WestVirginiaUniversity
THE RESEARCH REPOSITORY @ WVU

Graduate Theses, Dissertations, and Problem Reports

2004

Information fusion schemes for real time risk assessment in adaptive control systems

Martin Mladenovski
West Virginia University

Recommended Citation

Mladenovski, Martin, "Information fusion schemes for real time risk assessment in adaptive control systems" (2004). *Graduate Theses, Dissertations, and Problem Reports*. 1500.
https://researchrepository.wvu.edu/etd/1500

INFORMATION FUSION SCHEMES FOR REAL TIME RISK ASSESSMENT IN ADAPTIVE CONTROL SYSTEMS

by

Martin Mladenovski

Thesis submitted to the College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Approved by

Bojan Cukic, Ph.D., Committee Chairperson
Frances VanScoy, Ph.D.
Katerina Goseva – Popstojanova, Ph.D.

Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2004

Keywords: Information Fusion, Data Fusion, Dempster-Shafer, Fuzzy Logic, Adaptive System Verification & Validation, DCS

ABSTRACT

INFORMATION FUSION SCHEMES FOR REAL TIME RISK ASSESSMENT IN ADAPTIVE CONTROL SYSTEMS

by Martin Mladenovski

Intelligent Flight Control System (IFCS) deploys a neural network for in-flight aircraft failure accommodation. Verification and validation (V&V) of adaptive systems is a challenging research problem. Our approach to V&V relies on real-time monitoring of neural network learning. Monitors detect learning anomalies and react to different failure conditions. We investigated data fusion techniques suitable for the analysis of neural network monitors. Monitor outputs are fused into a measure of confidence, indicating the belief in the correctness of failure accommodation mechanism provided by the neural network. We investigated two data fusion techniques, one based on Dempster-Shafer theory and the other based on fuzzy logic. Our techniques were applied to nine flight simulation datasets including those with failures. The monitor fusion algorithms provide unique, meaningful and novel technique for V&V of adaptive flight control systems. Being theoretically sound, the algorithms can be applied to a broad range of other data fusion applications.

DEDICATION

Dedicated to my family that always supported me.

Milorad
Gordana
Saso

ACKNOWLEDGMENTS

Many people need to be credited for carrying out this work. I am very glad that I had the opportunity to work with my advisor and mentor Dr. Bojan Cukic and I would like to thank him for his understanding and support. Thanks to my other Graduating Committee members, Dr. Frances VanScoy and Dr. Katerina Goseva–Popstojanova, I gained valuable knowledge during my studies at West Virginia University. I would also like to thank Srikanth Gururajan for providing the data sets from the F-15 simulator and Sampath Yerramalla for helping me to perform the experiments.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

The advance in technology and science resulted in creation of more complex systems. These systems have more sensors that provide information about the system itself or about the environment where the system is implemented. Observing sensors and making decisions is a difficult task. The human brain already has the ability to fuse information from different sensors (eyes, ears ...) and make decisions based on them. In automated intelligent systems decisions have to be made by the system, resulting in a need for data fusion. Another reason for data fusion is that observation of many sensors can bring confusion. Data needs to be fused into one sensor that will be simple to read and interpret the meaning of it. A significant effort has been put into the development of data fusion methods and algorithms. A number of them based on Bayesian theory [5], some of them based on Dempster-Shafer theory, [9], [10]. Also important methods of data fusion are the methods based on fuzzy logic [13] [14].

A complex system that we will concentrate in this work is the Intelligent Flight Control System (IFCS), [1], shown in Figure 1. The system consists of sensors from an aircraft, Real-time PID (parameter identification), baseline neural network, online DCS NN (Dynamic Cell Structure Neural Network) and a controller. Data collected from the sensors goes to the PID and the baseline NN. Using both of these components we have derivative estimations for the controller and the output from the PID is sent to the DCS network which is of our interest. This network, based on what it learned is making additional corrections to the PID output. Finally these corrected derivatives are sent to the controller of the aircraft. Given that the DCS network is an online network, we need to find a way to validate its outputs and its behavior, i.e., to come up with a real time risk assessment for this adaptive control system. To observe the online DCS network, four monitors were developed, [3]. These monitors react to different failures of the aircraft. However, to conclude whether the network's output can recognize and accommodate system failures or not, we need to use data fusion methods to fuse the data from the four monitors into a single measure of how confident we are in the correct accommodation provided by the DCS network. We will call this measure a confidence measure.

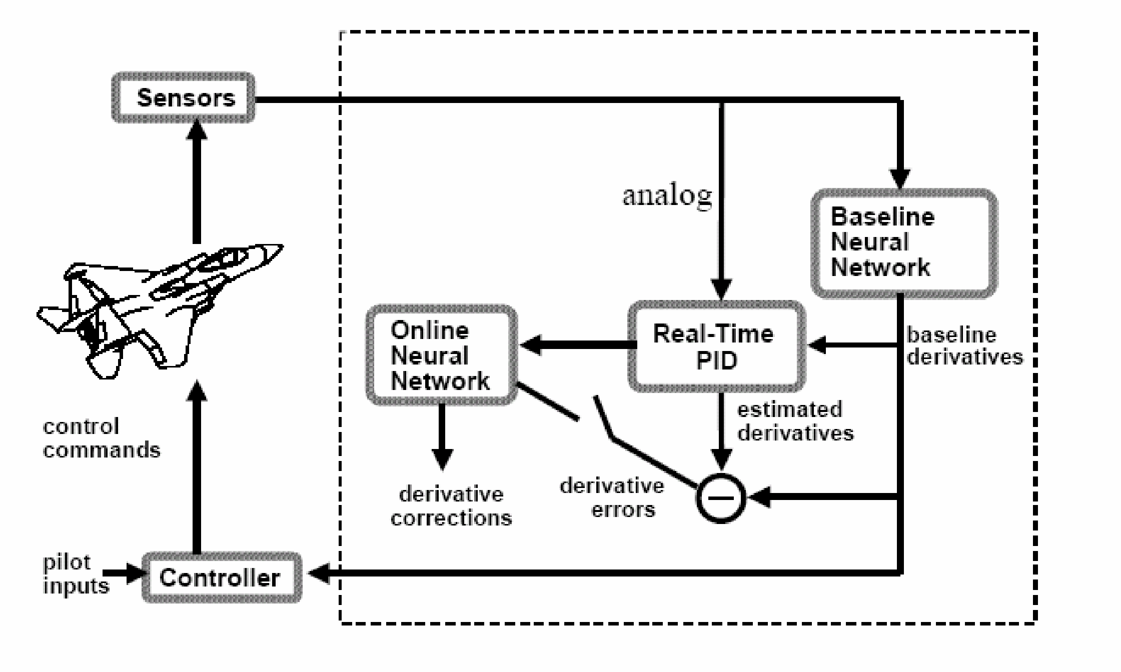

Figure 1 Intelligent Flight Control System (IFCS)

The current methods for verification and validation of neural network based adaptive systems provide weak results, as they are scientifically immature. To address this problem [15] and [16] propose a multi-layered approach which is very similar to the standard software verification and validation lifecycle and very difficult to perform. However, one of the proposed layers in this approach is dynamic monitoring. Our approach to verification and validation relies on real-time monitoring of the DCS neural network learning. As mentioned we will use data fusion techniques to integrate the information provided from the four monitors and achieve a simple yet meaningful insight into the real-time behavior of the DCS neural network.

