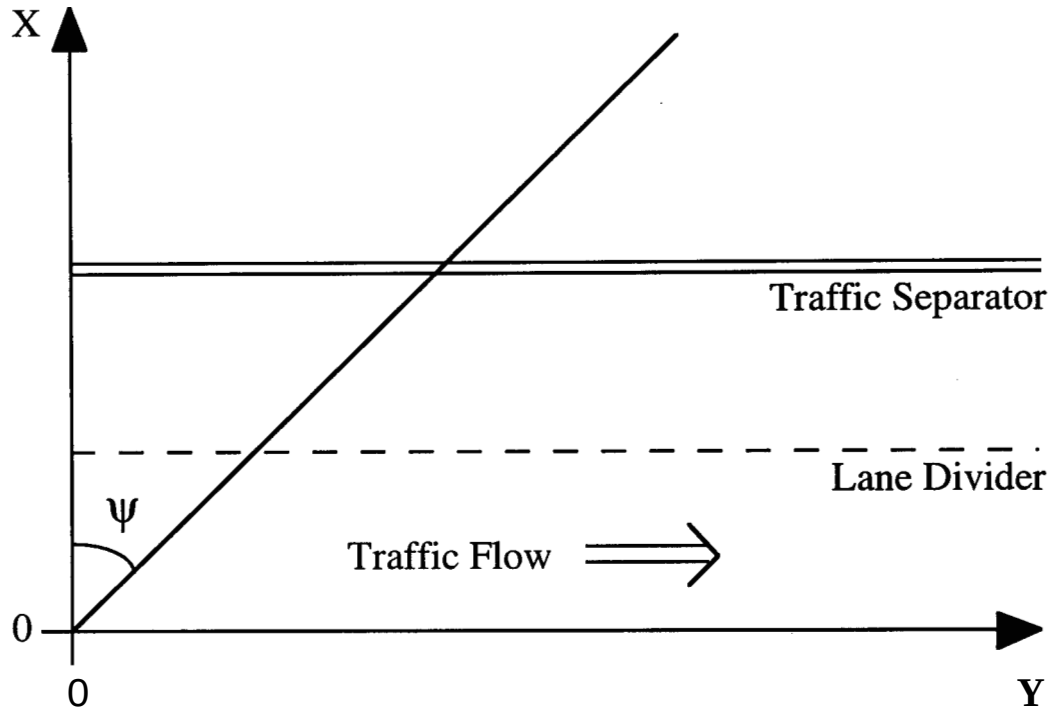


Fig. 5.2-1: Vehicle coordinate system

The m components of the initial condition vector $\hat{x}_j$ include the initial velocity and planar coordinates (X, Y, and ψ) for vehicle j = 1, ..., J (J=2). Velocity is expressed in both longitudinal and lateral terms.

$\hat{x}_j = x_{j,1}, x_{j,2}, \ldots, x_{j,m}$.

Vector $\hat{x}$ consists of all $\hat{x}_j$, j=1,...,J

Each vehicle is equipped with a variety of sensors which track vehicle operating conditions (e.g. velocity, acceleration, braking). This research will limit its consideration to radar, sonar, vision, and **GPS** sensors which estimate the longitudinal distance from the vehicle under consideration to the next obstacle or vehicle. Assume the total number of sensors is n. Sensor output vector $\hat{y}_j$ includes values for all n sensors on vehicle j.

$\hat{y}_j = y_{j,1}, y_{j,2}, \ldots, y_{j,n}$.

Vector $\hat{y}$ consists of all $\hat{y}_j$, j=1,...,J.

5.3 Hazard Diagnosis

Since we have restricted ourselves to longitudinal sensory input, the hazards that we will consider are similarly constrained. Hazards under consideration include: stationary obstacle or nonautomated vehicle blocking roadway (H1), rain (H2), and roadway debris (gravel, dirt, trash,

etc.)(H3). Each hazard can occur in varying amounts; e.g., an obstacle can vary in its separating distance from the vehicle, and rain can range from drizzle to torrents. The presence of these hazards affects the vehicle's safety in two ways: obstruction of a sensor can greatly lessen the sensor's accuracy and a hazard can diminish the controllability of the vehicle through reduced braking and steering capabilities.

Assume that each hazard can be represented by a random variable H_i. The probability density function f_j describes the likelihood that vehicle j encounters the three hazards, given that $\hat{x}$ and $\hat{y}$ describe the current initial conditions and sensory information. Let $\hat{f}$ be the vector of density functions for all J vehicles.

$$\hat{f}(h1, h2, h3 | \hat{x}, \hat{y})$$

At least initially, we will assume that hazards are independent. In other words,

$$f_j(h1, h2, h3 | \hat{x}, \hat{y}) = f_j(h1 | \hat{x}, \hat{y})\, f_j(h2 | \hat{x}, \hat{y})\, f_j(h3 | \hat{x}, \hat{y})$$

A methodology for diagnosing these hazard likelihoods within the **PATH** project is described in a later section and also by Chao and Agogino (1997). Results from Bellm (1995) about the effects of various hazard sources (e.g., snow, rain, fog, dirt) are incorporated into vehicle sensor readings. This enables the determination of each hazard's signature. Comparison of the sensor readings with the established hazard signatures allows for the calculation of the likelihood of any defined hazardous condition.

Bellm (1995) achieved experimental results for the effects of hazard sources while testing with radar and sonar devices. In these experiments, he observed the changes in radar and sonar longitudinal measurements between two cars when the following environmental hazards were independently present: rain, plastic, dirt, fog, and mechanical vibrations. This research refers mainly to the results from the first three hazards and leaves the remaining two hazards for further investigation.

Furthermore, because the experimental periods used by Bellm exceeded a desirable real-time measurement time period for hazard diagnosis, we have broken the experimental data into **6** data samples per experiment. Very similar statistical results were achieved for all **6** samples.

Figure 5.3-1 shows the data for one sample period using a radar to measure longitudinal distance between the two test vehicles. Measurements for each of the following **4** cases are illustrated: static (no hazard), rain, plastic, and dirt.

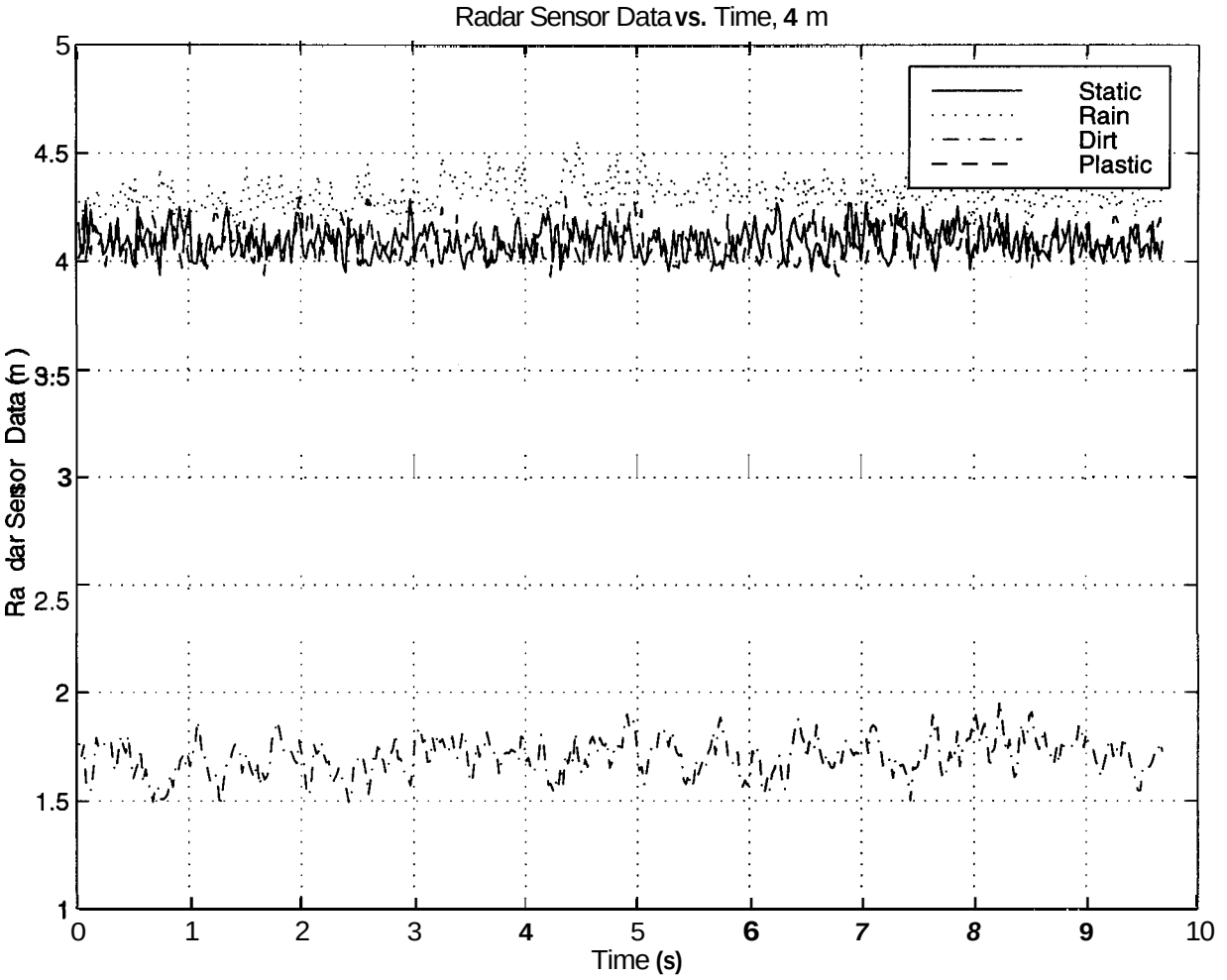


Fig. 5.3-1: Radar output over time

Figure **5.3-2** shows the data for one sample period using a sonar to measure longitudinal distance between the two test vehicles. Measurements for each of the following **4** cases are illustrated: static (no hazard), rain, plastic, and dirt.

Problems in mean values evident in Tables **5.3-1** (radar statistics) and **5.3-2** (sonar statistics) show significant differences from the "actual" value of **4** meters, especially for the dirt hazard experiments. These discrepancies are due to experimental calibration error. We will therefore derive statistical conclusions from the variance, which is unaffected by miscalibrations.

Observation of Figures **5.3-1** and **5.3-2** lead us to a couple of immediate conclusions. First, the data appears to be a stochastic process, since data points undergo irregular motion cycles which never repeat exactly. Second, the mean and variance for each hazard type seem to not vary in significant amounts from sample to sample (therefore over different periods of time). ***So*** the statistical properties do not change with time.

Concluding that the data are stochastic and stationary, we can now employ autocorrelation and Fourier (frequency domain) analysis to further investigate patterns due to different hazards.

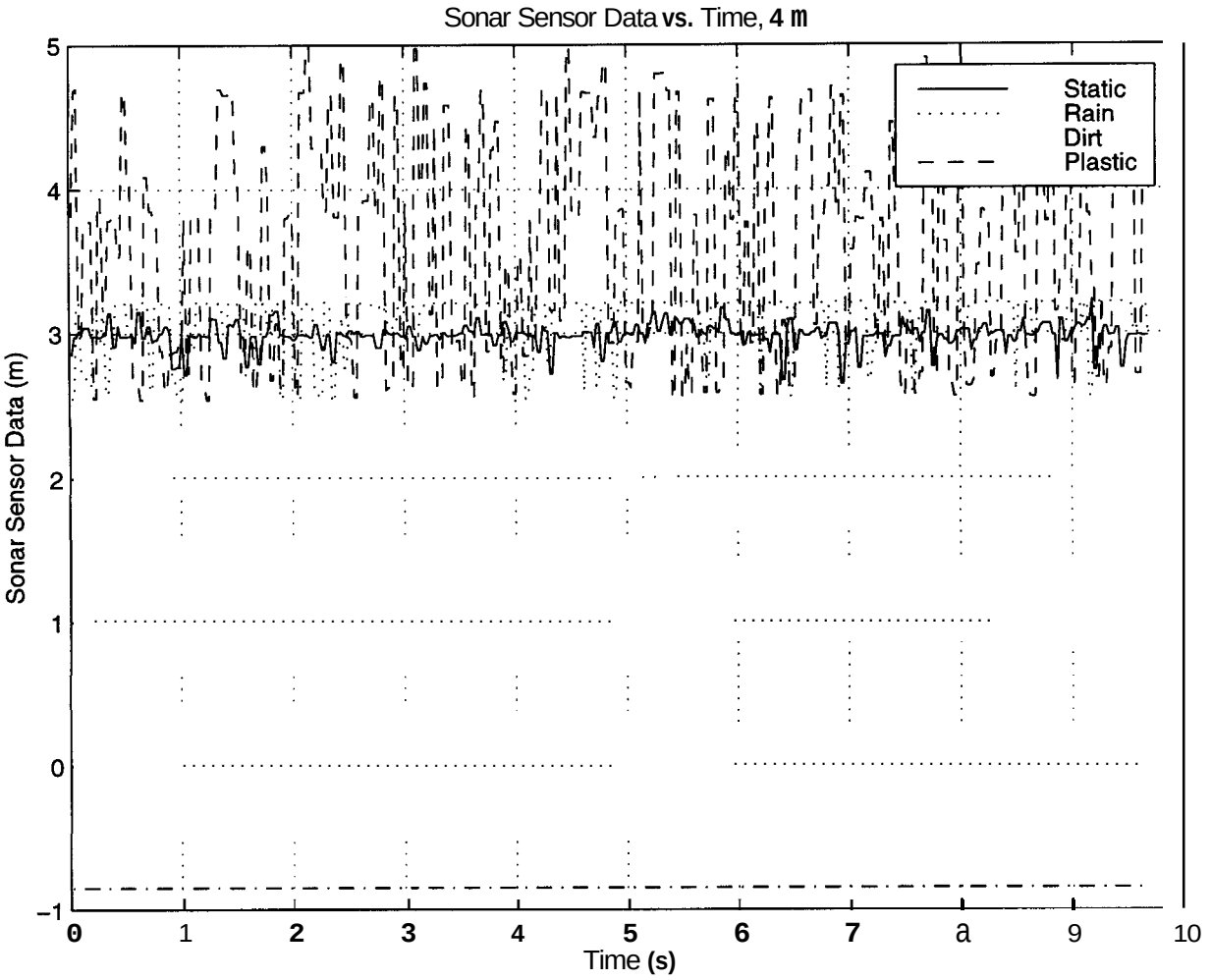


Fig. 5.3-2: Sonar output over time

Environmental Conditions	Static	Rain	Dirt	Plastic
Sample **1**				
Mean	3.1004e+00	4.0907e+00	1.7049e+00	4.0862e+00
Variance	4.2437e-02	5.0102e-03	7.3444e-03	5.5197e-03
Sample **2**				
Mean	3.1260e+00	4.0903e+00	1.6616e+00	4.0903e+00
Variance	2.8194e-02	4.4437e-03	8.2618e-03	6.1500e-03
Sample **3**				
Mean	3.1446e+00	4.0894e+00	1.6647e+00	4.0811e+00
Variance	2.6799e-02	5.4452e-03	1.1020e-02	5.4475e-03
Sample **4**				
Mean	3.2062e+00	4.0969e+00	1.6766e+00	4.0683e+00
Variance	2.3547e-03	4.3644e-03	9.5148e-03	5.5293e-03
Sample ***5***				
Mean	3.2004e+00	4.0872e+00	1.6572e+00	4.0814e+00
Variance	4.4513e-03	5.1070e-03	8.3543e-03	5.7412e-03
Sample **6**				
Mean	3.0609e+00	4.0939e+00	1.6889e+00	4.0763e+00
Variance	5.7691e-02	5.3539e-03	8.7959e-03	6.3276e-03

Table **5.3-1:** Radar statistics — Six sample sets

Environmental Conditions	Static	Rain	Dirt	Plastic
Sample **1**				
Mean	2.9856e+00	3.1004e+00	-8.4500e-01	3.6287e+00
Variance	7.2663e-03	4.2437e-02	6.0535e-29	6.4951e-01
Sample **2**				
Mean	2.9916e+00	3.1260e+00	-8.4500e-01	3.7442e+00
Variance	1.0528e-02	2.8194e-02	6.0535e-29	6.0815e-01
Sample **3**				
Mean	2.9805e+00	3.1446e+00	-8.4500e-01	3.8421e+00
Variance	1.0936e-02	2.6799e-02	6.0535e-29	5.3970e-01
Sample **4**				
Mean	2.9853e+00	4.4161e+00	-8.4500e-01	3.6534e+00
Variance	1.2324e-02	8.7689e-03	6.0535e-29	6.1468e-01
Sample ***5***				
Mean	2.9883e+00	4.5772e+00	-8.4500e-01	3.6832e+00
Variance	1.1143e-02	2.9185e-02	6.0535e-29	6.6390e-01
Sample **6**				
Mean	2.9733e+00	4.2935e+00	-8.4500e-01	3.7954e+00
Variance	1.1078e-02	8.4917e-03	6.0535e-29	7.5545e-01

Table **5.3-2:** Sonar statistics — Six sample sets

Figure 5.3-3 shows the Fourier transforms of one sample period for each of the **4** hazards measured by sonar. Note the comparatively very small magnitude range for the static and dirt cases, medium magnitude range when the rain hazard is present, and large magnitude range for the plastic case. These observations are consistent through similar analysis of the other 5 available data sets.