In chapter 1, we will present the related work including description of a DCS NN with its monitors and two aspects of data fusion (Dempster-Shafer theory and Fuzzy Logic). Chapter 2 describes a method of data fusion based on the Murphy's rule of combination from the Dempster-Shafer framework. A fuzzy logic data fusion method is presented in chapter 3. Case studies that include seven failure modes of the aircraft and two no-failure modes are presented in chapter 4. Conclusion and the comparison between the two proposed methods of data fusion are done in the last chapter.

CHAPTER 1: RELATED WORK

Before we present data fusion techniques, first we have to explain the DCS network and the meaning of its monitors.

1.1. DCS Neural Network

DCS networks like any other neural networks consist of neurons and connections between the neurons. Each neuron represents a location in the output space. The goal of training this type of networks is to achieve a spatial representation of the data the network is trying to learn. This training is done using Kohonen rule and Hebbian rule, which adapt the neurons' location and the connection weights between them [2], [3], and [4]. When the DCS network is "asked" to provide the output for certain input it finds the closest neuron (best matching unit) to the input, and gives the appropriate output. The online DCS network in IFCS is trained every timeframe. It grows up to a certain number of neurons and after that the neurons are adjusted to better represent the incoming data. In addition of getting the output from the DCS network every time frame, we can also get the values from the following four monitors [3]:

1. **BMU Error** is the Euclidean distance between each data element of the presented input (training) data pattern and its closest neuron (node) of the neural network, known as the Best Matching Unit.
2. **SBU Error** is the Euclidean distance between each data element of the presented input (training) data pattern and its second closest neuron (node) of the neural network, known as the Second Best Unit.

3. **Neighborhood (NBR) error** is the mean Euclidean distance between each data element of the presented input (training) data pattern and the set of neighborhood neurons (connected nodes) of the BMU of the neural network, known as the NBR-set.
4. **Non-Neighborhood (Non-NBR) Error** is the mean Euclidean distance between each data element of the presented input (training) data pattern and the set of laterally connected, non-neighboring neurons of the BMU of the neural network, known as the Non- NBR-set.

These four monitors are providing information about how well the training data is represented by the DCS network. Certain monitors will react to some types of external data irregularities, other monitors to different types of irregularities. Therefore we need methods to fuse this information provided by the monitors into a single measure of confidence.

1.2. Dempster-Shafer theory

The first interest in the Dempster-Shafer theory was from the artificial intelligence community. Later, it found applications in software engineering, to interpret inconsistencies in software requirement specifications, and in pattern classification.

Dempster-Shafer theory [6] is a general form of Bayesian theory. Like every probabilistic model, the Bayesian model has limitations. These limitations are due to the fact that people are not good probability estimators. The probabilistic functions are replaced with belief functions in the Dempster-Shafer theory. Expressing someone’s degree of belief is more suitable than estimating a probability.

Smets, [7], gives reasons for the need of a model to represent the belief process:

> "The need of a mathematical model to simulate the belief process can be justified by cognitive, normative or pragmatic arguments:
>
> 1. *Cognitive*: It helps in obtaining a better understanding of the underlying psychological process.
> 2. *Normative*: It establishes rules of behavior that should be applied by everyone.
> 3. *Pragmatic*: It provides a mathematical construct that can be implemented on computers, as in expert systems, in order to simulate a cognitive process where the concept of belief is relevant."

Shafer's early work, [6], was on the transferable belief model and his later work was on upper and lower probability model. In our work we will focus on the transferable belief model. Belief functions in this model quantify the degrees of belief and they are an outcome of distinct sources of evidence. Dempter's rule of combination is used to combine these functions. There is no relation between the Dempster-Shafer's model and the probability model, as Smets argues in [7]:

> "We insist on the fact that:
> 1. The transferable belief model is built without ever introducing explicitly or implicitly any concept of probability.
> 2. Dempster's rule of conditioning is one of the natural ingredients of the transferable belief model. It is not ad hoc. It is a center of the model.
> 3. Dempster's rule of combination will be derived from the transferable belief model"

To better explain this model we need to give details about the frame of discernment, assigning beliefs and the Dempster's rule of combination.

1.2.1. The Frame of Discernment

Similar to other models of reasoning, the transferable belief model has a finite boolean algebra of propositions, Θ. Beliefs are built on this finite boolean algebra. Θ is know as the frame of discernment and has other names like the universe of discourse, the domain of reference. All propositions included in the frame of discernment are considered as possible propositions. Propositions not included are stated as impossible propositions. The Bayesian theory assumed only these two sets of propositions, the set of known possible propositions (PP) and the set of known impossible propositions (IP). In view of the fact that this is a closed-world assumption, some processes from reality could not be modeled well using this assumption. For that reason, the set of unknown propositions (UP) is taken into account, an open-world approach, [7]. These are the propositions that we are not sure if they will occur. As soon as we collect more evidence that support or do not support our propositions, we can rearrange them in the appropriate sets. For the simplicity and for applications in the IFCS system we will assume that the set of unknown propositions is an empty set.

Shafer [6] suggests the following notations when we are interested in the true value of some quantity:

- θ to denote the quantity
- the set of all possible values (elementary propositions) Θ (as mentioned previously, the frame of discernment)
- propositions of interest are in the form of "The true value of θ is in T", where T is a subset of Θ.

From the above, it can be concluded that the propositions of interest are in a one-to-one association with the subsets of Θ. In addition, the set of all propositions relates to the set of all subsets of Θ. As in [6] we will indicate the set of all subsets as 2^{Θ}. To demonstrate how to construct Θ and how to define θ, let us consider that we want to determine the location and the time period of the existence of Troy. All pairs of possible location and time period will form the set Θ. Our interest θ, as stated, will be "the location and the time period of the existence of Troy".

It is important to be aware that the meaning and the knowledge we place in Θ depends on our current understanding of the reality. In addition our language and its constructs influence the correctness and meaningfulness of the propositions (possibilities) we put in the frame of discernment, Θ.

In order to explain the logical concepts of negation, conjunction, disjunction and implication with graphical ideas we can do the translation to complementation, intersection, union and inclusion. Let A and B be subsets of Θ and their matching propositions A' and B', then the following translations can be made:

- $A = \overline{B}$ if and only if "A' is negation of B'"
- $A \bigcap B$ corresponds to "conjunction of A' and B'"
- $A \bigcup B$ corresponds to "disjunction of A' and B'"
- $A \subset B$ if and only if "A' implies B'"

A proposition that is known to be false is represented trough the empty set, $\varnothing$ which is an element of 2^{Θ}. Opposite to the empty set, the set Θ corresponds to a proposition known to be true.

1.2.2. Assigning beliefs

The next step after forming the frame of discernment is to assign beliefs to propositions. Some portion of the belief assigned to the proposition is assigned also to the propositions it implies. A part of the total belief assigned to a subset of Θ, A, may be assigned to one or more proper subsets of A and the rest only to A. Shafer gives the following definition:

Definition 1: If Θ is a frame of discernment, then a function $m : 2^{\Theta} \to [0,1]$ is called a *basic probability assignment* whenever:

1. $m(\emptyset) = 0$
2. $\sum_{A \subset \Theta} m(A) = 1$.

Here $m(A)$ represents the assigned belief just to A, called *basic probability number*. The assigned belief to the empty set is 0 and all assigned beliefs must sum up to 1. To acquire the total belief assigned to A we have to add all basic probability numbers of all proper subsets of A, $Bel(A) = \sum_{B \subset A} m(B)$.

Definition 2: A function $Bel(A) : 2^{\Theta} \to [0,1]$ is called a *belief function* over Θ if it is given by $Bel(A) = \sum_{B \subset A} m(B)$ for some basic probability assignment $m : 2^{\Theta} \to [0,1]$.

1.2.3. Dempster's Rule of Combination

Up to now we were focused on how to assign beliefs from one source of evidence to each proposition (subset of Θ). In a case of a several distinct sources of evidence there is a need to combine their belief functions. Dempster's rule of combination is just a computation of their orthogonal sum, which will be explained below. A very good way to explain the Dempster's rule is if we introduce the basic probability numbers geometrically.

Let m_1 be the basic probability assignment of a belief function Bel_1 over frame Θ. The focal elements (elements of interest) of Bel_1 are denoted as $A_1,...,A_p$. Similar, for the other source of evidence m_2 will be the basic probability assignment of a belief function Bel_2 and its focal elements $B_1,...,B_q$. Figure 2 and Figure 3 (both taken from [6]) show the geometrical representation on a line segment with length one of the basic probability numbers of m_1 and m_2 respectively.

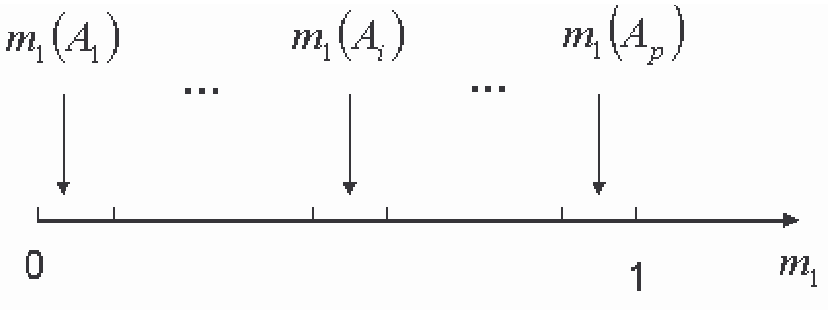

Figure 2 Representation of m_1 with line segments

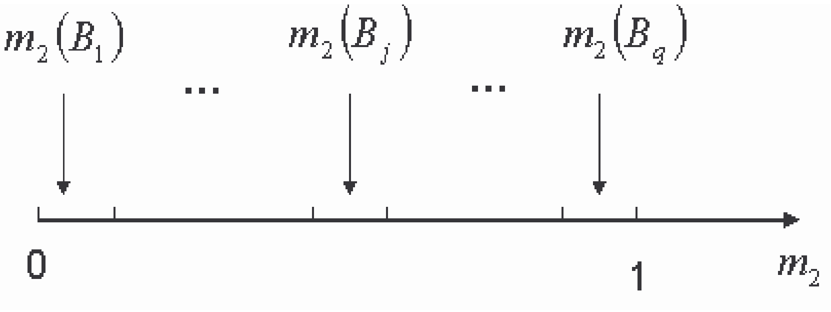

Figure 3 Representation of m_2 with line segments

The probability masses $m_1(A_i)$ and $m_2(B_j)$ are exactly assigned to A_i and B_j correspondingly. Our interest is focused on the intersection $A_i \cap B_j$. This orthogonal combination of m_1 and m_2 is represented in Figure 4 (taken from [6]).

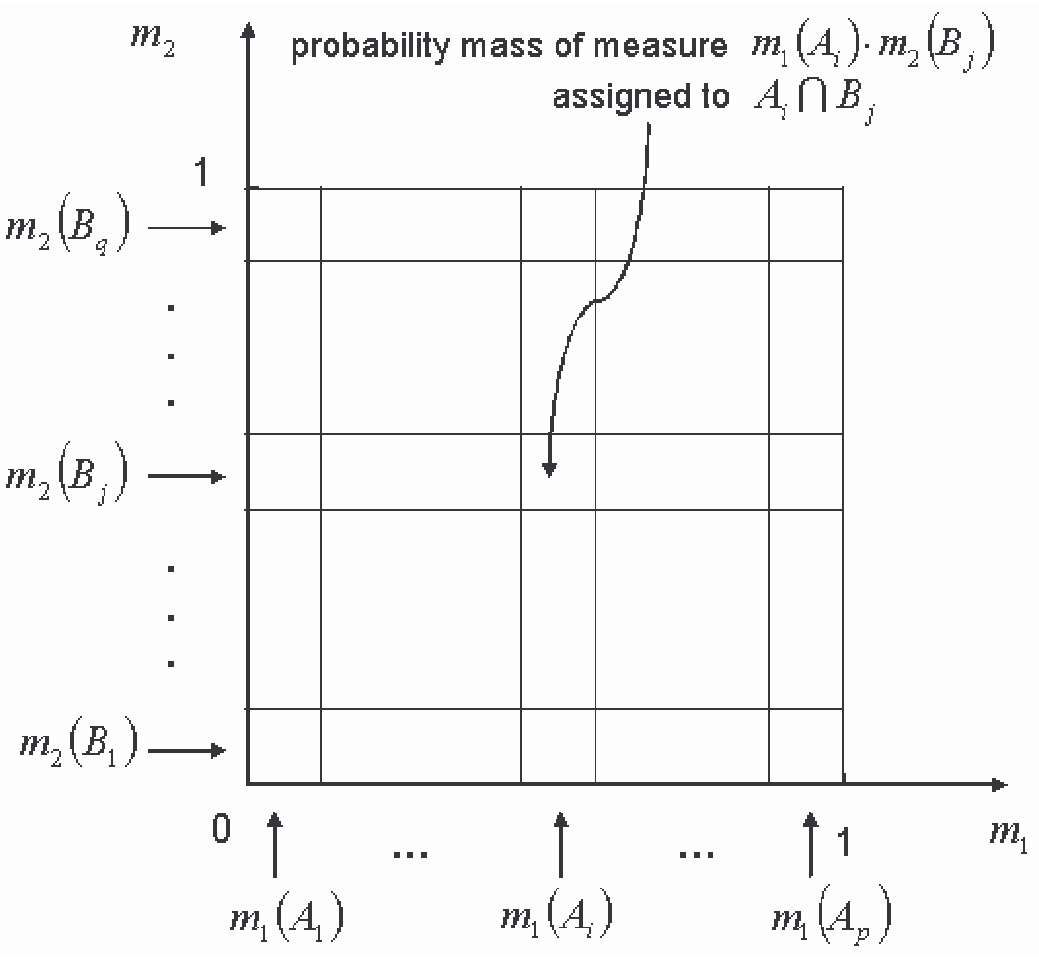
m_2
probability mass of measure $m_1(A_i) \cdot m_2(B_j)$
assigned to $A_i \cap B_j$
1
$m_2(B_q)$
$m_2(B_j)$
$m_2(B_1)$
0
...
...
1
m_1
$m_1(A_1)$
$m_1(A_i)$
$m_1(A_p)$