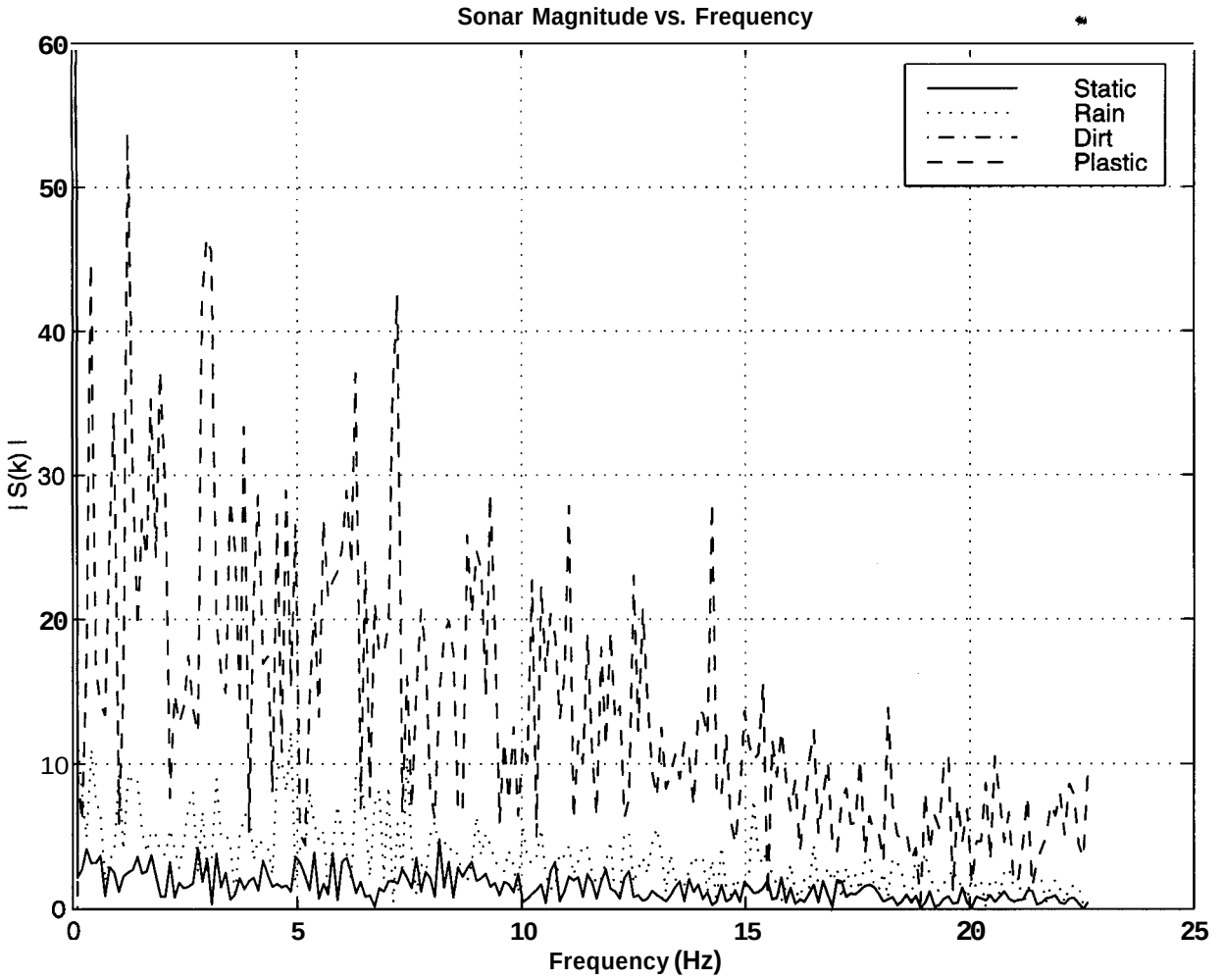


Fig. 5.3-3: Sonar magnitude

Next we consider power spectral density, the Fourier transform of autocorrelation, which describes the data set's power distribution over its frequency range. Figure **5.3-4** shows the power spectral densities of one sample period for each of the **4** hazards measured by sonar. Note the large peak (>1.25) for just the plastic case. Contrast this to comparatively smaller peaks for the other three cases (<=0.2). These observations are consistent through similar analysis of the other 5 available data sets.

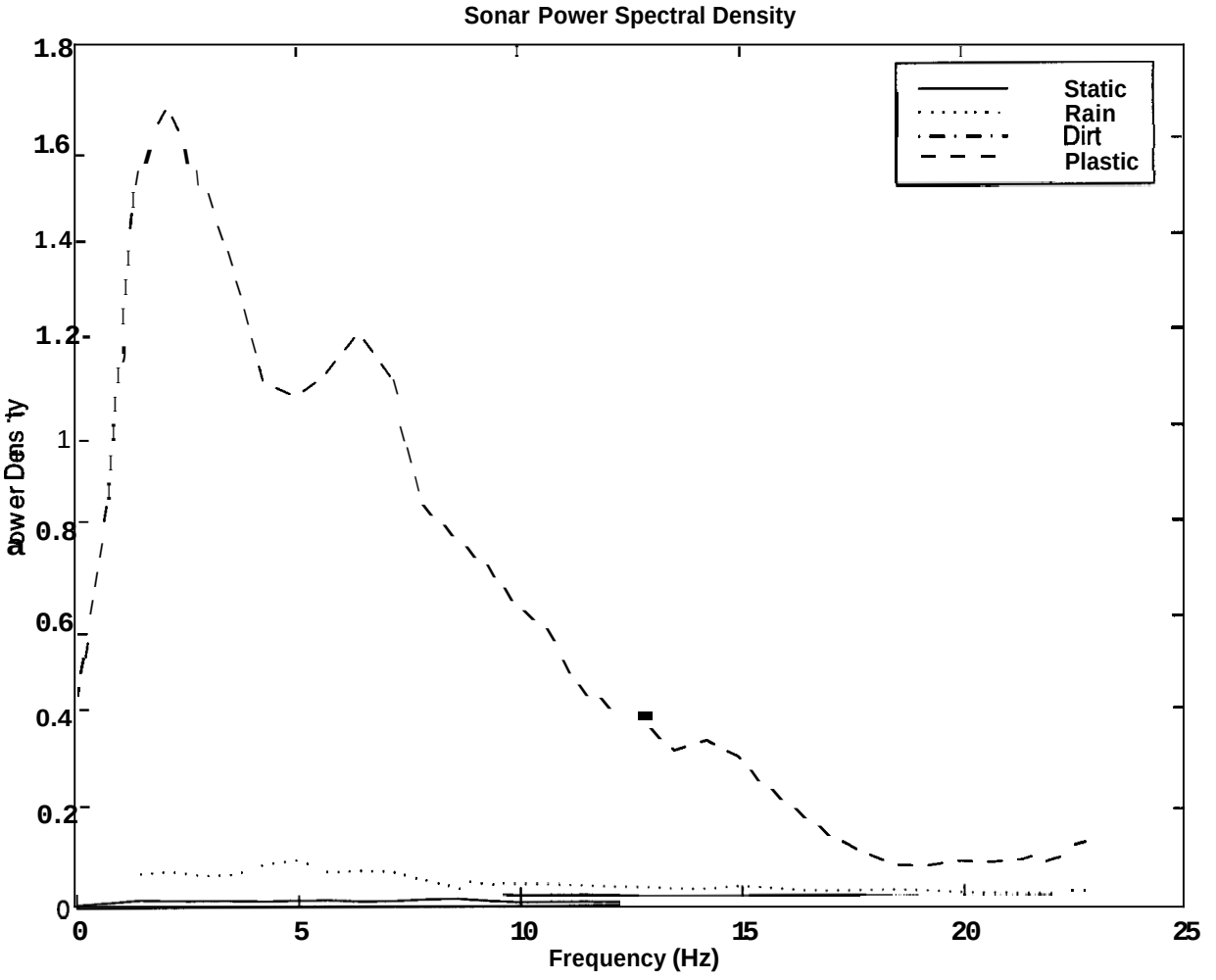


Fig. 5.3-4: Sonar power spectral density

Figure 5.3-5 illustrates the power spectral densities of one sample period for each of the **4** hazards measured by radar. Note the small peak (<=.01) for just the static case. Contrast this to medium-sized peaks (>.01, <.02) for the dirt and plastic cases. Rain and dirt hazards on the other hand exhibit large peaks (>.02). These observations are consistent through similar analysis of the other 5 available data sets.

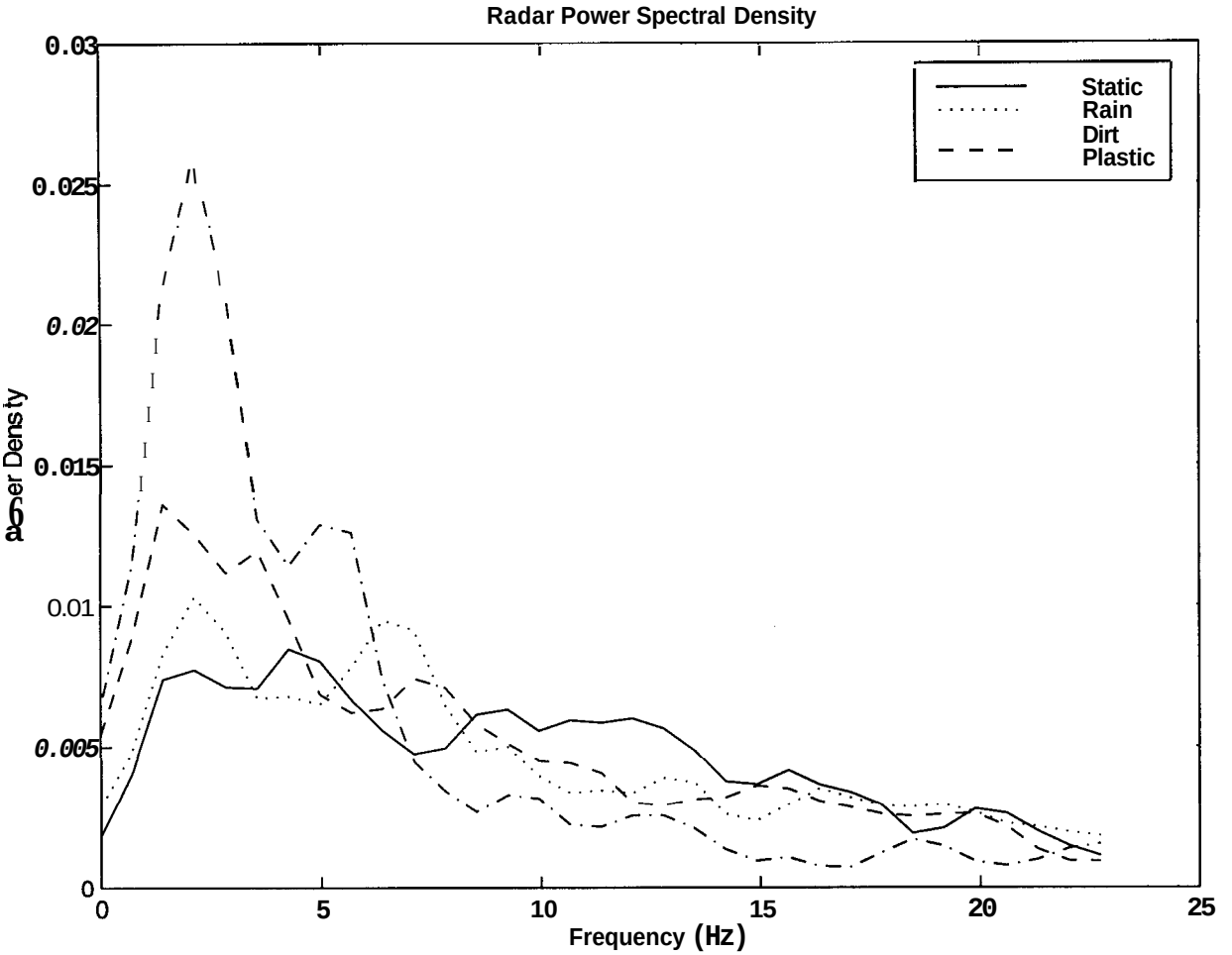


Fig. 5.3-5: Radar power spectral density

The next method of analysis will be cross spectral density, or the Fourier transform of the cross-correlation. Strongly frequency dependent but nearly time independent, cross spectral density analysis shows an interesting identifying characteristics for the rain hazard. **A** significant peak in the **1.2-3.5 Hz** region is noticeably absent in the static, dirt, and plastic cases. This observation was also observed with the other five data samples.

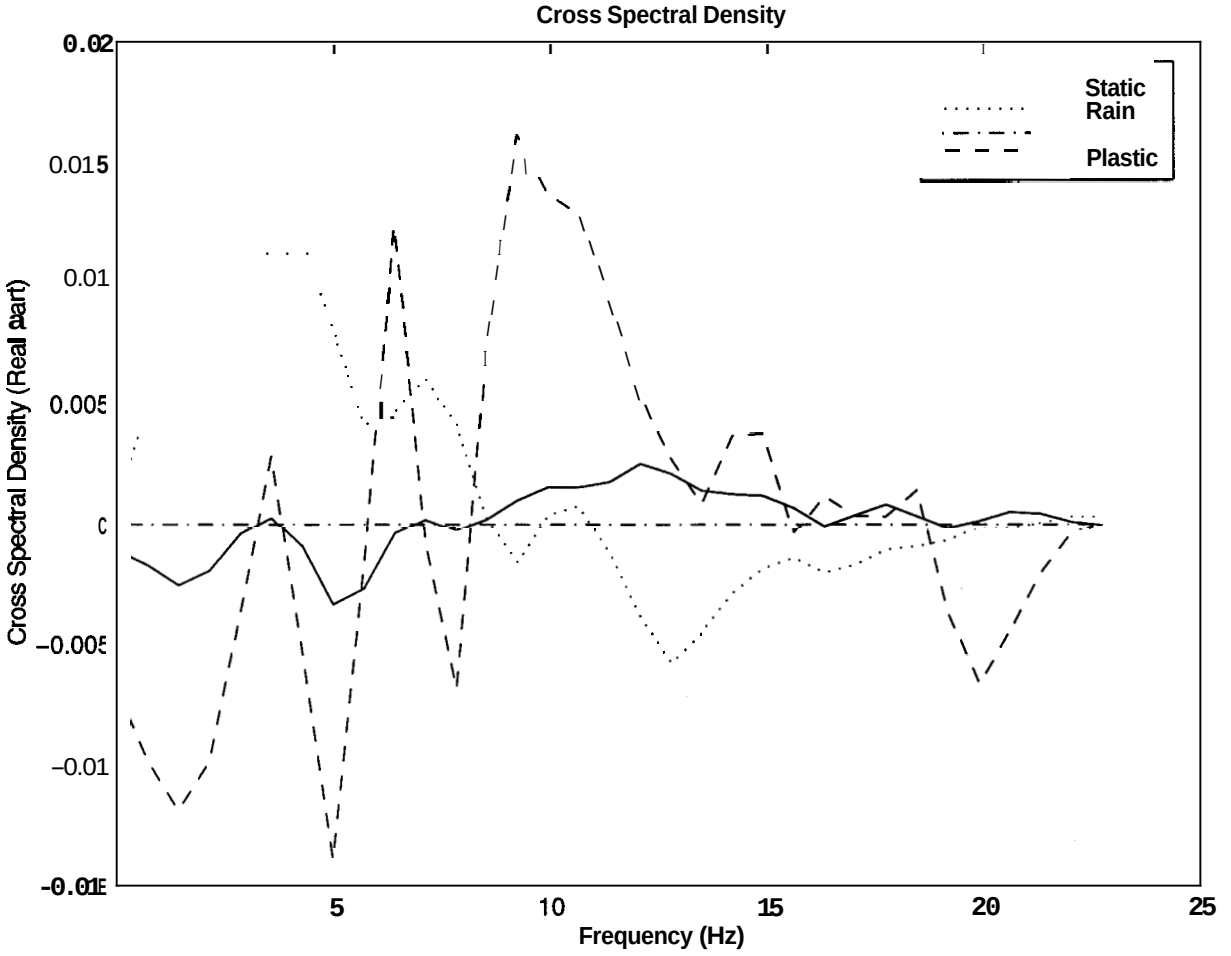


Fig. 5.3-6: Sonar/radar cross spectral density

Table 5.3-3 summarizes the above observations about statistical and spectral characteristics when analyzing data sets exhibiting different independent hazards. With this information, we are now able to assess hazard signatures by using the nearest neighbor rule. To do so, we would let each characteristic be represented along a separate dimension in an n-dimensional space. For example, we would require six dimensions to adequately describe the six defining characteristics in Table 5.3-3.

Next, we would map out a region in the n-dimensional space which corresponds to a particular hazard. Because hazards are identified by ranges of characteristics rather than particular values, the corresponding area linked to a specific hazard in the n-dimensional space will be a region rather than a point. From any point representing an experimental measurement, then, we can calculate the Euclidean distance to all six possible hazard regions. Each distance quantifies the respective likelihood of any independent hazard. The minimum of the six distances can then be concluded to correspond to the most likely case. Similarly, the region furthest from the measured point would indicate the least likely scenario.

Since these observations were drawn from a limited data set (i.e., small number of independent hazards), we will postpone the nearest neighbor calculations for further research, when more complete and better executed experiments can be undertaken in order to establish the hazard

regions. At that time, additional conclusions can be drawn about a more exhaustive collection of potential dependent hazards.

Environ-mental Condi-tions	Radar Variance	Sonar FFT Magn.	Radar PSD Peak	Sonar PSD Peak	Radar Sonar CSD Max in 1.2-3.5 Hz	Sonar Failure
Static	Small	Small	Small	Small	No	No
Rain	Large	Medium	Medium, Large	Small	Yes	No
Dirt	Small	Small	Large	Small	No	Yes
Plastic	Small	Large	Medium	Large	No	No

Table 5.3-3: Differentiating environmental effects on sensors

5.4 Trajectory Prediction

Depending on input variables, a strategy will be chosen to prevent vehicle collision with other vehicles or with roadway obstacles. This strategy will be selected based on hazard likelihood, initial conditions, and sensor readings. Hence, represent the strategy function with $\hat{S}(\hat{x},\hat{y},\hat{f})$. Strategy $\hat{S}$ is a vector of control variables including steering and acceleration/deceleration for each vehicle over a series of time steps.

Therefore, for inputs of initial conditions $\hat{x}$, hazard likelihoods $\hat{f}$, and vehicle strategy $\hat{S}$, we predict the vehicles' trajectories due to reduced vehicle handling capabilities under adverse conditions with the Trajectory Prediction module. The module uses the Engineering Dynamics Corporation Vehicle Analysis Package (EDC, 1989) to yield Collision Deformation Classifications (CDC) and relative velocities at impact (Delta-V) for each vehicle over a series of T time steps.

These p values are combined into a crash vector $\hat{z}_j(\hat{x},\hat{f},\hat{S})$ for each vehicle. Vector $\hat{z}_j(\hat{x},\hat{f},\hat{S}) = z_{j,1}, z_{j,2}, \ldots, z_{j,p}(\hat{x},\hat{f},\hat{S})$, where the crash vector describes impact variables: output Delta-V and Collision Deformation Classifications, area of deformation, specific longitudinal or lateral area, specific vertical area, type of damage distribution, and maximum extent of penetration (EDC, 1989). Vector $\hat{z}$ consists of all $\hat{z}_j$, $j=1,\ldots,J$.

Our research used the Reconstruction of Accident Speeds on the Highway portion (EDCRASH) of the EDC Vehicle Analysis Package. Inputs into the package include vehicle size, type, tires, position, initial velocity, etc. In addition, we can also specify road conditions like rain, fog, snow, and ice through steering, acceleration, and deceleration capabilities. This software has two major

limitations: only 2 vehicles can be simulated and no non-vehicles (like barriers) can be modeled. The second limitation can be circumvented by simulating a barrier by giving appropriate dimensions and weight to vehicle #2 and running collisions between vehicle #1 and the 'barrier'.