Figure 4 Orthogonally combined line segments of m_1 and m_2

From Figure 4 one can notice that the intersections of the bands $m_1(A_i)$ and $m_2(B_j)$ is the measure $m_1(A_i) \cdot m_2(B_j)$. This measure is exactly assigned to the intersection $A_i \cap B_j$. In a similar way we can specify the exact assigned measure for each intersection rectangle. Any subset $C \in \Theta$ can cover up some rectangles that are exactly assigned to it. Then, the total probability mass assigned to C will be:

$$\sum_{\substack{i,j \\ A_i \cap B_j = C}} m_1(A_i) \cdot m_2(B_j).$$

The problem with the sum above is that it adds up also the mass of the possible intersections $A_i \cap B_j$ that are equal to the empty set $\varnothing$. In that case the sum of these intersections will be greater than 0:

$$\sum_{\substack{i,j \\ A_i \cap B_j = \varnothing}} m_1(A_i) \cdot m_2(B_j) > 0 .$$

A solution to this problem, provided in [6], is to discard these rectangles and the remaining rectangles to be multiplied by the factor:

$$\left(1 - \sum_{\substack{i,j \\ A_i \cap B_j = \varnothing}} m_1(A_i) \cdot m_2(B_j)\right)^{-1}$$

This ensures that the total probability mass will again have measure one. The following theorem (proven in [6]) clarifies the Dempster's rule of combination:

Theorem 1: Suppose Bel_1 and Bel_2 are belief functions of the same frame Θ, with basic probability assignments m_1 and m_2 and focal elements $A_1,..., A_p$ and $B_1,..., B_q$, respectively.

Suppose:

$$\sum_{\substack{i,j \\ A_i \cap B_j = \emptyset}} m_1(A_i) \cdot m_2(B_j) < 1 .$$

Then the function $m : 2^{\Theta} \to [0,1]$ defined by $m(\emptyset) = 0$ and

$$m(C) = \frac{\sum_{\substack{i,j \\ A_i \cap B_j = C}} m_1(A_i) \cdot m_2(B_j)}{1 - \sum_{\substack{i,j \\ A_i \cap B_j = \emptyset}} m_1(A_i) \cdot m_2(B_j)}$$

for all non-empty $C \in \Theta$ is a basic probability assignment. The core of the belief function given by m is equal to the intersection of the cores of Bel_1 and Bel_2.

This belief function, called the orthogonal sum Bel_1 and Bel_2 and given by m, usually is denoted as $Bel_1 \oplus Bel_2$.

The rule of combination can be rewritten in the following way:

$$m(C) = \frac{\sum_{\substack{i,j \\ A_i \cap B_j = C \\ C \neq \emptyset}} m_1(A_i) \cdot m_2(B_j)}{\sum_{\substack{i,j \\ A_i \cap B_j \neq \emptyset}} m_1(A_i) \cdot m_2(B_j)}$$

1.2.4. Murphy’s Rule of Combination

Murphy’s research [9] and [10], concerning combination of evidence in mobile autonomous robots, showed some problems in Dempster’s rule of combination. This rule has two assumptions that were not suitable for mobile robots. The assumptions are that the sources of evidence are independent and that the order of combination is not important, [6] and [7]. To meet the requirements of the mobile robot system, Murphy came up with a different rule of combination based on the following assumptions:

- The sources of evidence are not independent
- The order of combination is important

Every multiplicative term $m_1(A_i)\cdot m_2(B_j)$ in the Dempster's rule was replaced with $f(m_1(A_i)\cdot m_2(B_j))$, where f is an arbitrary nonnegative function of two variables: $f : f(m_1(A_i)\cdot m_2(B_j))\in R^+$. Experimental research showed that the function $f(m_1(A_i)\cdot m_2(B_j)) = [m_1(A_i)\cdot m_2(B_j)]^n, n \in [0,1]$ gives good results. The choice of parameter n makes this rule adaptive, [9]. When new evidence has more weight, then n should be greater than 0.5 (*optimistic*). Opposite of that, when the new evidence has less weight the choice of n should be less than 0.5 (*pessimistic*). Values of n around 0.5 are called *neutral* values, and mean that the revision of the belief is nor pessimistic nor optimistic. Another clarification on the choice of n is through correlation. Low correlation values between the sources of evidence, result in higher values of n closer to 1. For $n = 1$ Murphy's rule becomes Dempster's rule of combination meaning that the sources of evidence are completely independent. High correlation values between the sources of evidence lead to lower values of n closer to 0.

1.3. Fuzzy Inference System

The fuzzy logic is widely used today in industry and science. Lofti Zadeh in 1964 had the idea regarding fuzzy sets and later in 1965 he published an influential paper on fuzzy sets which was the birth of fuzzy logic technology.

Following this, there was a significant scientific progress followed by industrial progress. One of the creators of the first fuzzy logic controller was E. Mamdani, who worked with Assilian in 1974 to develop a controller for a steam generator. A big “fuzzy boom” was in Japan starting from 1987.

There were two main goals that motivated fuzzy logic researchers. One goal was to use fuzzy logic on analysis and development of complex systems, thus overcoming problems of conventional mathematical tools. The other goal was to make use of human reasoning, concepts and knowledge that are not well defined. These two goals defined two requirements from fuzzy logic, [11]:

- A requirement that allows fuzzy logic to work in quantitative and numeric domains
- A requirement that enables fuzzy logic to have descriptive and qualitative form, because imprecise concepts are usually described qualitatively.

The most extensively used fuzzy logic techniques based on fuzzy sets are the fuzzy “if-then” rules. Their application is in many fields like control systems, decision making, pattern recognition and other. Industrial applications are in robotics, process control, financial trading, medical imaging and many more. A fuzzy if-then rule is defined in [11]:

Definition 3: A fuzzy if-then rule associates a condition described using linguistic variables and fuzzy sets to a conclusion.

There are two types of fuzzy rules:

1. *fuzzy mapping rules*: describe a functional mapping relationship between inputs and output using linguistic terms

2. *fuzzy implication rules*: describe a generalized logic implication relationship logic between two logic formulas involving linguistic variables and imprecise linguistic terms

In this work we will be focused on the *fuzzy rule-based models for function approximation*, which are based on fuzzy mapping rules. Before we continue with specific fuzzy models we will quote some definitions from [11]:

Definition 4: A fuzzy subspace is a region whose boundary allows a gradual transition from “inside the region” to “outside the region”.

Definition 5: A fuzzy model is a model obtained by fusing multiple local models that are associated with fuzzy subspaces of the given input space.