5.5 Impact Evaluation

Given the predicted trajectories in the last section and their respective angles and velocities of impact, we now determine the severity of impact in the Impact Evaluation module as rated by previous crash data for different impact configurations. Data from the National Transportation Highway Traffic Administration (BTS, 1996) are coupled with $\hat{z}$ to determine values for the resulting injury and damage. Variable $I_j(\hat{x},\hat{f},\hat{S})$ represents the bodily injury amassed by the driver (assume no other passengers) in vehicle j. Variable $D_j(\hat{x},\hat{f},\hat{S})$ represents the property damage sustained by vehicle j. Vector $\hat{I}$ consists of all $\hat{I}_j$, $j=1,...,J$, and vector D consists of all $\hat{D}_j$, $j=1,...,J$.

5.6 Expected Injury/Damage Minimization

Finally, let parameters α and β describe the relative proportions of total bodily injury, total vehicle damage, and average vehicle speed, respectively, in the weighted sum

$$\sum_j [\alpha * I_j(\hat{x},\hat{f},\hat{S}) + \beta * D_j(\hat{x},\hat{f},\hat{S}) - (1-\alpha-\beta)\frac{1}{J} S_{j,4}]$$

where $\alpha \in [0,1]$ and $\beta \in [0,1]$. Initial condition component $S_{j,4}$ refers to the final longitudinal velocity of vehicle j.

The determination of actual values for α and β will be left for transportation policy makers. The goal of this research will be to choose the strategy $\hat{S}^*$ with the minimum expected total damage/casualty but maximum average speed. In other words, we will minimize the weighted sum of injury, damage and negative average speed subject to the system's physical constraints and initial conditions. Note that average speed is subtracted in the objective function, indicating its maximization. Strategy vector $\hat{S}^*$ details optimal steering and acceleration values over T time steps. ***So*** the objective for the Intelligent Decision Advisor is to

Minimize

$$\sum [\alpha * I_j(\hat{x},\hat{f},\hat{S}) + \beta * D_j(\hat{x},\hat{f},\hat{S}) - (1-\alpha-\beta)\frac{1}{J} S_{j,4}]$$

Subject To

Constraints defining $\hat{x}$, $\hat{y}$, $\hat{f}$, $\hat{z}$, $\hat{S}$, $\hat{D}$, and $\hat{I}$

and we solve exhaustively using stochastic optimization.

5.7 Conclusions

This chapter has described the framework which we developed to enable the automated vehicle to choose an optimal maneuver or strategy in hazardous conditions. First, we discussed hazard diagnosis and results from extended analysis of earlier tests performed under hazardous

conditions. This analysis yielded a table of characteristics differentiating the four hazardous conditions (rain, fog, debris, no hazard) under consideration. Next we discussed our implementation of EDCRASH in determining vehicle trajectories after impact. Environmental inputs to EDCRASH were briefly described and outputs detailing impact on the vehicles were listed. The effects of impact were then assessed according to statistics from the National Transportation Highway Traffic Administration, resulting in objective figures for bodily and property damages. Finally, we described the overall optimization problem as a minimization of damages subject to a variety of constraints.

This research assumed a limited case in which we restricted our longitudinal distance assessment to two sensors and in which we considered only three independent hazards. Because of the restrictions, we were able to solve exhaustively, using stochastic optimization. More difficult cases will also yield interesting results in future research when we extend our techniques to scenarios with multiple sensors and dependent hazards.

6. Preliminary Investigation of Three New Sensors

6.1 Overview

Reliability is one of the most important aspects in IVHS. High fidelity of sensor data is required. However, a single sensor cannot always work reliably under different situations. It is therefore desirable to have various kinds of sensors to obtain redundant readings for each quantity we measure. In some cases when some of the sensors fail, we can still obtain correct information from other sensors. In order to know to what degree the readings of each sensor could be trusted under different motions and environmental conditions, it becomes very important to study the behaviors of each sensor. This includes noise characteristics, noise models, work range, factors that might affect the performance of the sensor under particular conditions, etc. Here we introduce several new sensors which are potentially very powerful positioning systems: the Global Positioning System (GPS) sensor, the vision sensor, and the laser radar sensor. To date we have collected only static data because of the limited experimental opportunities and data resources and therefore have not yet developed exact noise models of the sensors. From the limited data, however, we can still obtain some interesting characteristics which provide directions for further research.

6.2 GPS sensor

6.2.1 Introduction to the GPS Sensor

GPS was developed by the U.S. Department of Defense and is based on a constellation of 24 satellites orbiting the earth at a very high altitude. The basic principle behind GPS is the use of satellites as reference points for triangulating cars' position on earth. Position is calculated from distance measurements to satellites. While three measurements are required to determine exact position, another measurement is required to eliminate clock offset. The distance to a satellite is determined by measuring how long a radio signal takes to reach us from that satellite. We need a receiver on our car to receive the radio signal. If a GPS satellite were directly overhead it would only take about 6/100ths of a second for the radio message to get to us. ***So*** the GPS sensor should work well even when the car measured is driven at very high speeds.

The major error sources in GPS measurements are: satellite clock error, ephemeris error, receiver errors, and atmospheric/ionospheric delay. In addition, the accuracy of GPS can purposefully be degraded by the Department of Defense using an operational mode called "Selective Availability" or "S/A". S/A is designed to deny hostile forces the tactical advantage of GPS positioning. When, and if, it is implemented it will be the main component of GPS error.

Differential GPS (DGPS) measurements can be much more accurate than standard GPS measurements. The main idea of DGPS is as follows. If we put a GPS receiver on the ground in a known location, we can use it to determine exactly which errors the satellite data contains. Acting as a static reference point, the receiver transmits an error correction message to any other GPS receivers in the local area, and they can then use that error message to correct their position measurements. The correction can therefore eliminate virtually all error in their measurements. One additional advantage of using the GPS receiver is that its performance would not be affected by weather.

While no measuring device is perfect, DGPS is no exception. There is still noise in its measurements though in much smaller quantity than in the standard GPS system. Furthermore, the update of GPS readings is quite slow (usually on the order of several Hertz), which is a major shortcoming of the GPS sensor. The reader is directed to (Jeff Hurn, 1989) for more information on the working principle of GPS.

6.2.2 Analysis of GPS Data

Here we will focus on the analysis of two sets of static data. The first set of data was collected at the Palo Alto, CA airport using two cars (SRI, Palo Alto, 1997). A **GPS** receiver was mounted on each of the two cars. The two cars were initially stationary for about 5 minutes (sample numbers 1-1050) during which time the static measurements were recorded. The cars were then driven around for a period of time (sample numbers 1050-1750) and stopped to retake measurements (sample numbers 1750-2250).

Figure 6.2.2.1 shows the original data.

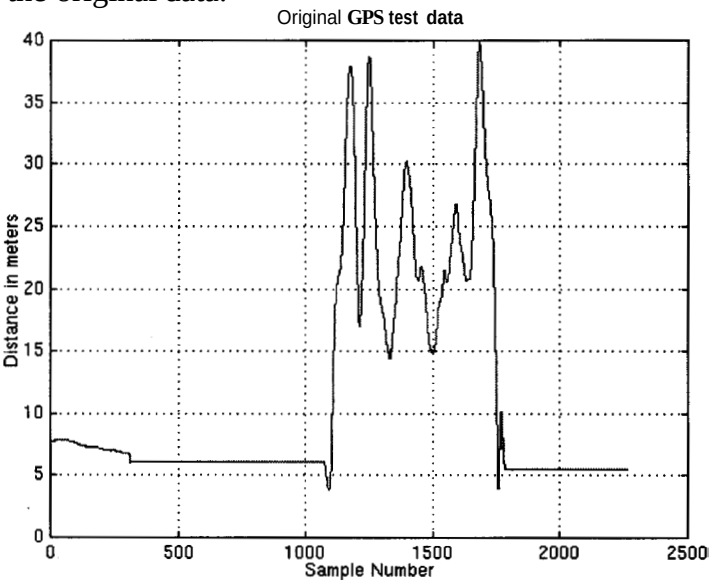


Fig. 6.2.2-1 The first set of original GPS data

We can see that there is a slope at the very beginning of the data (sample numbers 0-300). From the principles of GPS sensors we know this occured because of initial carrier ambiguity resolution, so we attribute a less accurate position during this period based on pseudoranges from the receivers. More data are needed to determine the expected length of duration and also the accuracy of the data during this period. From this set of data, it can be seen that the error could total several meters; we therefore recommend regarding the initial readings as sensor malfunction and ignoring the GPS readings at this stage.

Figure 6.2.2.2 shows the first static part of the data (sample numbers 400-1050) and its validation using the Kalman Filter (Alag, 1996). Figure 6.2.2.3 illustrates the second static part of the data (sample numbers 1750-2250) and its validation using the same Kalman Filter. Note their difference in scale compared to Figure 6.2.2.1. From them we can observe some noise characteristics. First, the magnitude of the noise is quite small (less than 4cm), which means that the GPS readings are already quite accurate. Both data parts exhibit increases in noise magnitude, possibly due to the change in the numbers of available satellites. It can be seen from next set of data that the noise is affected by the numbers of the satellites available. But that is not the only possible reason here; it could also be caused by the GPS receiver itself, either from the receiver's electronics or antenna.

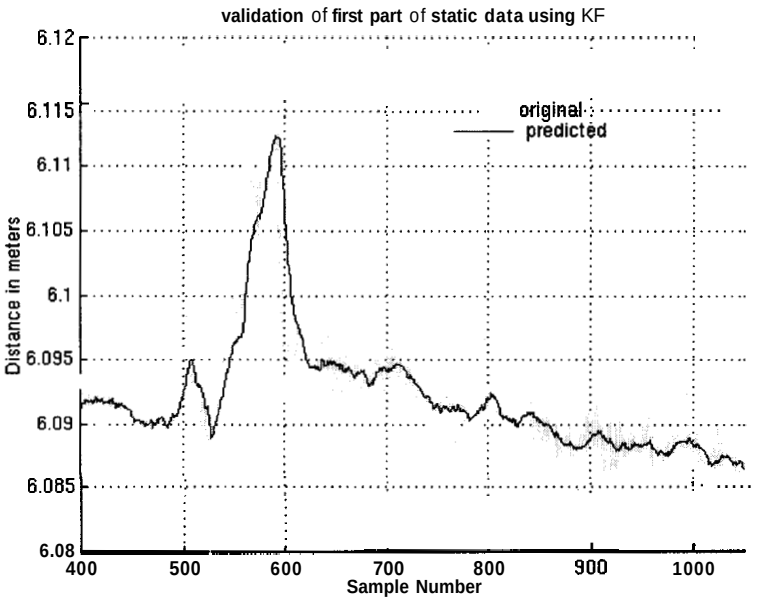


Fig. 6.2.2-2 First static part of first set of data and its validation

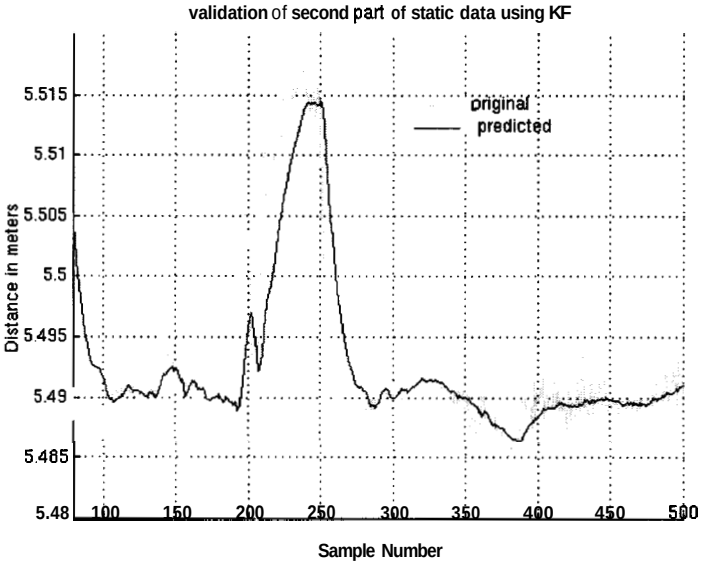


Fig. 6.2.2-3 Second static part of first set of data and its validation

In order to learn more about noise characteristics, we can now compare it with white Gaussian noise. By doing this, we can derive some statistical knowledge of the noise and then determine which type of validation and fusion algorithm we should employ to reduce the noise and obtain the most reliable evaluation of the sensor reading. Though we have not determined the sources of **GPS** data noise, this does not affect the practice of our analysis method. Here we use the first static part (sample numbers 450-1050) of the data to compare with the same length of white Gaussian noise, generated in Matlab. First we compare their histograms. The more random numbers we take, the closer the shape of the histogram to the probabilistic density function (pdf) of a Gaussian random variable. That is the nature of Gaussian random numbers.

Figure 6.2.2.4 shows the histogram of 10^5 samples of a Gaussian white process with variance 1 and Figure 6.2.2.5 shows the histogram of 600 samples of Gaussian white process with variance 1. Compare the histogram of the first static part (450-1050) of GPS data (Figure 6.2.2.6) with that of the same number (600) of samples of Gaussian white process with variance 1 (Figure 6.2.2.5). We cannot determine, from the noise of the plots, whether the GPS noise is Gaussian, because one histogram could correspond to a different process but one process could have only one histogram. Further conclusions on the **GPS** noise need further experimentation and investigation.

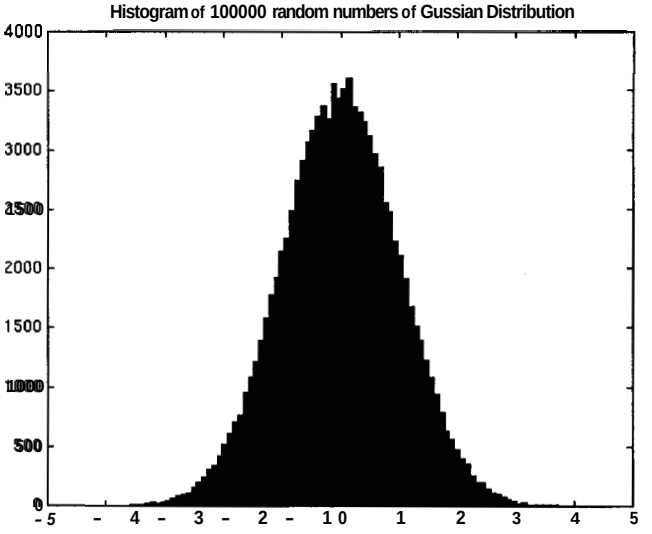


Fig. 6.2.2-4 Histogram of 10^5 samples: Gaussian white process, variance=1

At the same time we are interested in whether the noise is white or to what extent it can be approximated by white noise. Here we use an autocorrelation estimate to qualitatively evaluate the whiteness of the noise. Calculation of the autocorrelation estimate is described in (Oppenheim and Schafer, 1989). White-noise processes should have their autocorrelation very small everywhere except at m=O (the first sample point from which the samples used in autocorrelation estimate begins). From this we can only get a qualitative answer to the question whether the noise is white or not. (For quantitative answers to this question, one could refer to Jenkins and Watts, 1968.) We would not calculate it qualitatively at this stage since we do not have enough data sets and it is yet uncertain whether this set of data is typical for the GPS sensor. Figure 6.2.2.7 shows the autocorrelation estimate of 400 samples of Gaussian white process with variance 1 and Figure 6.2.2.8 illustrates the autocorrelation estimate of the first static part of GPS data (samples 450-850). It can be seen from these two plots that the noise of this set of GPS data is quite different from white noise.

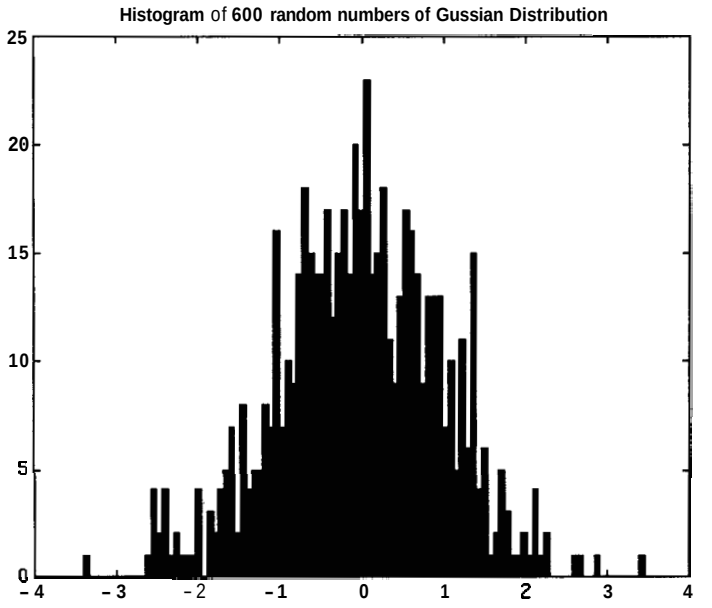


Fig. 6.2.2-5 Histogram of 600 samples: Gaussian white process with variance= 1

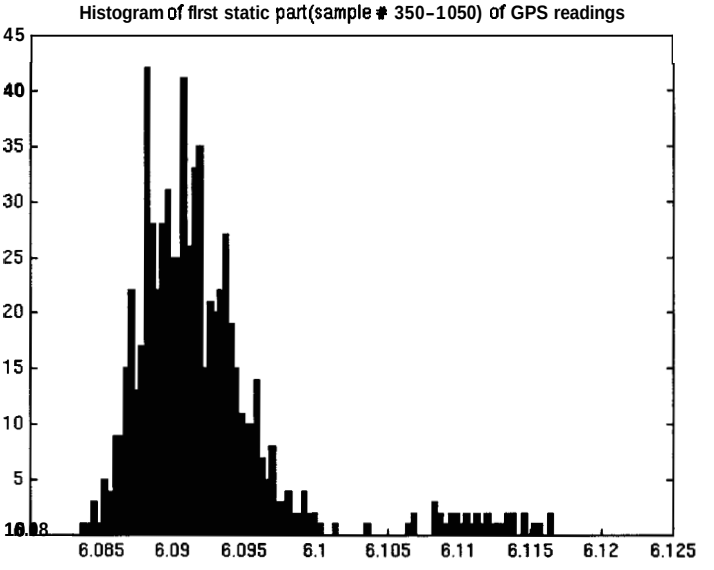


Fig. 6.2.2-6 Histogram of first static part of GPS data (sample450-1050)

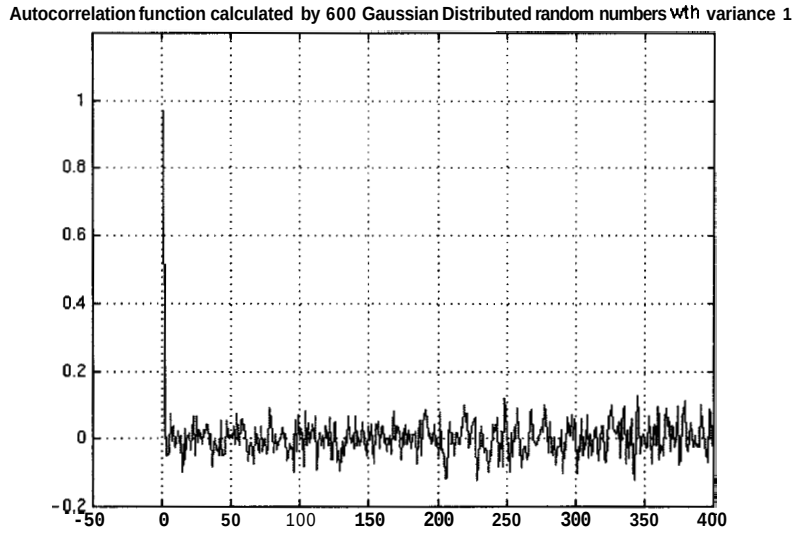


Fig. 6.2.2-7 Autocorrelation estimate of 400 samples of Gaussian white process

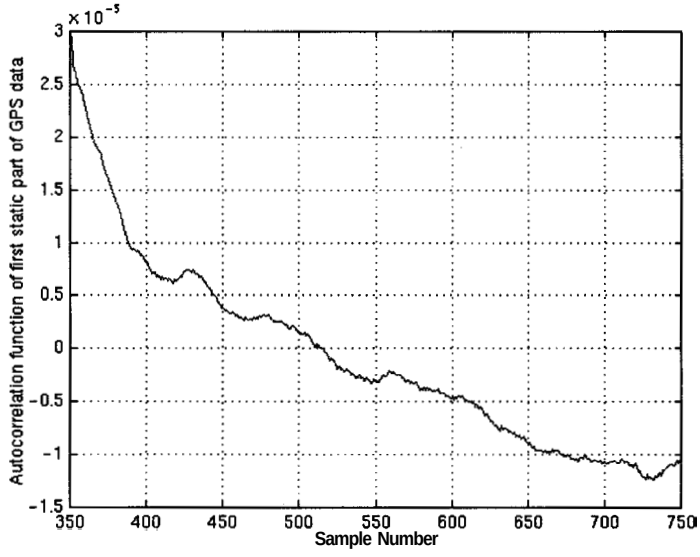


Fig. 6.2.2-8 Autocorrelation estimate of the first static part of GPS data

The second set of data is recorded in (SRI, Northbound, 1997). It was collected by SRI International when the car was driven northbound on California Highway 85 around Sheller Road. The initial part of this freeway segment is clear of overpasses. The two antennae of the **GPS** receivers were on one car, with one on the front and the other on the rear. The first part of the data is quite good, however the second part is noisy. What we are concerned with in this set of data is that the noise increases when the number of available satellites is low. This might occur when the vehicle going travels under an overpass or passes large trucks that cover a portion of the sky. More experiments need to be performed in order to determine the cause. **A** video camera will be used in future experiments to get a record of the surroundings during the tests to see if overpasses and large trucks really affect the available number of satellites.

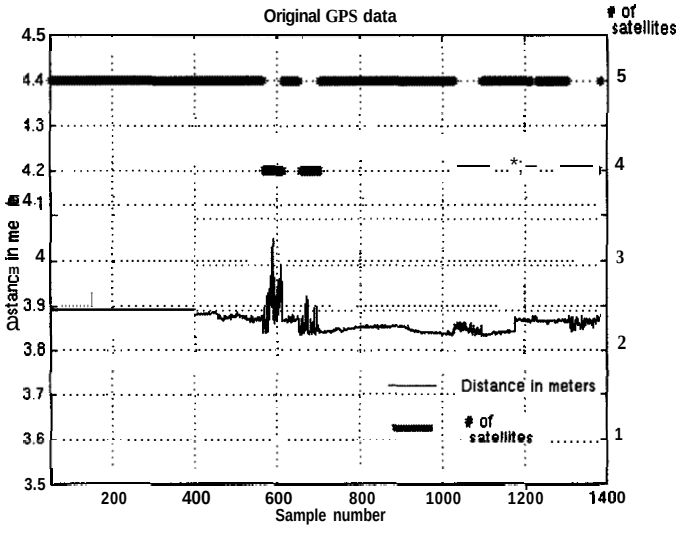


Fig. 6.2.2-9 Second set of original GPS data

Figure 6.2.2.9 shows that the first part of the data is quite good. **As** we can clearly see the data is much noisier when only four satellites are available than when five are available. Also we can see that at some parts of the data, even if the number of satellites is the same, the magnitude of the noise varies by a significant amount. Other factors need to be considered in reviewing this phenomenon (*e.g.* the receiver itself). In addition, we are yet unsure of the the cause of the step which occurs at the last part of the data. We hope that upcoming experiments will help to determine those factors that affect the performance of GPS sensors.

Preliminary Conclusions on the GPS Sensor

From the above two sets of data, we can derive some initial conclusions about the characteristics of the GPS sensor. 1) When enough satellites are available, the readings of the sensor can be very accurate, i.e., the noise magnitude could be less than **0.4**meters. 2) The accuracy of the GPS reading is affected by the number of satellites used in resolution. **3)** Since there is an initial period in which the GPS receiver software resolves carrier ambiguity, output from the GPS sensor is less accurate and we recommend discarding the respective GPS readings. **4)** In addition to the number of satellites available, there are still some other reasons which would result in the increase of noise in GPS readings on which more investigation is needed. 5. Based on our current GPS data, the noise is not white.

Above are some conclusions we can draw from the limited data. Although we were unable to derive a specific noise model from them, we were alerted to some features of the GPS sensor which will be very useful for in deciding areas of focus for future experiments.

6.3 Vision Sensor

6.3.1 Introduction to the Vision Sensor

The vision system is composed of two cameras installed on the front of the car. It tracks the car ahead by matching feature points of the vehicle in the two image windows of the cameras. Our data (McLauchlan and Wang, San Diego, 1997) were collected by Philip McLauchlan from the vision group of the UC Berkeley Department of Electrical Engineering and Computer Science and Jiangxin Wang, from our group, in San Diego using cars from Honda in July 1997. In this experiment, we used a stereo vision system to provide real-time distance information between two

cars in conjunction with a laser radar sensor. In the next two sections we will focus on the vision data separately and then analyze the laser radar data and the fused data of these two sensors.

6.3.2 Analysis of Vision Data

Figure 6.3.2.1 shows the original vision and laser radar data recorded during the experiment in San Diego. The data is static since we stopped two cars at a distance for a few minutes and measured the distance and then drove one car forward or backward and stopped again to take the measurement. Repeating this process several times resulted in the data shown in Figure 6.3.2.1. Comparing vision data and laser radar data in this figure, it can be seen that there is a bias between the two set of data. Figure 6.3.2.2 illustrates the original and estimated vision data.

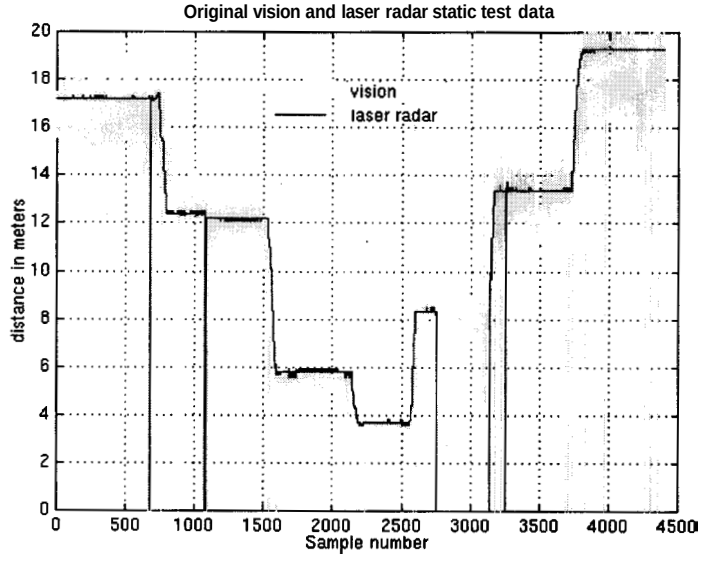


Fig. 6.3.2-1 Original vision and laser radar data

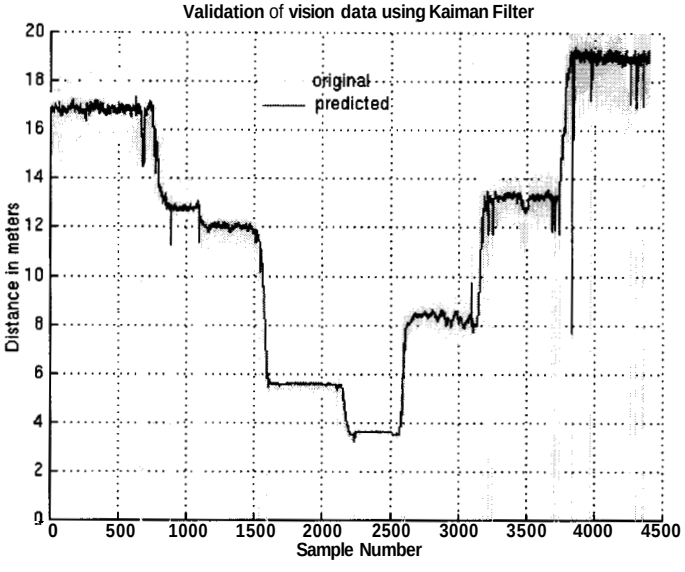


Fig. 6.3.2-2 Vision data and its validation using KF

The estimated data is obtained by using a Kalman Filter (Alag, 1996) together with laser radar data which we will discuss in more detail in Section 6.4. We can see that the vision data is quite noisy. Furthermore, the longer the distance, the noisier is the data. From the data, we can roughly calculate the variances of the vision data at each distance we measured. According to the results of calculation, we can derive the relationship between distance and variance of the vision data shown in Figure 6.3.2.3. The true relationship might not be exactly the same as is shown, but we can roughly say that it is almost in direct proportion.

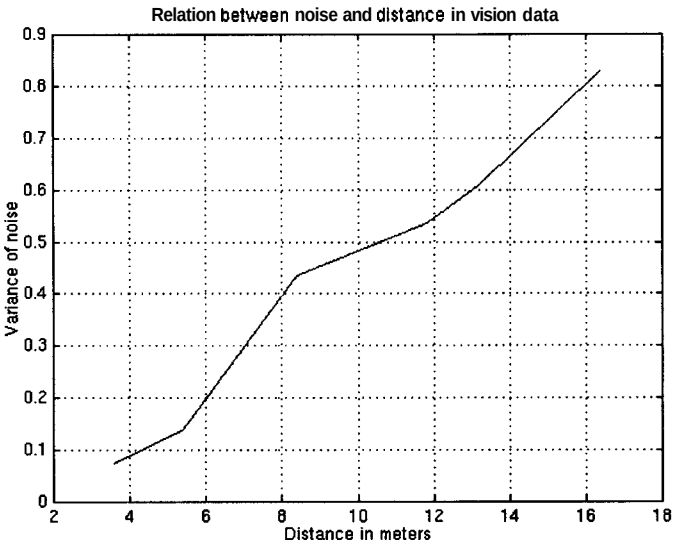


Fig. 6.3.2-3 Relationship between distance and variance of vision data

Just as we did with the GPS data, here we will analyze the vision data by comparing the first part of the data (samples 1-600) with a Gaussian white process. Figure 6.3.2.4 is the histogram of the vision data. Comparing Figures 6.3.2.4 and 6.2.2.5, the histogram **of** 600 samples of Gaussian white process, we can see that they are quite similar. ***So*** we would not exclude that the vision noise is Gaussian. Therefore qualitatively we can say that the vision noise is white (Oppenheim and Schafer, 1989).

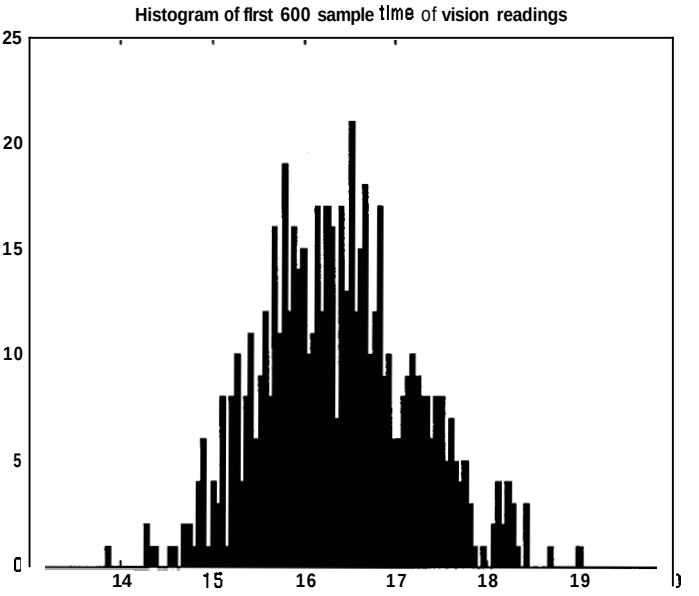


Fig. 6.3.2-4 Histogram of first 600 samples of vision data

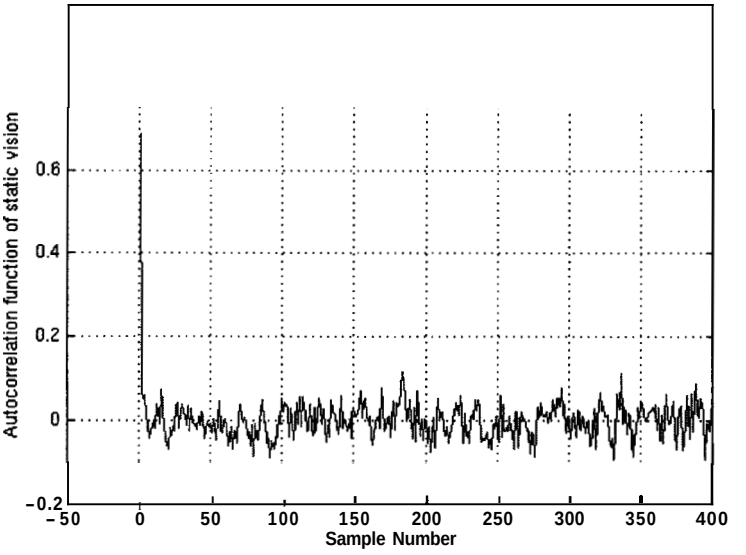


Fig. 6.3.2-5 Autocorrelation estimate of the first 400 samples of vision data

6.3.3 Preliminary Conclusions on the Vision Sensor

From the above analysis, we can draw following conclusions on the vision sensor we used. 1) There is a drift in vision data compared with laser radar data which is more accurate. 2) The variance of vision readings is in proportion **to** the distance it measures. **3.** The vision noise is white and might be Gaussian. ***So*** the PDAF (Alag, 1996) could be a **good** algorithm for sensor validation and fusion when the vision sensor is involved in the measurement.

6.4 Laser Radar Sensor

6.4.1 Introduction to the Laser Radar Sensor

The laser radar sensor which we used in our experiments is a fan beam scan laser radar. The detection area is 350 mrad horizontally and 50 mrad vertically. The maximum range has been set at 100 m so that a stationary object can be detected three seconds before the calculated time of impact when the vehicle's speed is 100km/h. The system has a range resolution of 0.1 m to discriminate rear reflectors of vehicles from road-side reflectors. The range accuracy is ± 1.Om.

6.4.2 Analysis of Laser Radar Data

The original data of Figure 6.4.2.1 are obtained from the same experiment with the vision data. The data is much less noisy than vision data. The estimated data are derived by using the same Kalman Filter mentioned with the last sensor. Note the sensor failed at about sample time 3000-5000, which was caused by someone passing between the two cars at that sample time.

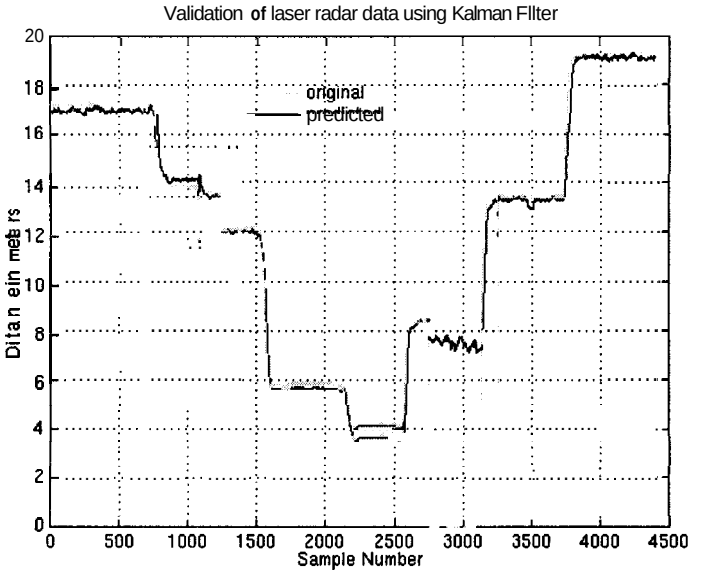


Fig. 6.4.2-1 Validation of laser radar data using KF

Figure 6.4.2.2 is the first part (samples 1-600) of the original data in a different scale. From it we can analyze the noise. The noise appears to be composed of small steps with almost the same magnitude (about 0.1 m). From the histogram of this part of the laser radar data (Figure 6.4.2.3), we can see that it is absolutely different from that of the Gaussian process. We can quickly conclude that the laser radar noise is not Gaussian. Therefore the Kalman Filter would not be very effective in filtering of this kind of noise. But we still use it here in order to use the fusion algorithm PDAF (Alag, 1996).