For function approximation there are three models combined in two groups, additive rule models and non-additive rule models. Additive rule models are the Takago-Sugeno-Kang (TSK) model and Kosko’s additive model (SAM). The Mamdani model belongs to the non-additive rule models.

For all these models there are few steps that are taken:

1. fuzzy partition
2. mapping of fuzzy sub regions to local models
3. fusion of multiple local models
4. defuzzification

The input space is partitioned into disjoint subspaces. Membership functions are used to achieve the partitioning. Classical partitioning does not have smooth transitions between subspaces. This is the main difference from fuzzy partitioning where the membership degree from one subspace smoothly increases while the membership degrees from other subspaces smoothly decrease. Using this approach transitions from neighboring subspaces are smooth. In Figure 5 there is an example of how an input variable temperature with triangular membership functions is partitioned into three subspaces: low, medium and high temperature.

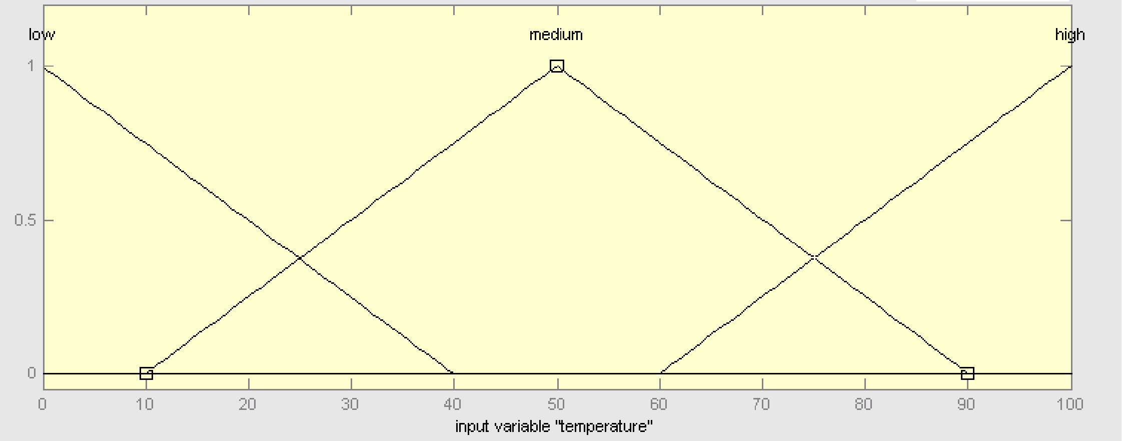

Figure 5: Membership function example

The next step is to map the fuzzy subspaces into a local model. Fuzzy if-then rules are used to accomplish this step. The rules are in the form of:

> “***IF*** *the input variables belong to some subspaces* ***THEN*** *the output is a local model from those variables.* “

Local models can be a crisp constant (a number), a fuzzy constant (e.g. NORMAL), a linear model or a nonlinear model.

Fusion of local models is done through interpolative reasoning. Depending on the quantity of belonging to a subspace for the situation of the rule, weight for each rule is decided.

The last step is the defuzzification method, and the most used methods are: mean of maximum (MOM), center of area (centroid) or the height method. This step determines the final output value. The defuzzification method, mean of maximum, from the output values that have the highest possibility degrees calculates the average. Centroid, unlike MOM, takes into consideration the entire possibility distribution.

Fuzzy logic systems have been used as a data fusion technique, [13], [14], in different areas. Their application showed up good results. For that reason we consider this technique as a data fusion technique for the IFCS system.

CHAPTER 2: DATA FUSION BASED ON DEMPSTER-SHAFER ‘S THEORY

This chapter presents a data fusion method based on the Dempster-Shafer theory. Murphy’s rule is used as a combination rule; since the four monitors, i.e. the sources of evidence are all dependent. The dependency comes from the fact that the monitors are based on the same DCS network.

When the input data is processed by the DCS network, the four monitors will provide values. These values are immediately processed by the method explained in this chapter, which results in a real-time confidence measure. Prior to sending the data from the monitors to data fusion method, the data is normalized in an online manner to be between 0 and 1. For each monitor we keep the maximum value that appeared to the current time frame. Thus, every time step the values from these monitors are divided by the current maximum from the corresponding monitor. We justify this way of normalizing the data because when the DCS network starts learning the values of the monitors (the errors of the represented data) are usually very high. Over time these values decrease and if some fault happens in the network or the incoming data to the network changes rapidly the normalized values from some monitors will show a spike close to 1.

In order to apply the Murphy’s rule on these four different sources of evidence we need to create the frame of discernment, Θ. Given that all these monitors suggest some belief about how distant is the represented data from the actual data, we can form two propositions:

- E - how distant is the represented data from the actual data, meaning how much we are not certain in the network’s outputs
- C - how confident we are in the network’s outputs and how much we can trust them. This is our confidence measure.

Since these propositions represent opposite beliefs we can write that $C = \overline{E}$ and $C \bigcap E = \varnothing$. The normalized value from each monitor will be assigned as a belief (basic probability number) to the proposition E and represented with $m(E)$. Consequently, the assigned belief to proposition C will be $m(C)=1-m(E)$ so that the total assigned belief to all propositions is 1. To exemplify how we combine the evidence from two monitors, first let m_1 and m_2 be their basic probability assignment functions. Then, from the following table, we can see their orthogonal combination:

$m_1 \backslash m_2$	E	C
E	$m_1(E)\cdot m_2(E)$	$\varnothing$
C	$\varnothing$	$m_1(C)\cdot m_2(C)$

Table 1 Orthogonal combination of beliefs from two monitors

Based on the table above we can now write the combined belief function m_{12}:

$$m_{12}(E)=\frac{(m_1(E)\cdot m_2(E))^n}{(m_1(E)\cdot m_2(E))^n+(m_1(C)\cdot m_2(C))^n}$$ for the proposition E and

$$m_{12}(C)=\frac{(m_1(C)\cdot m_2(C))^n}{(m_1(E)\cdot m_2(E))^n+(m_1(C)\cdot m_2(C))^n}$$ for the proposition C.

As we can see for the combined belief function m_{12}, the combined belief for proposition C is $m_{12}(C)=1-m_{12}(E)$. Thus, it is not important whether we combine only the beliefs for proposition E with $m_{12}(E)$ and at the end calculate the belief for proposition C using $m_{12}(C)=1-m_{12}(E)$; or directly calculate $m_{12}(C)$. Given that the normalized values of the monitors are the basic probability numbers for proposition E we decided first to combine the beliefs of the proposition E and then calculate the belief for proposition C.

Let us label the assigned beliefs from every monitor for proposition E in the following way:

$$a_1 = m_1(E), a_2 = m_2(E), ..., a_k = m_k(E),$$

where k is the number of monitors and $m_1, m_2, ..., m_k$ are the basic probability assignment functions for monitor #1, monitor #2, ..., monitor #k correspondingly.

There are many orderings to combine k beliefs. For example we can combine them in groups of two and then calculate the combined belief from each group. Another approach is to combine the beliefs from the first two monitors. Then, that result is combined with the belief from the third monitor and so on until we reach the last monitor. This way of ordering is illustrated in Figure 6:

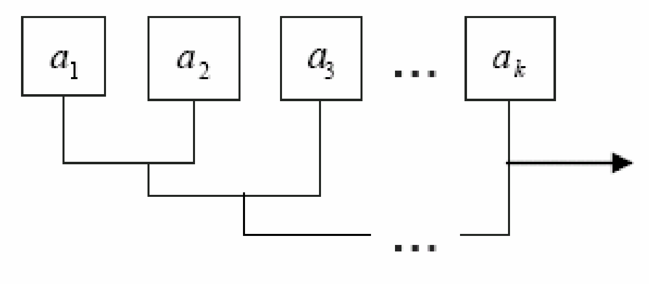

Figure 6 Way of combining beliefs from monitors

When combining like this the last monitor combined always has the highest influence on the final result. As stated in [9] and shown in section 2.1, different order of monitors results in different combined beliefs. Our concern is finding the order that gives the minimum value of the combined belief as well as the order that gives the maximum value. These two values and their absolute difference carry useful information about the learning process of the DCS network. In order to get the minimum and the maximum the first approach is to generate all possible orderings and calculate the combined belief, thus having every monitor to have highest influence on the combined belief. The number of the possible orderings of the monitors is $k!$. Generating all these orderings is time consuming as k

increases. For that reason, we claim that if we sort the beliefs of the monitors in increasing order the combined belief will have the maximum value. When sorting them in decreasing order, the combined belief will have the minimum value. Now, the total time to compute the minimum and the maximum is $O(k \cdot \log(k))$. The proof that we can compute the minimum and the maximum value of the total combined belief in $O(k \cdot \log(k))$ time is shown below in this chapter.

One must notice that if some of the beliefs have a value of 0 then the combination with another belief will always be 0. Opposite to that, if some belief has a value 1, when combined with another nonzero belief the combination will be 1. To avoid this, for beliefs equal to 0 we assign a number greater than but close to 0 and for beliefs equal to 1 we assign a number smaller than but close to 1. So, all beliefs $a_1 = m_1(E), a_2 = m_2(E), ..., a_k = m_k(E)$ will have values in the interval $(0,1)$. Another important detail is that the parameter n is constant during the process of getting the minimum and the maximum. Our choice of n is the neutral value (0.5), since we have no good understanding of the dependence between the monitors.

This method showed good results (see case studies) when put into the IFCS system. Some case studies had a failure of the aircraft. This failure was induced at a certain time. The failure data that comes to the DCS network is different than the data it has already learned. Therefore, the structure of the DCS network changes and the monitors show change. The fused data based on the method described here is represented by two values: minimum and maximum. During the failure, these two values decrease and also their difference increases. For this reason the DCS network outputs can not be trusted while these values are low and their difference is large. During no-failure modes the minimum and the maximum confidence have higher values closer to 1 and their difference is very low, closer to 0.

Previously, we mentioned the claim: *If the beliefs are sorted in increasing order, combining them as in Figure 6 will obtain the maximum value of the final combined belief; and if sorted in decreasing order then the minimum value will be* *obtained*. To prove this claim, we will use mathematical induction. Before we begin with the proof, for easier mathematical analysis we can rewrite Murphy's rule of combination of two beliefs x and y in the following way:

$$m(x,y) = \frac{(x \cdot y)^n}{(x \cdot y)^n + (1-x)^n (1-y)^n}, x, y \in (0,1);\ n \in [0,1]; n = const$$
$$m(x,y) \in (0,1)$$

From the partial derivations on x and on y one can conclude that $m(x,y)$ is an increasing function:

$$\frac{\partial m}{\partial x} = \frac{n \cdot (1-x)^{n-1} \cdot (1-y)^n \cdot (x \cdot y)^n}{x\left((1-x)^n \cdot (1-y)^n + (x \cdot y)^n\right)^2} > 0$$

$$\frac{\partial m}{\partial y} = \frac{n \cdot (1-x)^n \cdot (1-y)^{n-1} \cdot (x \cdot y)^n}{y\left((1-x)^n \cdot (1-y)^n + (x \cdot y)^n\right)^2} > 0$$

This fact will later be used in the proof. The way that we combine the beliefs $a_1, a_2, ..., a_k$, shown in Figure 6, formally can be expressed as:

$$comb(1) = a_1$$
$$comb(i) = m(comb(i-1), a_i), \quad i = 2,...,k \, .$$

Final combined belief is the value of $comb(k)$.

It is easy for one to notice that $m(x,y) = m(y,x)$ since multiplication is a commutative operation. However, if there are more than two beliefs, different order of combining them gives different results.

Now we are ready to start the proof of our claim using mathematical induction. First we will prove the base case for three beliefs, and after that by induction we will prove that it holds for any number of beliefs.

2.1. Base case

Let there be three beliefs:

$$a_1 \le a_2 \le a_3$$

Let $E_{123}, E_{213}, E_{132}, E_{312}, E_{231}$ and E_{321} be the values of $comb(3)$ on all possible permutations (3! =6). The total number of different combined values reduces to 3 since $m(x, y) = m(y, x)$:

$$\begin{aligned}
E_{123} &= m\left(m(a_1, a_2), a_3\right) = m\left(m(a_2, a_1), a_3\right) = E_{213} \\
E_{132} &= m\left(m(a_1, a_3), a_2\right) = m\left(m(a_3, a_1), a_2\right) = E_{312} \\
E_{231} &= m\left(m(a_2, a_3), a_1\right) = m\left(m(a_3, a_2), a_1\right) = E_{321}
\end{aligned}$$

Before we continue to compare these values we need to express them in a more appropriate format. The next set of mathematical expressions leads us to the desired format.

$$E_{12} = m(a_1, a_2) = \frac{(a_1 \cdot a_2)^n}{(a_1 \cdot a_2)^n + (1 - a_1)^n (1 - a_2)^n}$$

$$E_{123} = \frac{(E_{12} \cdot a_3)^n}{(E_{12} \cdot a_3)^n + (1 - E_{12})^n (1 - a_3)^n}$$

After some simplifications, E_{123} gets the following form:

$$E_{123} = \frac{a_1{}^{n^2} a_2{}^{n^2} a_3{}^{n}}{a_1{}^{n^2} a_2{}^{n^2} a_3{}^{n} + (1-a_1)^{n^2} (1-a_2)^{n^2} (1-a_3)^{n}}$$

In a similar way we get the values for E_{132} and E_{321}:

$$E_{132} = \frac{a_1{}^{n^2} a_2{}^{n} a_3{}^{n^2}}{a_1{}^{n^2} a_2{}^{n} a_3{}^{n^2} + (1-a_1)^{n^2} (1-a_2)^{n} (1-a_3)^{n^2}}$$

$$E_{321} = \frac{a_1{}^{n} a_2{}^{n^2} a_3{}^{n^2}}{a_1{}^{n} a_2{}^{n^2} a_3{}^{n^2} + (1-a_1)^{n} (1-a_2)^{n^2} (1-a_3)^{n^2}}$$