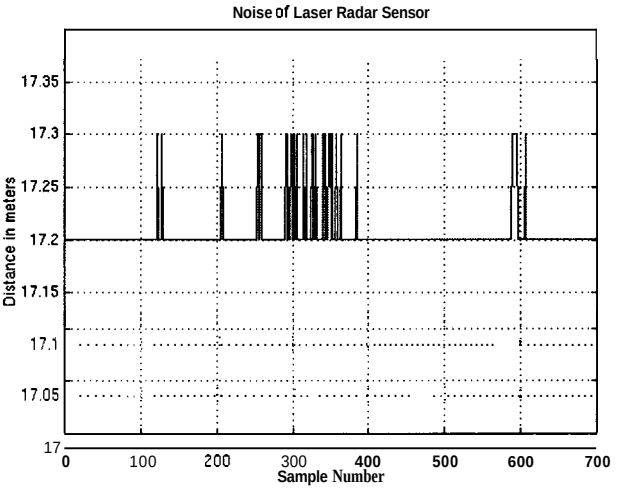


Fig. 6.4.2-2 Noise of laser radar data

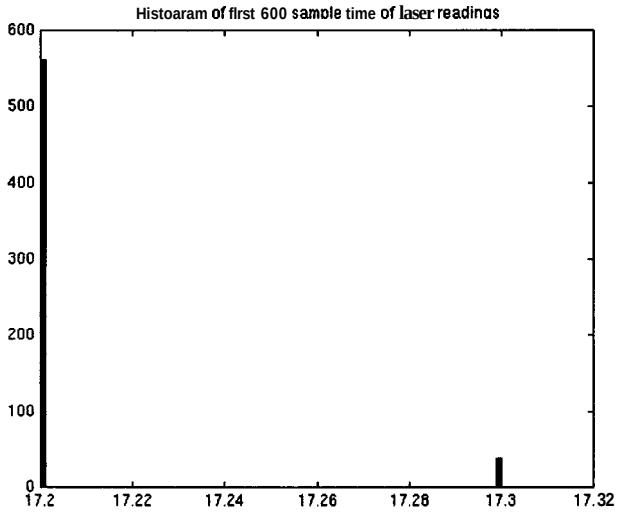


Fig. 6.4.2-3 Histogram of first 600 samples of laser radar data

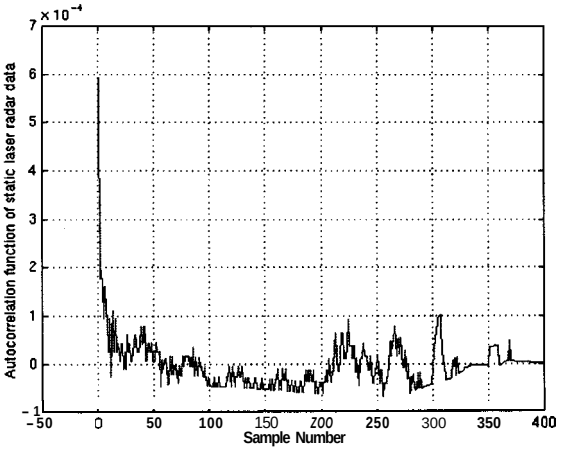


Fig. 6.4.2.4 Autocorrelation estimate of first 400 samples of laser radar data

Figure 6.4.2.4 shows the autocorrelation estimate of samples 1-400 of the laser radar data. The autocorrelations of all of the other sample times are small with respect to that of the first sample time, though they are relatively greater than the vision data after normalization. ***So*** we can say that the laser radar noise is roughly white but not Gaussian.

6.4.3 Preliminary Conclusions for the Laser Radar Sensor

From the above analysis, we get form the following conclusions about the laser radar sensor. 1) It is quite accurate, since the noise magnitude is much smaller than that **of** the vision sensor. 2) The noise is not Gaussian and it is roughly white.

6.5 Fusion of Vision Data and Laser Radar Data Using PDAF

Because the vision and radar readings came at different times and different frequencies, a method of synchronization was used before data validation and fusion took place. The frequency of the vision data is about 5 Hz and that of radar data is about ***5.5*** Hz. We set a standard time stamp of 5 Hz, the most recent readings of both of the data before every time stamp are the synchronized readings which we will use. Note that this is valid only for the static case, the synchronization method might change under dynamic situations.

The fused value of vision and laser radar data are shown in Figure 6.5.1. Here we use the **PDAF** algorithm, which, as the figure illustrates, appears to work very well. The sensor failure of the laser radar is removed from the fused value. This demonstrates once again the importance of redundant information.

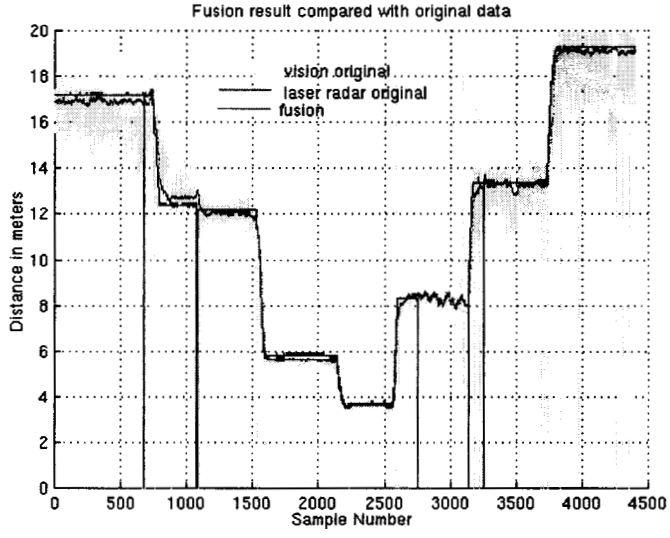


Fig. 6.5-1 Fused data compared with original data

6.6 Summary

From the above analysis, we know that each of the three sensors has some interesting characteristics. The vision data are quite noisy, but since the noise is white and might be Gaussian it could be easily ruled out by a Kalman filter. The laser radar data is relatively accurate, but the noise is not Gaussian so a Kalman filter would not work well; another filter should be designed for this specific sensor. Since the high accuracy and low cost of the GPS sensor make it an appealing

candidate for position sensing in IVHS, we will focus more efforts on the detailed modeling of its noise characteristics in our future research work. Our next step will be to continue GPS noise investigation (especially under dynamic situations) and integration of other sensors into the GPS scheme. In our upcoming GPS experiments, we will use at least one other type of sensor (e.g., radar). And we will evaluate the performance of the integrated scheme using different algorithms. We also plan to continue studying the vision and laser radar sensors if conditions permit.

7. Sensor Validation and Fusion Simulation using SmartPath

In the Partners for Advanced Transit and Highways (PATH) Automated Highway System (AHS) model, platooning is proposed as a method to increase highway capacity and throughput. Effective platooning requires forward-looking vehicle sensors to accurately measure inter-vehicle spacing. As sensor noise and sensor failure is inevitable, sensor validation and fusion techniques have been developed as a means to filter out noise and accommodate for sensor failure.

SmartPath is a computer simulator/animator that was developed to aid in AHS design in the areas of controller design, platooning schemes, maneuvers, network configuration, communication protocols, etc. It addresses the questions of feasibility, safety and performance of the AHS.

Two types of sensor validation and fusion algorithms were integrated for the case of straight following with SmartPath, and their relative performances evaluated. Sensor validation and fusion using the Probabilistic Data Association Filter was shown to provide excellent filtration of the sensor noise modeled when initialized properly. This algorithm had tight constraints on initial conditions and sensor characteristics, as it was found to be very sensitive to initialization. Fuzzy sensor validation and fusion techniques managed to filter out a considerable amount of noise and perform quite effectively, yet not as well as the Probabilistic Data Association Filter. However, fuzzy techniques presuppose no prior knowledge of initial conditions and proved to be very robust over a wide range of conditions.

This integration process allowed for visualization of the algorithm performance and has fostered further development and improvement processes. The resulting SmartPath animations were useful in understanding the physical manifestation of sensor noise and its effect on vehicle control. The integration process is documented and discussed, as well as issues dealing with the utility of SmartPath as an AHS design too. Also, "lessons learned" are discussed as a foundation for development of new, improved, modular simulation design tools.

7.1 Introduction

When most people think of commuting to and from work, certain images usually come to mind: rush hour, traffic congestion, accidents, exhaust, smog, stress, etc. The many millions of Americans that commute to and from work daily can probably relate to these images. The actual fact is that not only is rush hour traffic bad, it is getting worse. In 1995, the average speed of vehicles during peak hours was 35 mph and is expected to drop to 11 mph by the year 2005. This phenomenon is due to population increase as well as metropolitan area expansion.

In efforts to improve the current situation, several solutions have been proposed and implemented to varying degrees. There has been an increased encouragement for individuals to carpool or use mass transit whenever possible. Highways have been widened or newly constructed as funding and real estate have permitted. Also, research has increased worldwide in the areas of improving highway traffic conditions through the use of technology.

In particular, the PATH (Partners for Advanced Transit and Highways) project has proposed an Automated Highway System (AHS) that would increase the throughput of the highway by automating the decision-making process for route-selection and vehicle control - effectively removing control of the human driver from the system. This approach claims dramatic increases in highway capacity, safety and energy efficiency (Varaiya and Shladover, 1991). The key aspect of the PATH-AHS proposal is the use of platooning. Platoons are essentially groups of two to ten vehicles following one another at highway speeds of about 65 mph with intervehicle spacing of approximately one meter. Considering the precision and attention required to maintain such a

platoon, automated control is necessary because a human operator could not react in a timely fashion to emergencies or even routine maneuvers.

This AHS would utilize intelligent vehicles as well as intelligent highways in order to achieve the proposed levels of safety and capacity. Intelligence is defined as the ability to collect information or stimuli from the environment, process the information through thought processes, and respond appropriately. To make a vehicle intelligent, it is necessary to give it the capacity to collect data from the environment, process the data, make a decision, and act on this decision. To collect data, vehicles need sensors. Information from these sensors is then be routed to an on-board computer for processing, at which point the embedded intelligence makes a decision and sends it to the vehicle controller. The vehicle controller processes this information and sends a command to the machine level actuators which perform a maneuver.

This process can become very complicated when many vehicles with multiple sensors are driving on the AHS with various platoons and destinations. Vehicle interactions, communication processes and inevitable hazards can lead to situations that are completely unpredictable. Thus, it is necessary to first simulate this proposed system in order to explore the design space in depth and experiment with system architectures, vehicle controllers, etc. One such simulator, SmartPATH, has been developed to address this need. It utilizes all of the above concepts and outputs a state description of each vehicle as well as a 3D animation of the ***AHS.***

However, a simulator is only as good as the information that goes into it. Therefore it is highly important to make sure that the simulator is as realistic as possible. Developing accurate models and representations of the AHS components is essential to the overall value of the simulator. One key aspect of this theme is the fact that sensors are not perfect. They are subject to the influence of noise sources and, as such, their output is quite uncertain. Thus, a method for simulating sensor noise and effectively filtering it out is necessary in order to improve the realism of the simulator. The sensor validation and fusion techniques developed by this and previous research addresses this issue.

In order to integrate our sensor validation and fusion techniques with the work by others on the PATH projects, we developed simulation modules for the SmartPATH simulator. This simulation allows for evaluation of the methods proposed in a modular manner, including the control architectures and sensor models available from other researchers.

Ultimately, this simulation would provide:

- visualization of the performance of our sensor validation and fusion algorithms and foster further developmental processes.
- continuous improvement to SmartPATH as it currently exists, by making the sensor data more realistic through the application of noise models that replicate experimental data.
- an opportunity for experiential learning, through modular modification of SmartPATH. This process would be documented and potentially utilized by the PATH community for future simulation projects.

7.2 Background

7.2.1 Sensor Validation and Fusion

In the Automated Highway System, sensors are essential in order to provide the vehicle with information about the system states and the vehicle's surroundings. Reliable information allows

the on-board computer to make decisions to ensure safe and accurate vehicle control. Forward looking sensors (radar and sonar) are used by each vehicle to measure relative intervehicle spacing and to detect the presence of obstructions in the road. Wheel speed sensors and accelerometers are used to accurately gauge vehicle speed, roll, pitch and yaw for maneuvers. In addition, GPS receivers can be used to determine absolute position and heading.

Each of these sensors is subject to uncertainty, due to internal and external noise sources, as well as inevitable sensor failure and/or degradation. This uncertain nature is quite unfavorable for the **AHS** due to the requirement of high safety, as human lives are at stake. Thus, sensor data have to be validated before they are used for vehicle control purposes. Often this validation is expressed in terms of probabilities or confidence measures. However, validation techniques are not singularly sufficient in the event of sensor failure. In order to reduce the inevitable risk of sensor failure, redundant sensor schemes are often deployed, with backup sensors providing an increased level of system reliability.

Using redundant sensors means that more than one sensor will constantly gather data. This may prove highly useful, in that each sensor may have a particular and unique set of optimal conditions. For example, a radar sensor may perform best when the distance to its target is within 35 meters, whereas a sonar sensor may only perform well when its distance is within 5 meters of the target. To combine the qualities and attributes of each sensor, it is necessary to "fuse" the data from each sensor into one signal that most accurately represents the actual state of the system. However, since two sensors almost never return the same measurement, it is very difficult to decide how this fusion should be performed. Validation can again be used in order to provide ranking of the incoming signal for fusion purposes.

7.2.1.1 Fuzzy Sensor Validation and Fusion

Fuzzy Sensor Validation and Fusion (Goebel, 1996) uses fuzzy logic techniques to validate and fuse multiple sensor readings. It consists of a fuzzy time series prediction model, fuzzy validation gates, and a weighted average fusion scheme.

Essentially, this algorithm determines a level of confidence in each incoming sensor value based on a validation gate in which sensor readings are expected to lie. Validation curves are dynamically constructed based on sensor characteristic, current sensor measurements, past sensor measurements, and predicted values. Depending on where on the curve a sensor reading actually lies determines its confidence value. Readings that equal the predicted value have a confidence value of one. Readings that vary from the predicted value have lower confidence values with a minimum value of zero. Once a confidence value has been assigned to the reading, this confidence value is used for fusion purposes. If the system has high confidence in a reading, the reading is weighted highly in the fusion algorithm. If the system has low confidence in a reading, the reading is appropriately discounted in the fusion algorithm. The fusion is performed using a weighted average of these confidence values and sensor readings plus a term which includes the predicted value weighted by an adaptive parameter and a constant scaling factor. The fused value is then sent to the vehicle controller as sensor output.

The strength of Fuzzy Validation and Fusion lies in its robustness. It performs acceptably under a wide range of conditions and in the presence of Gaussian as well as non-Gaussian noise. This is largely due to its ability to dynamically adjust its validation gate as well as its use of a separate validation gate for each sensor - based on the sensor characteristics.

7.2.1.2 Sensor Validation and Fusion using the Adaptive PDAF

This methodology uses a combination of several techniques to perform sensor validation and fusion. First, a rule-based system is used to find the operating state of the vehicle. This system builds models that are used to construct validation gates with the Kalman filter scheme. These validation gates are used in concert with the Algorithmic Sensor Validation (ASV) for the validation process. Next, data fusion is performed using the Probabilistic Data Association Filter. It uses probabilities assigned to sensor readings for this process, assuming all readings within the validation gate are from the sensor target. If no readings lie within the validation gate, the reading from the last time step is assumed to be most likely (Alag, 1996). This process produces a fused estimate of the sensor reading to be passed to the vehicle controller. This technique is valuable because it utilizes a knowledge bases system that tries to estimate sensor bias, detect sensor faults and examine sensor performance.

7.2.2 SmartPATH

SmartPATH development was begun in the Spring of 1991 to address the questions of feasiblity, safety and performance in the PATH-AHS proposal. It is intended to be used as a design tool for researchers to test different system architectures, vehicle controllers, communication protocols, maneuver schemes, etc. It is constructed in a modular manner, such that modification is relatively straightforward. As SmartPATH is a graphical simulator, it provides a natural environment to view the simulation, along with a comprehensive, numerical state description of each vehicle.

SmartPATH currently contains three modular components: the simulator, the highway designer, and the animator. These modules are loosely coupled, and therefore open up seemingly endless possibilities for experimentation.

Within the simulator module, SmartPATH models the Shladover-Varaiya (1991) proposed AHS hierarchy. This hierarchy consists of four control layers: network, link, coordination, and regulation. The network and link layers are on the roadside, while the coordination and regulation layers are aboard the vehicle. The network layer is responsible for providing routing information from any point on the AHS to any exit. This routing should involve a minimum of vehicle conflicts, and route the vehicle to its destination using the shortest route and in the least amount of time. The link layer is responsible for the smooth flow of traffic within the individual sections of the AHS. It should balance the traffic among the lanes by providing micro-level routing commands to each vehicle. It uses information about the actual traffic flow in a given highway section, the destination of the vehicle and the route provided by the network layer. The coordination layer receives commands from the link layer about where the vehicle should go, i.e. left, right, straight, etc. It then determines which maneuver to make, and subsequently coordinates the maneuver with neighboring vehicles. The vehicle coordinates maneuvers by requesting permission from its neighbors. When it receives acknowledgment from all neighbors, it sends a command to the regulation layer to perform the maneuver. The regulation layer then implements and performs the maneuver. The network layer and link layer together make up the intelligent highway system. The coordination layer and the regulation layer together comprise the intelligent vehicle.

The highway designer in SmartPATH allows the user to construct an endless array of highway structures, as simple as a straightaway, or as complex as a bowtie intersection. This flexibility allows for various feasibility and safety studies to be performed.

After simulating and designing a scenario, the output can be animated in order to obtain visual feedback. The animator provides the user with a view of the highway and the vehicles in the

simulation. Using a "traffic helicopter", the user is able to follow the vehicles and view the entire scene from virtually any vantage point, angle and perspective.

7.3 Methods

This section is intended to serve two primary purposes:

- to document and detail the specific methods used to perform the integration of sensor validation and fusion techniques with SmartPath, and
- to discuss the evaluation of said validation and fusion techniques in order that SmartPath might be deemed a useful tool for such evaluative purposes.

In reference to documenting the specific methods, the intent is to provide future researchers and SmartPath users with the step-by-step procedures that were undertaken in order to modify and run the simulator. These procedures might be useful in eliminating redundant efforts, as well as providing simulation developers with a "case-study" example of an end-user's product experience. This would potentially aid in improving future simulator designs. In reference to the evaluation of the validation and fusion techniques, it is important to determine how well SmartPath performs. Does it simulate data and information in a predictable, anticipated fashion? Is it accurate and reliable? Can we trust it to be a good representation of the real world? In evaluation of Smartpath's output compared to previously obtained results, we can make a judgment about Smartpath's utility as a design tool.

7.3.1 Specific Methods

This section is not intended to provide a comprehensive guide to SmartPath and its capabilities. It is merely intended to provide documentation for specific methods that were employed to achieve the integration of sensor validation and fusion algorithms with SmartPath. For a detailed guide to SmartPath, see Eskafi and Khorramabadi (1996).

7.3.1.1 SmartPath3.0 System Requirements and Setup

SmartPath3.0 is available in two packages. One contains only the executables and sample data for users who want to modify the configuration of the **AHS,** and not the source code. The other package contains the source code and libraries for users who want to make changes to the architecture, controllers, communication protocols, etc. The simulator module of SmartPath3.0 can be compiled and executed on SunSparc, SunSolaris, and Silicon Graphics (SGI) workstations. The animator uses the Silicon Graphics GL library and can only be run on SGI platforms. The source code can be compiled on either Sun or SGI. To compile the animation module, SGI needs to have IRIX 5.3, X Window System X11R5, Motif 1.2 toolkit, and Performer 2.0.

SmartPath3.0 runs in a distributed fashion to allow the use of multiple processors for compilation. However, if using only one processor, SmartPath3.0 still opens another shell on the same machine to complete the compilation. In order to allow for this, it is necessary to add the machine's name to the .rhosts file in the user's home directory.