For easier comparison, we can bring the formulas of E_{123}, E_{132} and E_{321} to a different form:

$$\frac{1}{E_{123}} = 1 + \left(\frac{1}{a_1} - 1\right)^{n^2} \left(\frac{1}{a_2} - 1\right)^{n^2} \left(\frac{1}{a_3} - 1\right)^{n}, P_{123} = \left(\frac{1}{a_1} - 1\right)^{n^2} \left(\frac{1}{a_2} - 1\right)^{n^2} \left(\frac{1}{a_3} - 1\right)^{n}$$

$$\frac{1}{E_{123}} = 1 + P_{123}$$

$$\frac{1}{E_{132}} = 1 + \left(\frac{1}{a_1} - 1\right)^{n^2} \left(\frac{1}{a_2} - 1\right)^{n} \left(\frac{1}{a_3} - 1\right)^{n^2}, P_{132} = \left(\frac{1}{a_1} - 1\right)^{n^2} \left(\frac{1}{a_2} - 1\right)^{n} \left(\frac{1}{a_3} - 1\right)^{n^2}$$

$$\frac{1}{E_{132}} = 1 + P_{132}$$

$$\frac{1}{E_{321}} = 1 + \left(\frac{1}{a_1} - 1\right)^n \left(\frac{1}{a_2} - 1\right)^{n^2} \left(\frac{1}{a_3} - 1\right)^{n^2}, P_{321} = \left(\frac{1}{a_1} - 1\right)^n \left(\frac{1}{a_2} - 1\right)^{n^2} \left(\frac{1}{a_3} - 1\right)^{n^2}$$

$$\frac{1}{E_{321}} = 1 + P_{321}$$

To prove that $E_{123} \geq E_{132} \geq E_{321}$, it is enough to prove that $P_{123} \leq P_{132} \leq P_{321}$.

First we will prove that $P_{123} \leq P_{132}$. Suppose that:

$$\frac{P_{123}}{P_{132}} \leq 1$$

holds. Then we have:

$$\frac{\left(\frac{1}{a_2} - 1\right)^{n(n-1)}}{\left(\frac{1}{a_3} - 1\right)^{n(n-1)}} \leq 1, \quad \frac{\left(\frac{1}{a_3} - 1\right)^{1-n}}{\left(\frac{1}{a_2} - 1\right)^{1-n}} \leq 1, \quad (1-n) \geq 0, \quad \frac{\left(\frac{1}{a_3} - 1\right)}{\left(\frac{1}{a_2} - 1\right)} \leq 1$$

Since $a_2 \leq a_3$, it follows that $P_{123} \leq P_{132}$ and that $E_{123} \geq E_{132}$.

The next part is to prove that $P_{132} \leq P_{321}$. As in the previous case suppose that the inequality:

$$\frac{P_{132}}{P_{321}} \leq 1$$

holds. Then the inequality can be simplified as follows:

$$\frac{\left(\frac{1}{a_1}-1\right)^{n(n-1)}}{\left(\frac{1}{a_2}-1\right)^{n(n-1)}} \le 1, \quad \frac{\left(\frac{1}{a_2}-1\right)^{1-n}}{\left(\frac{1}{a_1}-1\right)^{1-n}} \le 1, \quad (1-n) \ge 0, \quad \frac{\left(\frac{1}{a_2}-1\right)}{\left(\frac{1}{a_1}-1\right)} \le 1$$

Due to the assumption that $a_1 \le a_2$, it follows that $P_{132} \le P_{321}$ and that $E_{132} \ge E_{321}$.

This completes the proof for three beliefs:

$$E_{123} \ge E_{132} \ge E_{321},$$

where the maximum is equal to E_{123} (when the beliefs are sorted in increasing order) and the minimum is equal to E_{321} (when the beliefs are sorted in decreasing order).

For the induction step we will use one property that holds when combining beliefs in this way.

Lemma 1: Let a,b and c be real numbers all of them $a,b,c \in (0,1)$ and we only know that $b \le c$ and $n \in [0,1]$ is a constant. Then let the function F be defined as $F(x,y,z) = m(m(x,y),z)$. In such a case the inequality $F(a,b,c) \ge F(a,c,b)$ holds.

Proof of Lemma 1:

Let $E_{abc} = F(a,b,c)$ and $E_{acb} = F(a,c,b)$. The final forms will be:

$$E_{abc} = F(a,b,c) = \frac{a^{n^2} b^{n^2} c^n}{a^{n^2} b^{n^2} c^n + (1-a)^{n^2} (1-b)^{n^2} (1-c)^n}$$

$$E_{acb} = F(a,c,b) = \frac{a^{n^2} b^n c^{n^2}}{a^{n^2} b^n c^{n^2} + (1-a)^{n^2} (1-b)^n (1-c)^{n^2}}$$

After some simplifications, like in the base proof, these formulas can be simplified like:

$$\frac{1}{E_{abc}} = 1 + \left(\frac{1}{a}-1\right)^{n^2} \left(\frac{1}{b}-1\right)^{n^2} \left(\frac{1}{c}-1\right)^{n}, P_{abc} = \left(\frac{1}{a}-1\right)^{n^2} \left(\frac{1}{b}-1\right)^{n^2} \left(\frac{1}{c}-1\right)^{n}$$
$$\frac{1}{E_{abc}} = 1 + P_{abc}$$

$$\frac{1}{E_{acb}} = 1 + \left(\frac{1}{a}-1\right)^{n^2} \left(\frac{1}{b}-1\right)^{n} \left(\frac{1}{c}-1\right)^{n^2}, P_{acb} = \left(\frac{1}{a}-1\right)^{n^2} \left(\frac{1}{b}-1\right)^{n} \left(\frac{1}{c}-1\right)^{n^2}$$
$$\frac{1}{E_{acb}} = 1 + P_{acb}$$

To prove that $E_{abc} \geq E_{acb}$ it is sufficient to prove that $P_{abc} \leq P_{acb}$. Suppose that the inequality:

$$\frac{P_{abc}}{P_{acb}} \leq 1$$

holds. In a similar way like previously we simplify the inequality:

$$\frac{\left(\frac{1}{b}-1\right)^{n(n-1)}}{\left(\frac{1}{c}-1\right)^{n(n-1)}} \le 1\,,\quad \frac{\left(\frac{1}{c}-1\right)^{1-n}}{\left(\frac{1}{b}-1\right)^{1-n}} \le 1,\quad (1-n) \ge 0\,,\ \frac{\left(\frac{1}{c}-1\right)}{\left(\frac{1}{b}-1\right)} \le 1$$

Since $b \le c$, it follows that $P_{acb} \le P_{acb}$ and that $E_{abc} \ge E_{acb}$, $F(a,b,c) \ge F(a,c,b)$. From this we can conclude that regardless of the value of the relation $F(a,b,c) \ge F(a,c,b)$ holds.

2.2. Induction Hypothesis

Now, suppose that for k beliefs $comb(k)$ will give us the maximum if the beliefs are sorted in increasing order and will give us the minimum if the beliefs are sorted in decreasing order. We will show the proof only for the maximum, since the proof for the minimum is analogous.