7.3.1.2 Background Preparation

As SmartPath3.O is written in C, it is necessary to possess a working knowledge of C in order to perform modifications. It was necessary to thoroughly understand the SmartPath3.0 directory structure, including each directory's purpose as well as the purpose of its included files. **As** SmartPath3.0 consists of 50,000 lines of code, this is not trivial. In addition, it was necessary to

understand each of the sensor fusion and validation techniques (Goebel, 1996; Alag, 1996) in order to successfully integrate them with SmartPath3.0.

7.3.1.3 Module modification and Compilation

The sensor validation and fusion algorithms were translated into C code and inserted as functions within the regulation layer of SmartPath3.0. Since this layer performs all of the control functions of the vehicle, it seemed to be the appropriate place to intercept the sensor data. In addition, it was necessary to impose the noise models, validate and fuse the data, then pass the output to the vehicle controller, also in the regulation layer. The sensor validation and fusion models, as well as the noise models were added to /src/regulation/house_keep.c.

As received, SmartPath3.0 did not initially contain a follower controller. It simply consisted of a follower law that retrieved the velocity and acceleration of the lead vehicle and set them to be its own. This type of follower law was not acceptable for the purposes of this project, so a new follower law was developed. Not only would this effort serve the purpose of providing a follower law, but it also served as a testing ground to determine how easy it would be to modify the vehicle control laws within SmartPath. A PID controller was developed by Jiangxin Wang, and added to /src/regulation/regfollow.c in order to demonstrate how to modify and test controller designs. This controller was tested and deemed quite satisfactory for vehicle following.

Function prototypes were added to one of the regulation layer header files, /src/regulation/regulation.h. Additionally, global variables were added at the top of /src/simulation/main.c, the main program for the simulator. For specific modifications, see Appendix **B**.

In order to compile the program, the make command was invoked in the top level of the directory, using "make -f Makefile.sun".

7.3.1.4 Configuration Files

The configuration files are stored in the /data directory. These files give the simulator information about the car type, highway structure, simulation parameters, intraplatoon distance, and animation parameters. These files were modified in order to set up an AHS with two vehicles, a leader and a follower, traveling at 25 m/s with an intervehicle distance of 4m. This setup was chosen in order to allow for a simple, focused view of the vehicle interaction. The configuration files were modified using a text editor and saved as "sample.config".

The simulation was initiated by invoking the command "$SMARTPATH/bin/sun/sm_sim /data/sample", where "sample" is the prefix for the configuration file of interest, namely "sample.config". The program simulated the data and produced output as "sample.state" in the form of an ascii file that is a comprehensive state description of each vehicle at each time step.

7.3.1.5 Data analysis

Matlab scripts were developed that allowed for examination and comparison of the simulator output. These scripts essentially accessed the simulator output (sample.state) and created a temporary array. Specific information about the vehicle ID number and its related speed and position at each time step was stored in the temporary array. This array of data was plotted to display information about the intervehicle spacing between the two vehicles as well as their respective velocity traces. Additionally, the root mean square (rms) error as well as the sum square error (sse) were calculated within these scripts.

7.3.1.6 Animation

The animation was performed on an SGI workstation with the system requirements previously mentioned. As the simulation was performed on a Sun workstation, it was necessary to transfer the "sample.state" and "sample.cars" files to an SGI via file transfer protocol (ftp). It was necessary to transfer all files that were supportive of the animation process. These files included /bin/sgi/sm_anim, /libraries/models, /libraries/messages, and /libraries/sgi. In order to start the animation, the command ./bin/sm_anim ./data/sample was used. The animator allows a great deal of flexibility and viewing manipulation. For a more comprehensive guide, the reader is referred to the SmartPath3.0 User's Manual.

7.3.2 Evaluation of Sensor Validation and Fusion Techniques in SmartPath

In order to evaluate the effectiveness of SmartPath as a simulation tool, it is useful to look at the objectives of SmartPath. It sets out to provide an AHS design environment in which controllers, decision-making algorithms, sensor models, etc. can be developed. In order for SmartPath to be effective at this, it needs to accurately represent real-life. By comparing the expected performance of both sensor validation and fusion techniques (gathered from previous experiments) with their performance in SmartPath, we can examine how well SmartPath replicates actuality.

Experiments were conducted to evaluate the performance of vehicles in the AHS in response to 1)perfect sensor information, 2) radar sensor noise, 3) fuzzy validation and fusion, and **4)** validation and fusion with the PDAF. These experiments were conducted at an intervehicle spacing of **4** meters. Although both radar and sonar noise models were employed, the sonar model is not very noisy at intervehicle distances of less than 5 meters. Since the actual vehicles used in the PATH project currently rely exclusively on the radar sensor, it was decided to look more closely at radar noise.

Each of these cases required a separate simulation and animation. The simulated data was plotted using the aforementioned Matlab scripts. In order to provide a quantitative basis for comparison, the root mean square (rms) error of each data set was determined. In animating the data, particular attention was given to the dynamic behavior of the follower vehicle. The effect of the radar noise on the ride quality of the vehicle was noted, as well as the ability of the validation and fusion algorithms to filter out this noise and subsequently improve ride quality.

7.4 Results

7.4.1 Quantitative Results

Four cases were examined:

I. Perfect Sensor case : assumes no sensor noise
11. Radar Noise case : adds a radar noise model to the sensor
111. FUSVAF case : uses Fuzzy Sensor Validation and Fusion
IV. Adaptive PDAF case : uses the Adaptive Probabilistic Data Association Filter for sensor validation and fusion.

These simulations were run using an intervehicle spacing of **4** meters and a target velocity of 25 m/s (65 mph). The data output was examined in two fashions. First, root mean square (rms) analysis was performed in order to quantify the error. The data was also plotted as velocity vs. time and intervehicle spacing vs. time; **as** these parameters are crucial to maintaining a platoon, they were closely examined. Secondly, the data was animated and the dynamic behavior of the follower vehicle was observed.

For the Perfect Sensor case, the intervehicle spacing had an rms error of 0.0m (Fig. 7.4-1). This is to be expected, as there is no noise in the perfect sensor case. The simulator simply passes the exact information about the position of the lead vehicle to the follower vehicle's "sensor". It assumes no noise.

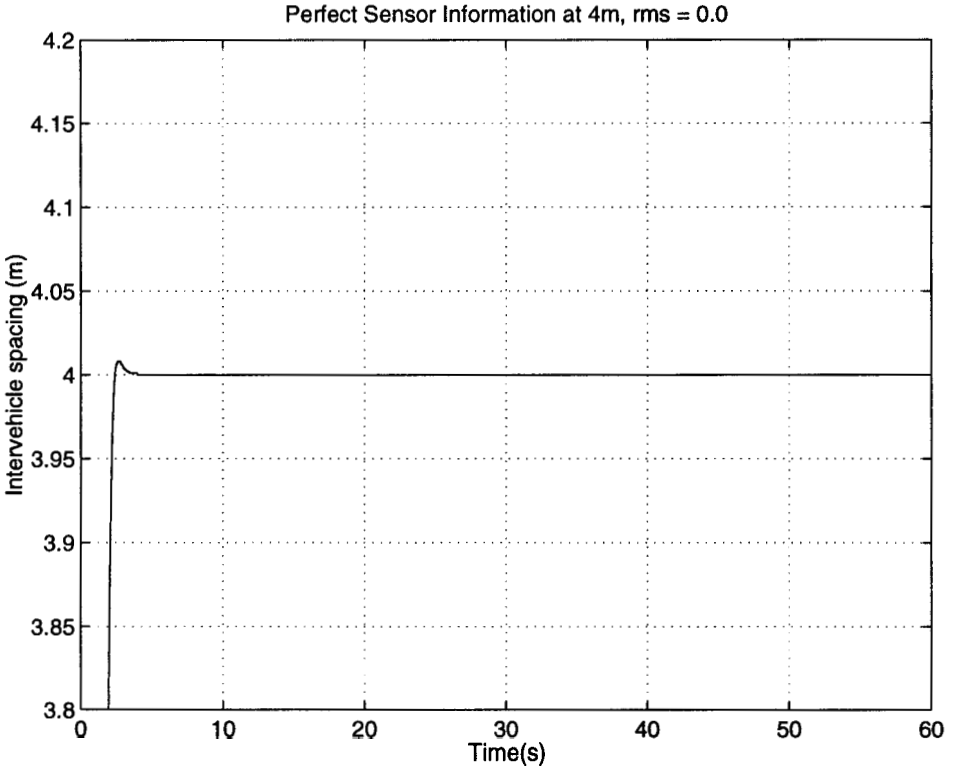


Fig. 7.4-1: Perfect sensor information at 4m, rms error = 0.0021m

Also for the perfect sensor case, the follower vehicle velocity exhibited an rms error of 0.0 m/s (Fig. 7.4-2). Again, the simulator assumes there is no noise in the transmission of the data.

In the Radar Noise case, the intervehicle spacing rms error increased to 0.1509 m (Fig. 7.4-3). This demonstrates how the follower vehicle's position is affected as it tries to respond to the noisy radar sensor signal. The vehicle's motion and subsequent position are, likewise, noisy. The radar noise model was developed from experimental data obtained from actual PATH vehicles at the Richmond Field Station. It assumes Gaussian noise.

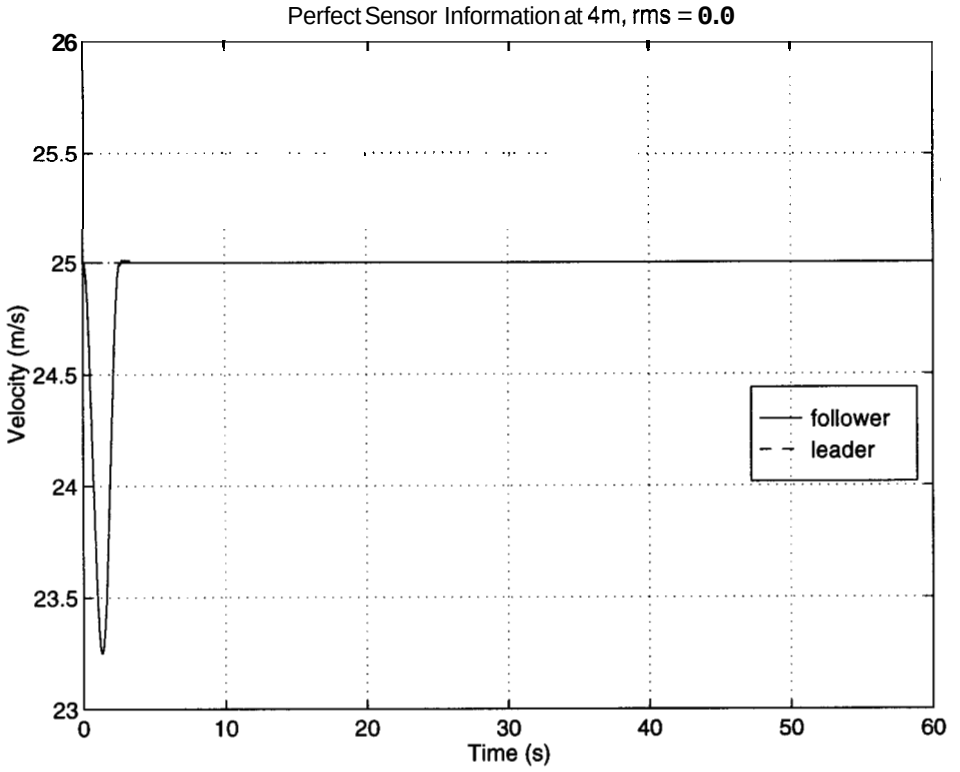


Fig. 7.4-2: Perfect sensor information at 4m, rms error =0.0107m/s

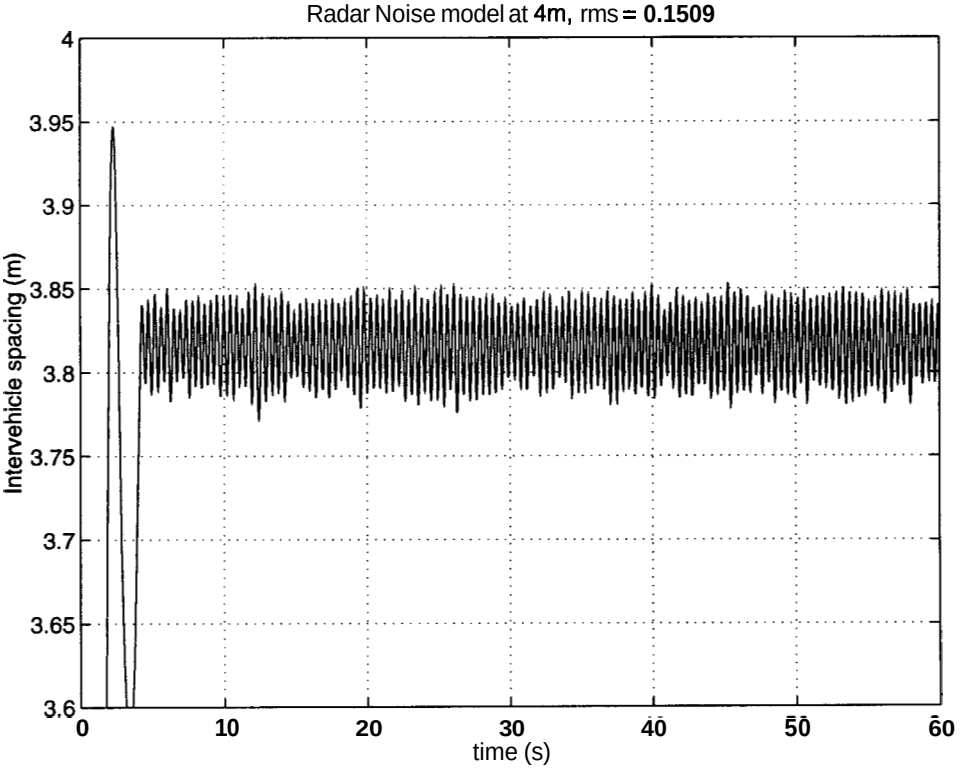


Fig. 7.4-3: Radar noise model added at 4m, rms error = 0.1509m

Also for the Radar Noise case, the rms error of the follower velocity increased to 0.2276 m/s (Fig. 7.4-4). As the vehicle tries to maintain an intervehicle spacing of **4m,** it responds to the noisy signal **as** shown in the velocity trace.

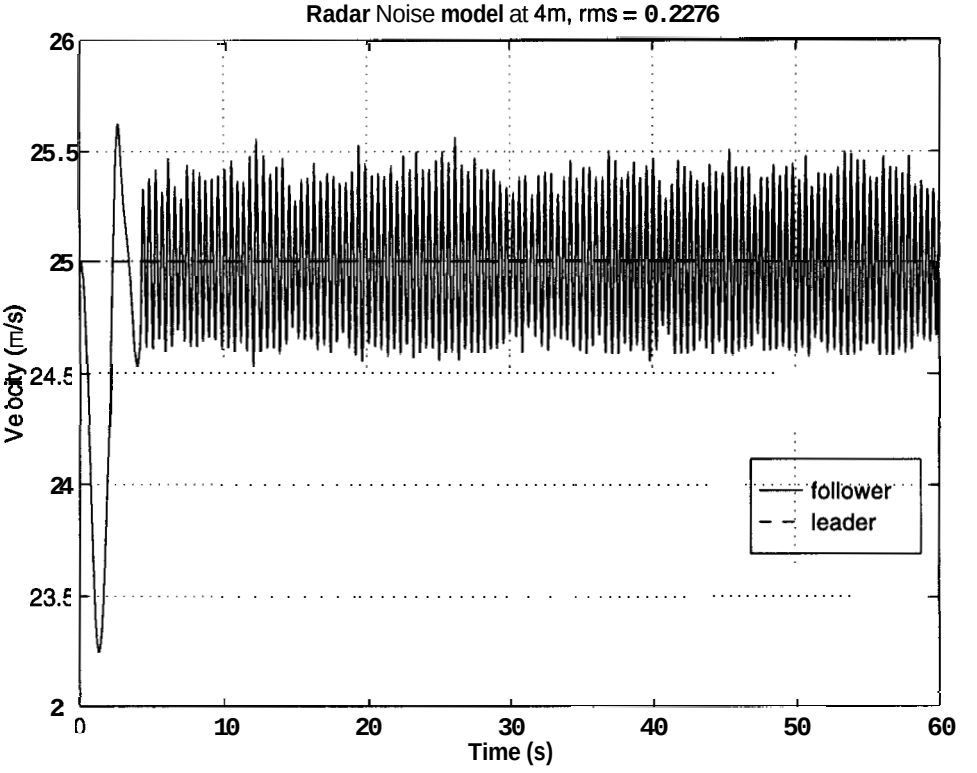


Fig. 7.4-4: Radar noise model added at 4m, rms error = 0.2276 m/s

In the FUSVAF case, fuzzy sensor validation and fusion techniques decreased the rms error of the intervehicle spacing to 0.1358 m (Fig. 7.4-5). **By** filtering out the radar sensor noise, the follower vehicle manages to smooth out its motion.

Similarly, when Fuzzy sensor validation and fusion was applied, the rms error of the follower velocity decreased to 0.0302 m/s (Fig. 7.4-6). This, again, is due to the algorithm's ability to filter out the noise from the radar sensor.

For the Adaptive PDAF case, the rms error of the intervehicle spacing decreased further to 0.0022 m (Fig. 7.4-7). The Adaptive PDAF uses a learning algorithm to allow it to filter out the noise more effectively.

Also, the Adaptive PDAF reduced the rms error of the follower velocity to 0.0094 m/s (Fig. 7.4-8).

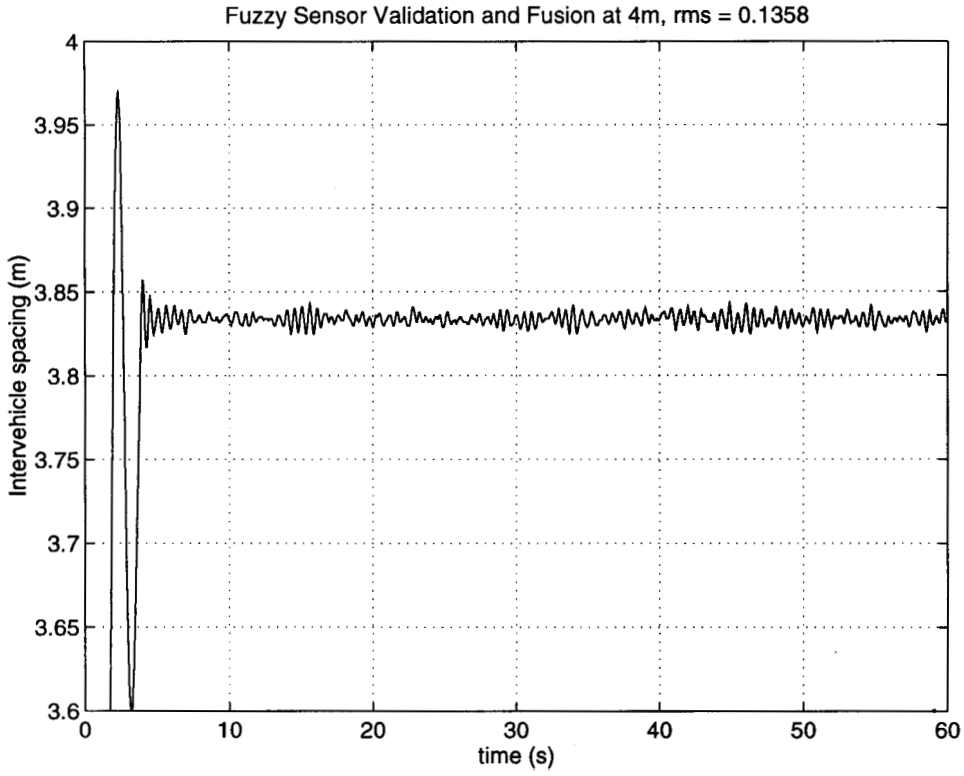


Fig. 7.4-5: Fuzzy sensor validation and fusion at 4m, rms error = 0.1358 m

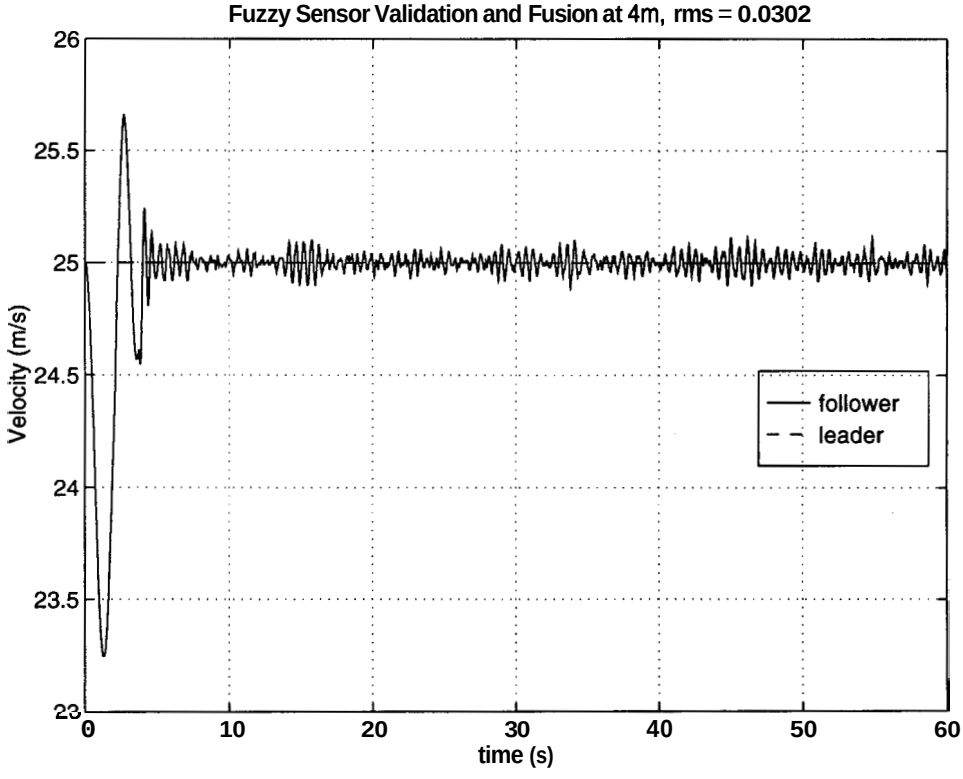


Fig. 7.4-6: Fuzzy sensor validation and fusion at 4m, rms error = 0.0302 m/s

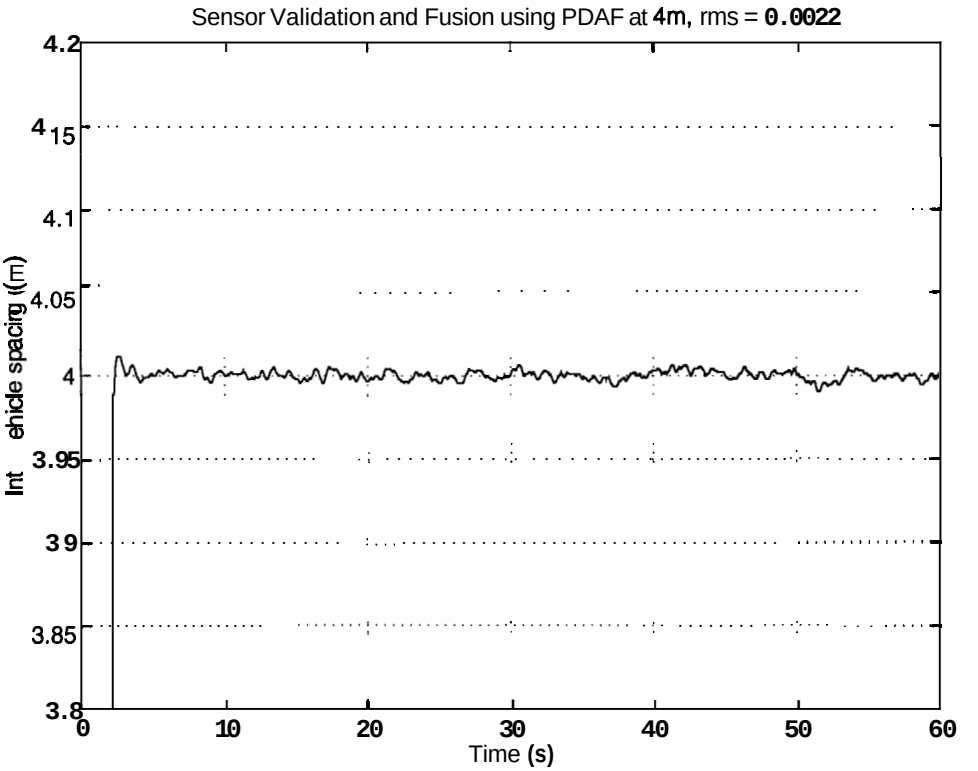


Fig. 7.4-7: Sensor Validation and Fusion using PDAF at 4m, rms error = 0.0022m

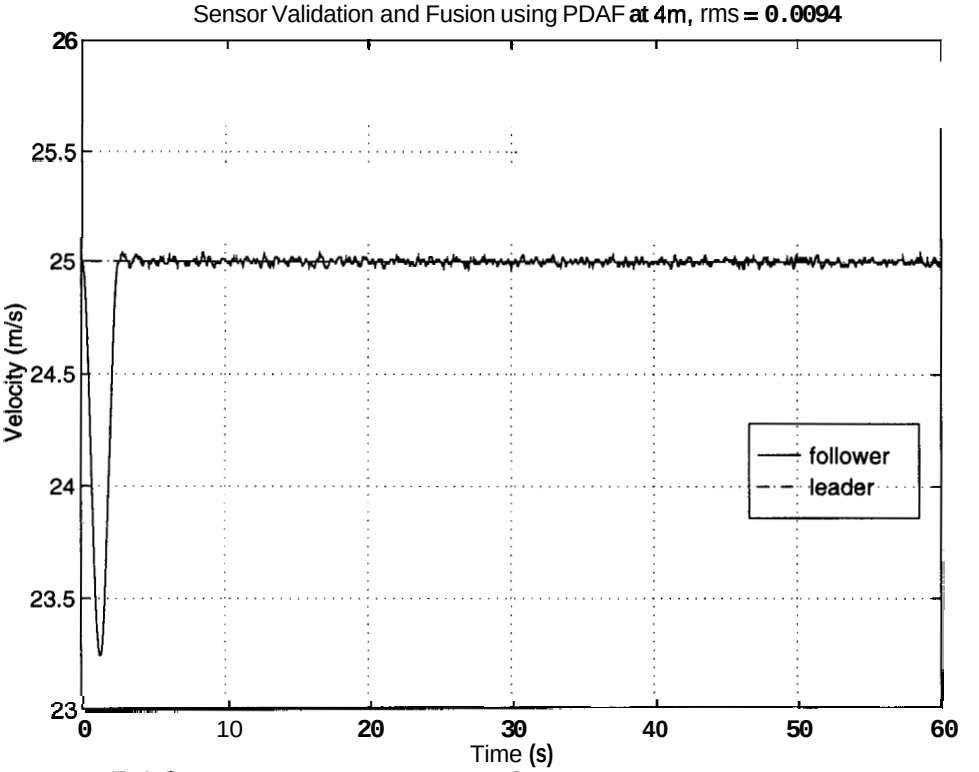


Fig. 7.4-8: Sensor validation and fusion using the PDAF at 4m, rms error = 0.0094 m/s

7.4.2 Qualitative Results

In animating the Perfect Sensor case, the follower vehicle behaved as one might expect: its ride was smooth and controlled. As there was no noise present in the signal, no unusual vehicle response was noted. The spacing between the vehicles remained constant as it appeared in the Intervehicle distance plot (Fig. 7.4-1). In the Radar Noise case, the follower vehicle exhibited a pulsing, jerking motion, with unacceptable ride quality. The distance between the two vehicles varied to a noticeable degree. Considering the nature of the noise demonstrated in the accompanying plots (Figs. 7.4-3 and 7.4-4), this behavior was expected. In animating the FUSVAF case, the ride quality was improved considerably. There was still some slight pulsing motion, but not nearly as severe as the Radar Noise case. The ability of the FUSVAF algorithm to filter out the radar noise was visually evident. In animating the Adaptive PDAF case, the ride quality was vastly improved over the Radar Noise case. The vehicle motion was smooth and controlled and very much like the perfect sensor information case. As the rms error of the Intervehicle distance of the PDAF case was $0.0022m$ (Fig. 7.4-7), the vehicle exhibited virtually no visual indication of this error.

7.5 Discussion

The results of the simulation and animation procedures generally suggest that the two sensor validation and fusion techniques behave as expected. In the presence of radar and sonar noise models, they both are capable of filtering out the noise to varying degrees, as well as fusing multiple readings.

According to the data, the probabilistic data association filter managed to filter out almost all of the sensor noise, nearing the level of perfect information. The PDAF uses a Kalman filter, which operates best in the presence of Gaussian noise. As the radar noise model utilizes Gaussian noise, the PDAF was operating under optimal conditions. However, the PDAF is extremely sensitive to initial conditions. It has to be initialized accurately, or else its performance decreases drastically.

The Fuzzy algorithm didn't filter out the noise as well as the PDAF, but it did filter out a significant amount. The true value of the fuzzy filter lies in its robustness. It has no restrictions on noise type or initialization. It performs well over a wide range of conditions and is much less sensitive to small variations in initialization.

SmartPath was quite useful in visualizing the simulated data. It made the numbers and plots 'come to life' in the form of 3D animation. The animator provided insight into the physical manifestation of noise in the automatic control of the vehicles. One could readily understand what these variations in sensor readings and subsequent vehicle responses meant in terms of physical space. After using the animator in tandem with the simulator, one can easily relate a physical image with rms error values and velocity traces.

7.6 Conclusions and Future Research

7.6.1 Conclusions

From the results obtained, it seems that SmartPath is a valuable tool for simulation and animation of AHS concepts and scenarios. Although not very well documented nor user-friendly, SmartPath is quite modular and receptive to modification. It is quite realistic, as it simulates experimental data quite well, and offers a natural 3D environment for viewing the vehicle interactions. SmartPath helps bring data 'to life' by offering a connection between vehicle motions and controller commands.

Also, the results of the algorithm comparison suggest that both the PDAF and Fuzzy techniques are quite useful in filtering radar sensor noise. The PDAF algorithm is more precise, yet more sensitive to initialization parameters and noise type. The Fuzzy algorithm is less precise, but much more robust to a wide range of initialization parameters and noise models. However, both algorithms offer very useful methods for dealing with inherent sensor uncertainty in the AHS, as well as other applications.

7.6.2 Future Research

It would be quite useful to perform experiments using sensor validation and fusion techniques with different vehicle maneuvers. Joins, splits, lane changes, etc. would likely offer some interesting insight as to the impact of sensor noise on a wide range of vehicle maneuvers. In a join maneuver the sensors must be active over a wide range of distances, because a vehicle approaches a platoon from hundreds of meters away and must adjust its relative velocity to achieve a specified intraplatoon distance. Sensors have different noise characteristics at various distances from the target, and it would be useful to note the effect of this noise on the join maneuver.

It would also be useful to take the positive attributes of both the fuzzy and PDAF techniques, and combine them to form a more robust, accurate validation and fusion algorithm. The fuzzy algorithm could be used to begin the validation process by getting the system 'on-track, and the PDAF algorithm might then refine the validation and fusion process with more precise, accurate predictions. An algorithm of this sort might be used as a complete "learning" algorithm that is applicable to any situation. It would potentially not have to be initialized, due to its ability to learn the behavior of the system and fine tune its predictions with each successive time step.

8. Summary, Conclusions and Recommendations

Past research has shown that exploratory sensors for IVHS exhibit a range of sensor failure behaviors. Although the integrity of these sensors will improve with future research and refinement, our research makes it clear that some degree of uncertainty will be inevitable in the class of sensors being considered and in the uncertain environmental conditions possible. Control systems that rely on an assumption of "perfect sensors" may result in control actions which are unsafe for passengers of the IVHS. We have developed a five-module approach to intelligent diagnosis based on validated and fused sensor data. The modules are: 1) Sensor Validation, 2) Sensor Fusion, 3) Fault Detection, **4)** Hazard Analysis, and 5) Safety Decision Maker. The first two modules and, to a certain degree, the third module, were addressed in previous funding projects (MOU 132; Agogino et al., 1995 and MOU 157; Agogino et al., 1997).

8.1 Summary

This report describes extensions to previous work and introduces the remaining modules, Fault Detection, Hazard Analysis, Safety Decision Maker, which take the validated and fused sensor data as input and gives probabilities and other multivariate measures of hazard likelihood as output. Failures can to a degree be predicted and actions to situations sensitive to certain sensor readings can be taken in advance not as a reaction to an already hazardous situation but as a preventive measure. We augmented previous work in sensor validation and failure by developing other techniques — such as fuzzy data validation, fuzzy fusion, and extensions of Bayesian networks. All of these techniques are designed to be used to complement the analytical methods for fault detection that are being developed by other PATH researchers. The combination of both fuzzy and probabilistic strategies allows the evaluation of all available information and knowledge of the system for fault detection such as degree of aging, the operational environment (e.g., the vehicle state whether lead vehicle or follower vehicle etc.).

This report also analyzed potential sensor failures and related hazards to recommend appropriate actions. Failures were identified and related to their cause and their effects on the system and classified accordingly (e.g., total or incipient). Our past work has clearly shown that various sensors react in different ways under the influence of different operating conditions such as fog or rain. We carried out a failure mode effect analysis for the longitudinal sensor in order to systematically capture the effects and reasons for failures and to match the signature of faulty sensor readings with specific failures.

The state of the IVHS system on the platoon level was modeled as an optimization problem, with possible states of each of the vehicles defined over the feasible stochastic search space. The states of individual vehicles in a platoon were the design variables in the optimization process. The objective function in this case was multi-attribute consisting of a list of possible hazards and the probability of each occurring obtained from the Intelligent Decision Module. With our ongoing work in PATH we will continue to test, characterize, and compare sensors operating under diverse and adverse conditions including new ranging sensors entering the scene, e.g. GPS sensors.

8.2 Conclusions and Recommendations

A summary of conclusion and recommendations follows. Some of these recommendations will be implemented in MOU-322 for the PATH project titled: "Aggregation of Direct and Indirect Positioning Sensors for Vehicle Guidance."

8.2.1 Fuzzy Sensor Validation, Fusion and Fault Detection

The fuzzy diagnosis framework we developed has proven to be a fast and accurate way to trace failures and their source, (if modeled) from observed symptoms in simulation studies. For on-line diagnosis in automated highway systems, its fast computation makes it possible for failures to be quickly detected and followed by timely reaction. Trends of failure may also be detectable, allowing prediction of failure in the future. Future research should focus on effective means of utilizing the results of fuzzy validation and fusion with diagnosis and decision making strategies.

8.2.2 Bayesian Sensor Validation, Fusion and Fault Detection

We also developed a comprehensive Bayesian methodology for intelligent sensor measurement validation, fusion, and sensor fault detection. The methodology is effective in detecting sensor faults, combining information from a number of sources such as sensor characteristics and the immediate history of the variable that is being monitored. This work addresses the very important problem of distinguishing between a sensor failure and a system failure for complex systems. The proposed four step methodology of redundancy creation, state prediction, validation and fusion, and fault detection can detect subtle sensor failures such as drift in mean and degradation of the sensor over time.

We have also illustrated how probabilistic reasoning can be used to handle the uncertainty inherent in the residue processing process. We have illustrated how to refine and build an appropriate belief network structure and how the conditional probabilities can be estimated both analytically and from learning from data (for which we have presented a new method which is applicable to only a special class of belief networks). We have illustrated how dynamic belief networks can be used for forecasting. We have derived a recursive formula for the on-line adaptation of the state evolution model. Based on studies utilizing both real and simulated data we have found the combined sensor fault detection methodology to be effective in both detecting and forecasting potential sensor faults.

8.2.3 Comparison of Fuzzy and Bayesian Techniques

In contrast to the Bayesian approach, the fuzzy approach avoids assumptions of failure independence and of relative frequency of disorder occurrence. Unlike the Bayesian approach, which has difficulties in detecting multiple simultaneous failures, the fuzzy approach is able to consider all interrelationships and prioritize major failures. The fuzzy approach tends to outperform the Bayesian approach on failure modes which are not well documented with a statistical history, particularly if the distributions largely deviations from a Gaussian model. Similarity between the two methods occurs in that links in the causal network represent causal strengths for failure-symptom relations.

Once the **IVHS** sensors and control systems are better defined, additional expert knowledge about the behavior of multiple fault-symptom should be codified and incorporated into the logic. This would require a symptom combination that differs from the default model, which would not be able to detect two faults which might (partially) cancel their symptoms.

The Bayesian approach, however, could outperform the fuzzy methods **as** more data are collected and as the system models are better defined and validated. Future research should focus on how these two approaches — each with advantages over the other in some circumstances — could be implemented in a complementary fashion to form a more robust, accurate validation, fusion and fault detection framework. The fuzzy algorithm could be used to begin the validation process by getting the system 'on-track, and the **PDAF** algorithm might then refine the validation and fusion process with more precise, accurate predictions. An algorithm of this sort might be used **as** a complete "learning" algorithm that is applicable to any situation. It would potentially not have to be

initialized, due to its ability to learn the behavior of the system and fine tune its predictions with each successive time step.

8.2.4 Hazard Analysis

A prototype Fault Tree Analysis of component faults was developed. The fault characteristics provide useful qualitative insight as to the impact on the controller when a component failure occurs. This information can be used to predict which hazards are imminent. The results gathered from the Fault Tree analyses show that the hazard definitions are far too broad to provide useful information in failure analysis and future work should include redefining the set of existing hazards. Similarly, the trees could not be analyzed quantitatively using Boolean algebra because of the large number of undeveloped events and the lack of data corresponding to those events. Although the Fault Trees developed for the radar and sonar sensors are very limited and exclusive in nature, they serve as a springboard to other applications. As more definition is given to the intelligent vehicle highways systems in the PATH project, further development of the fault trees is recommended.

8.2.5 Decision Analytic Approaches to Supervisory Control

We introduced a framework to enable the automated vehicle to choose an optimal maneuver or strategy in hazardous conditions. We discussed hazard diagnosis and results from extended analysis of earlier tests performed under hazardous conditions. This analysis yielded a table of characteristics differentiating the four hazardous conditions (rain, fog, debris, no hazard) under consideration. We also discussed our implementation of EDCRASH in determining vehicle trajectories after impact. Environmental inputs to EDCRASH were briefly described and outputs detailing impact on the vehicles were listed. The effects of impact were then assessed according to statistics from the National Transportation Highway Traffic Administration, resulting in objective figures for bodily and property damages. Finally, we described the overall optimization problem as a minimization of damages subject to a variety of constraints. We used off-line optimization techniques that are too slow to run in real time within the AVCS framework. We recommend that a real-time version can be developed by "compiling" out the optimal control strategies.

This research assumed a limited case in which we restricted our longitudinal distance assessment to two sensors and in which we considered only three independent hazards. Because of the restrictions, we were able to solve exhaustively, using stochastic optimization. More difficult cases will also yield interesting results in future research when we extend our techniques to scenarios with multiple sensors and dependent hazards.

In future work we recommend the use of influence diagrams (Bayes' networks with the addition of decision/control and cost/value nodes) for decision making to avoid or avert potentially dangerous states. These decisions are to be optimized with respect to safety, low jerk, and smooth riding criteria. Integration with results of our fuzzy techniques should also be investigated.

8.2.6 GPS, Vision and Laser Radar Sensors

From the above analysis, we know that each of the three sensors has some interesting characteristics. The vision data are quite noisy, but since the noise is white and might be Gaussian it could be easily ruled out by a Kalman filter. The laser radar data is relatively accurate, but the noise is not Gaussian so a Kalman filter would not work well; another filter should be designed for this specific sensor. Since the high accuracy and low cost of the GPS sensor make it an appealing candidate for position sensing in IVHS, we will focus more efforts on the detailed modeling of its noise characteristics in our future research work. Our next step will be to continue characterizing GPS noise (especially under dynamic situations) and integrating other sensors into the GPS

scheme. In our upcoming **GPS** experiments in MOU-322, we will use at least one other type of sensor (e.g., radar). And we will evaluate the performance of the integrated scheme using different algorithms. We also plan to continue studying the vision and laser radar sensors if conditions permit.

8.2.7 Simulations

There are many competing simulation program being developed for the **PATH** project. When these simulators are better integrated, it would be quite useful to redo our simulations and perform experiments using sensor validation and fusion techniques with different vehicle maneuvers. Joins, splits, lane changes, etc. would likely offer some interesting insight as to the impact of sensor noise on a wide range of vehicle maneuvers. In a join maneuver the sensors must be active over a wide range of distances, because a vehicle approaches a platoon from hundreds of meters away and must adjust its relative velocity to achieve a specified intraplatoon distance. Sensors have different noise characteristics at various distances from the target, and it would be useful to note the effect of this noise on the join maneuver. Finally, such simulations would help us develop the best hybrid architecture for combining fuzzy, probabilistic and decision-analytic approaches to sensor validation, fusion, fault detection and recovery.

9. References

Agogino, A. M., Srinivas, S. and Schneider, S. 1988. Multiple Sensor Expert System for Diagnostic Reasoning, Monitoring and Control of Mechanical Systems. *Mechanical Systems and Signal Processing*, vol. 2, no. 2, pp. 165-185.

Agogino, A.M., Goebel, K., and Alag, S. 1995. Intelligent Sensor Validation and Sensor Fusion for Reliability and Safety Enhancement in Vehicle Control. MOU132, Final Report, UCB-ITS-PRR-95-40, *California PATH Research Report.*

Agogino, A.M., Goebel, K., and Alag, S. 1997. Intelligent Sensor Validation and Fusion for Vehicle Guidance Using Probabilistic and Fuzzy Means. MOU157, Final Report, UCB-ITS-PRR-D97-29, *California PATH Research Report.*

Alag, S., 1996. *A Bayesian Decision-Theoretic Framework for Real-Time Monitoring and Diagnosis of Complex Systems: Theory and Application*, PhD Thesis, Department of Mechanical Engineering, University of California at Berkeley.

Alag, S. and Agogino, A. 1995. Intelligent Sensor Validation for On-Line Process Monitoring, Diagnosis and Control, published in *Final Report MICRO Projects* 1993-94, working paper #95-0302-P, Berkeley Expert Systems Technology Lab., UC Berkeley.

Alag, S., Goebel, K., and Agogino A. 1995. A Framework for Intelligent Sensor Validation, Sensor Fusion, and Supervisory Control of Automated Vehicles in IVHS. *Proceedings of the ITS America Annual Meeting*, Washington, D.C.

Alag, S., Goebel, K., and Agogino A. 1995. Intelligent Sensor Validation and Fusion used in Tracking and Avoidance of Objects for Automated Vehicles. *Proceedings of the ACC 1995*, Seattle.

Bar-Shalom, Y. and Fortmann, T. E. 1988. *Tracking and Data Association*. Boston. MA: Academic.

Barton, D. et al. 1991. Radar Evaluation Handbook. ANRO Engineering, INC.

Basseville, M. and Nikiforov, I. 1993. *Detection of Abrupt Changes: Theory and Application*. Prentice Hall Information and System Science Series, NJ.

Bhattacharyya, A. 1943. On a Measure of Divergence between two Statistical Populations Defined by Probability Distributions, *Bull. Calcutta Math Soc.*, vol. 35, pp. 99-109.

Bellm, D. 1995. *Characterization of the Sonar and Radar Distance Sensing Devices Under Sub-Optimal Operating Conditions for the California Partners for Advanced Transit and Highways.* Report for Master's Thesis at UC Berkeley.

Brown, C., Durrant-Whyte, H., Leonard, J., Rao, B. and Steer, B. 1992. Distributed Data Fusion Using Kalman Filtering: A Robotics Application. *In Data Fusion in Robotics and Machine Intelligence*, ed. Abidi, M. A., and Gonzalez, R. C., Chap. 7, pp. 267-309, Academic Press.

Buntine, W. L. 1994. Operations for Learning with Graphical Models. *Journal of Artificial Intelligence Research*, 2.

Clark, R. N. 1989. State Estimation Schemes for Instrument Fault Detection. Chapter 2, Patton, Frank and Clark, *Fault Diagnosis in Dynamic Systems: Theory and Application*, Prentice Hall.

Dalton, R. 1993. Towards a Theory of Fuzzy Causality for Diagnosis. *Information Sciences*, Vol. 80.

DeGroot, M. 1970. *Optimal Statistical Decisions.* McGraw-Hill, New York.

Ding, X. and Frank, P. M. 1991. Frequency Domain Approach and Threshold Selector for Robust Model-Based Fault Detection and Isolation, *SAFEPROCESS '91*, Baden-Baden.

Driver, E. And Morrell, D. 1995a. Implementation of Continuous Bayesian Networks Using Sums of Weighted Gaussians. Proceedings of *Uncertainty in Artificial Intelligence UAI-95.*

Driver, E. And Morrell, D. 1995b. Implementation of Continuous Bayesian Networks Using Sums of Weighted Gaussians. ***Technical Report TRC-SP-DRM-9501***, Electrical Engineering Department, ASU.

Emami-Naeini, A. 1988. Effect of Model Uncertainty on Failure Detection: the Threshold Selector. ***IEEE Transaction on Autom. Contr.,*** Vol. 33, no. 2, pp. 1106-1115.

Eskafi, F. H. 1996. ***Modeling and Simulation of the Automated Highway System.*** Ph. D. Thesis. Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

Eskafi, F. and Khorramabadi, D. 1996. ***SmartPath3.0 User's Manual.*** Department of Electrical Engineering and Computer Sciences, University of California at Berkeley.

Frank, P. M. 1990. Fault Diagnosis in Dynamic Systems Using Analytical and Knowledge-based Redundancy- A Survey and Some New Results. ***Automatica***, vol. 26, no. 3, pp. 459-474.

Frank, P. M. 1993. Advances in Observer-Based Fault Diagnosis. ***Proc. of the Conf. TOOLDIAG' 93,*** Toulouse, France.

Frank, P. M. 1994. Application of Fuzzy Logic to Process Supervision and Fault Diagnosis. ***IFAC Fault Detection, Supervision, and Safety for Technical Processes,*** Finland.

Geiger, D. and Heckerman, D. 1994. Learning Gaussian Networks. ***Technical Report MSR-TR-94-10,*** Microsoft Research, Advanced Technology Division, Microsoft Corporation, WA.

Godbole, D., and Lygeros, J. 1993. Longitudinal Control of the Lead Car of a Platoon. ***Technical Report PATH Memorandum 93-7.*** Institue of Transportation Studies. University of California at Berkeley.

Goebel, K. 1996. ***Management of Uncertainty in Sensor Validation, Sensor Fusion, and Diagnosis of Mechanical Systems Using Soft Computing Techniques,*** PhD. Thesis, Department of Mechanical Engineering, University of California at Berkeley.

Goebel, K. and Agogino, A. 1996. An Architecture for Fuzzy Sensor Validation and Fusion for Vehicle Following in Automated Highways. ***Proceedings of the 29th ISATA,*** Florence, Italy.

Gu, Y., Peiris, D., Crawford, J., NcNicol, J., Marshall, B. and Jefferies, R. 1994. An Application of Belief Networks to Future Crop Production. In ***Proceedings of the Tenth Conference on Artificial Intelligence for Applications,*** San Antonio, TX, pp. 305-309. IEEE Computer Society Press, Los Alamitos, CA.

Heckerman, D. 1995. A Tutorial on Learning Bayesian Networks. ***Technical Report MS-TR-95-06,*** Microsoft Research Advanced Technical Division.

Heckerman, D., Horvitz, E. and Nathwani, B. 1992. Toward Normative Expert Systems: Part I. The Pathfinder project, ***Methods of Information in Medicine,*** vol. 31, pp. 90-105.

Hedrick, J. K., McMahon, D., Swaroop, D., Garg, V., Gerdes, J., Maciuca, D., Blackman, T., and Yip, P. Longitudinal Control Development for IVHS Fully Automated and Semi-Automated Systems-Phase 1. ***California PATH Research Report*** UCB-ITS-PRR-91-03.

Hitchcock, A. 1992. Method of Analysis of IVHS Safety. ***PATH Research Report*** UCB-ITS-PRR-92-14, pp. 17-19.

Hurn, J. 1989. ***GPS:A Guide to the Next Utility,*** Trimble Navigation Ltd.

Jefferys, H. 1946. An Invariant Form for the Prior Probability in Estimation Problems. ***Proc. Roy. Soc. A.,*** vol. 186, pp. 453-461.

Jenkins, G. M., and Watts, D. G. 1968. ***Spectral Analysis and Its Applications,*** Holden-Day, San Francisco.

Kailath, T. 1967. The Divergence and Bhattacharyya Distance Measures in Signal Selection. ***IEEE Trans. Commun. Technol., COM-15***, pp. 52-60.

Kaufmann, A. 1971. ***Introduction to the Theory of Fuzzy Subsets,*** Academic Press, New York.

Kennedy, J. B. and Neveille, A. M. 1986. ***Basic Statistical Methods for Engineers and Scientists,*** Harper and Row, Publishers Inc.

Keresteciogiu, F. 1993. ***Change Detection and Input Design in Dynamical Systems, Control Systems.*** Center Series, Somerset, England.

Kikuchi, H., Ishiyama, M. and Toyohei, N. 1994. *Development of Laser Radar for Radar Brake System,* AVEC '94

Kim, Y-J. *1992. Uncertainty Propagation in Intelligent Sensor Validation.* Ph.D. Dissertation, Department of Mechanical Engineering, University of California at Berkeley, Berkeley.

Kim, Y-J. and Agogino, A.M. *1991.* Signal Validation for Expert System Development. *Proceedings of Expert System Applications for Electric Power Industry* (Sept. *9-11, 1991*; Boston, MA), Electric Power Research Institute.

Kim, Y.J., W.H. Wood, and Agogino, A. *1992.* Signal Validation for Expert System Development. *Proceedings of the 2nd International Forum on Expert Systems and Computer Simulations in Energy Engineering,* Erlangen, Germany), pp. *9-5-1* to *9-5-6.*

Kosko, B. *1986.* Fuzzy Entropy and Conditioning. *Information Sciences,* vol. *40,* No. *2.*

Lauritzen, S. L. *1991.* The EM Algorithm for Graphical Association Models with Missing Data. *Technical report TR-91-05,* Department of Statistics, Aalborg University.

Leonard, J. and Hugh, F. *1992.* Directed Sonar Sensing for Mobile Robot Navigation. Kluwer Academic Publishers.

Levitt, T., Agosta, J. and Binford, T. *1990.* Model-Based Influence Diagrams for Machine Vision, in Kanal., L. and Lemmer, J., editors, *Uncertainty in Artificial Intelligence* vol. *5,* pp. *371-388,* North-Holland, New York.

Lygeros, J., Godbole, D., and Broucke, M. *1995.* Design of an Extended Architecture for Degraded Modes of Operation of AHS. *Proceedings of American Control Conference,* Seattle.

McLauchlan, P. F., and Malik, J. 1997. *Vision for Longitudinal Vehicle Control,* BMVC'97.

McLauchlan, P. F., and Wang, J. *1997.* Static Vision Data Set in North of San Diego, California.

Minkoff, J. *1992. Signals, Noise, & Active Sensors: radar, sonar, laser radar,* John Wiley & Sons, Inc.

Nadi, F, Agogino, A.M. and Hodges, D. A. *1991.* Use of Influence Diagrams and Neural Networks in Modeling Semiconductor Manufacturing Processes. *IEEE Transactions on Semiconductor Manufacturing,* Vol. *4,* No. *1,* Feb. *1991,* pp. *52-58.*

Neal, R. M. *1991.* Connectionist Learning of Belief Networks. *Artificial Intelligence,* vol. *56,* pp. *71-113.*

Neapolitan, R. E. *1990. Probabilistic Reasoning in Expert Systems, Theory and Algorithms.* Wiley, New York.

Olmsted, S. M. *1984.* On Representing and Solving Decision Problems. *Doctoral Dissertation, Department of Engineering-Economic Systems, Stanford University.*

Oppenheim, A. V. and Schafer, R. W. *1989. Discrete-Time Signal Processing,* Prentice Hall.

Patton, R. J. *1994.* Robust Model-Based Fault Diagnosis: The State of the Art. *IFAC Fault Detection, Supervision, and Safety for Technical Processes,* Espoo, Finland.

Pearl, J. *1988. Probabilistic Reasoning in Intelligent Systems.* San Mateo, Ca: Morgan Kaufmann.

Pearl, J. *1995.* Bayesian Networks. Arbib, M. (editor), *Handbook of Brain Theory and Neural Networks,* MIT Press.

Ramamurthi, K and Agogino, A. *1993.* Real-Time Expert System for Fault-Tolerant Supervisory Control. *Journal of Dynamic Systems, Measurement, and Control,* vol. *115,* pp. *219-227.*

Rege, A. and Agogino, A.M. *1988.* Topological Framework for Representing and Solving Probabilistic Inference Problems in Expert Systems. *IEEE Systems, Man, and Cybernetics,* Vol. *18* (3), pp. *402-414.*

Rojas-Guzman, C. and Kramer, M. A. *1993.* Comparison of Belief Networks and Rule-based Expert Systems for Fault Diagnosis of Chemical Processes. *Engineering Application of Artificial Intelligence,* vol. *6,* no. *3,* pp. *191-202.*

Russell, S. and Norvig P. *1995. Artificial Intelligence: A Modern Approach.* Prentice Hall Series in Artificial Intelligence.

Sauter, D., Dubois, G., Levrat, E. and Bremont, J. 1993. Fault Diagnosis in Systems using Fuzzy Logic. EUFIT '93, *First European Congress on Fuzzy and Intelligent Technologies,* Aachen, Germany.

Schneider, H. and Frank, P. M. 1993. Implementation of a Fuzzy Concept for Supervision and Fault Detection of Robots. EUFIT '93, *First European Congress on Fuzzy and Intelligent Technologies,* Aachen, Germany.

Shachter, R. D. 1986. Intelligent Probabilistic Inference. *Proceedings of the AAAI Workshop on Uncertainty and Probability in Artificial Intelligence,* pp. 237-244.

Shachter, R. D. 1987. Evaluating Influence Diagrams. *Operations Research,* Vol. 34 (6), pp. 871-882.

Shachter, R. and Kenley, C. 1989. Gaussian Influence Diagrams. *Management Science,* 35, pp. 527-550.

Shachter, R. D. 1990. Editor special issue on influence diagrams. *Networks: An International Journal,* vol. 20 (5).

Shafer, G. and Pearl, J. 1990. (Eds) *Readings in Uncertain Reasoning,* Morgan Kaufmann, San Mateo, CA.

Spiegelhalter, D. J., David, P., Lauritzen, S. L. and Cowell, R. G. 1993. Bayesian Analysis in Expert Systems, *Statistical Science,* vol. 8, no. 3, pp. 219-283.

SRI International. 1997. Palo Alto Airport Data Set. California.

SRI International. 1997. Northbound on Highway 85 Around Sheller Road Data Set. California.

Turtle, H. and Croft, B. 1991. Evaluation of an Inference Network-Based Retrieval Model. *ACM Transactions on Information Systems,* vol. 9, pp. 1878-222.

Varaiya, P. 1991. Smart Vehicles on Smart Roads: Problems of Control. *PATH Technical Memorandum 91-5,* UC Berkeley.

Varaiya, P. and Shladover, S. 1991. Sketch of an IVHS Architecture. *PATH Technical Report* UCB-ITS-PRR-91-03.

Villanueva, I. 1996. Fault Tree Analysis of Sonar and Radar Sensors: Hazard Anticipation in the IVHS Environment. *SUPERB final report,* University of California.

Vesely, W. E, et al. 1981. *Fault Tree Handbook.* U.S. Nuclear Regulatory Commission. pgs. Iv-1 -v-3.

Wald, A. 1947. *Sequential Analysis.* John Wiley and Sons, New York.