2.3. Induction proof for $k+1$

Let us prove that for sorted (in increasing order) $k+1$ beliefs ($a_1 \le a_2 \le ... \le a_{k+1}$), $comb(k+1)$ will give us the maximum. Below, in the Table 2, there are $k+1$ possible orderings. The possible orderings are generated when every belief is put as the last one, and the previous k are sorted in increasing order, since we assume that for k beliefs we can get the maximum when they are in increasing order.

order #(i)	sorted $k-1$ beliefs ($comb_i(k-1)$)	k^{th} belief	$(k+1)^{th}$ belief
0	$a_1, a_2, ..., a_{k-1}$	a_k	a_{k+1}
1	$a_2, a_3, ..., a_k$	a_{k+1}	a_1
...	...	...	...
i	$a_1, ..., a_{i-1}, a_{i+1}, ..., a_k$	a_{k+1}	a_i
...	...	...	...
k	$a_1, a_2, ..., a_{k-1}$	a_{k+1}	a_k

Table 2: Possible orderings for $k+1$ beliefs

For each row $\mathrm{i} = 1,...,\mathrm{k}$ in the table we have:

$comb_i(k+1) = m(comb_i(k), a_i)$ and $comb_i(k) = m(comb_i(k-1), a_{k+1})$.

Also, $comb_i(k+1)$ can be expressed using the F function:

$comb_i(k+1) = F(comb_i(k-1), a_{k+1}, a_i)$.

For the order labeled #0, we have:

$comb_0(k+1) = F(comb_0(k-1), a_k, a_{k+1})$ and $comb_0(k) = m(comb_0(k-1), a_k)$.

By Lemma 1,

$comb_i(k+1) = F(comb_i(k-1), a_{k+1}, a_i) \leq F(comb_i(k-1), a_i, a_{k+1})$,

since $a_i \leq a_{k+1}$ for all rows $\mathrm{i} = 1,...,\mathrm{k}$.

Using the assumption for k beliefs,

$m(comb_i(k-1), a_i) \leq comb_0(k) = m(comb_0(k-1), a_k)$.

Due to the fact that $m(x, y)$ is increasing function for x and y, the following holds:

$$F\left(comb_i(k-1), a_i, a_{k+1}\right) = m\left(m\left(comb_i(k-1), a_i\right), a_{k+1}\right) \le m\left(m\left(comb_0(k-1), a_k\right), a_{k+1}\right)$$

.

$$F\left(comb_i(k-1), a_i, a_{k+1}\right) \le m\left(m\left(comb_0(k-1), a_k\right), a_{k+1}\right) = comb_0(k+1)$$

We showed that $comb_i(k+1) \le comb_0(k+1)$ for each $\mathrm{i} = 1,\ldots,\mathrm{k}$, which completes the proof.

CHAPTER 3: DATA FUSION BASED ON MAMDANI FUZZY RULE BASED MODEL

Fuzzy rule based models are similar to human reasoning, for the reason that they use concepts and knowledge that are not well defined. Imprecise concepts are typically described qualitatively. The qualitative description is achieved through membership functions that partition the input and the output space. These aspects of fuzzy rule based models can be used for information fusion method that will fuse the information provided by the four monitors. The output of this method will be a single value (confidence measure), unlike the two values (minimum and maximum) provided by the method in chapter 2.

The non-additive rule-based model, called Mamdani model, has some advantages that are in our interest than the additive rule-based models TSK and SAM. Mamdani's model is in fact a very simple model and that is the reason of its broad applications. Inputs in this model can be crisp or fuzzy inputs, while the output is expressed through linguistic variables, which make the data fusion very close to human reasoning. Combination of all conclusions is done with the max operator.

Crisp inputs are the requirement for the SAM model, which is one limitation compared to the Mamdani model. This model for defuzzification uses only the centroid method, while Mamdani can have any deffuzification method. The conclusions are combined with addition operator.

Same as the SAM model, the TSK model requires crisp inputs. It is mostly used to model complex systems for accurate approximations. One reason for that is that the outputs in every rule are expressed as a linear combination of the inputs. The problem with this linear combination is determining the coefficients of every linear combination. Usually, this is done through some learning processes, like genetic algorithms, [13]. Due to the fact that our IFCS requires data fusion to be done in real-time manner, adaptation of these coefficients over time might be a slow process.

For all these reasons we select the Mamdani model as a very simple model that can be implemented in the IFCS and will make data fusion a fast process.

As explained in section 1.1, there are four monitors of the DCS network. The fuzzy model that is built is fusing data from these monitors into a single confidence measure.

When building this fuzzy system the first phase is to partition the input space and select the appropriate membership functions. Subsequent to that is creating the fuzzy rules followed by selecting the deffuzification method.

3.1. Partitioning of the Input and Output Space and Creation of Fuzzy Rules

The inputs to this system are the monitors from the DCS network. The ranges of these monitors vary over time, so we cannot determine what will be the maximum error that they will report. Therefore the online normalization method, explained in chapter 2, is used for making all these inputs to be in the range [0..1]. This normalization will make the partitioning of the input space much easier.

Another important issue is into how many partitions the input space should be divided. The number of partitions has big impact of the number of rules. Let:

- k denote the number of monitors,
- n denote the number of input partitions,
- $m_1, m_2, ..., m_k$ denote monitor values,
- $P_1, P_2, ..., P_n$ denote input partitions
- $V_1^i, V_2^i, ..., V_k^i$ denote the i th variation of the input partitions with k elements and

- $O_1, O_2, ..., O_s$ denote the output partitions, where s is the number of partitions of the output.

Every rule from the Mamdani model can be expressed in the following way:

Rule #i: “IF m_1 in V_1^i AND m_2 in V_2^i AND ... m_k in V_k^i THEN output is one partition from $O_1, O_2, ..., O_s$.

For every variation of the input partitions we have to create a rule to be sure that all possible cases are included in the fuzzy system. This leads to a conclusion that the total number of rules is the number of variations of the input partitions with k elements, which is n^k. In our DCS network $k = 4$, so partitioning the input space into too many subspaces will result in many rules that will have to be processed every time when the data from the monitors need to be fused. For this system we have chosen the number of partitions to be $n = 3$. The total number of rules is 81, which is not a large number and the input space is partitioned into reasonable subspaces. If we select $n = 4$ or $n = 5$ the number of rules grows very fast to 256 and 625, respectively. Input space is partitioned into the following subspaces:

- *good* – small errors, close to 0,
- *normal* – errors around 0.5 and
- *bad* – errors close to 1.

The above subspaces make construction of the rules much easier and more similar to human reasoning. Besides partitioning the input space, also the output space has to be partitioned. The range of the output, confidence measure, is between 0 and 1. A similar approach as for the input space leads to subspaces of the output space:

- *low* – confidence measure close to 0,
- *medium* – confidence measure around 0.5 and
- *high* – confidence measure close to 1.

However, this partitioning is not sufficient to describe the outputs of all 81 rules, since many rules will have same output partitioning. Therefore we propose partitioning of the confidence measure output into the following subspaces: **very low**, **low**, **medium**, **high** and **very high**.

3.2. Choosing Membership Functions

The next phase is choosing membership functions for the inputs and the output. Triangular functions are very simple to start with. Figure 7 and Figure 8 show the triangular membership functions for the input space and the output space.

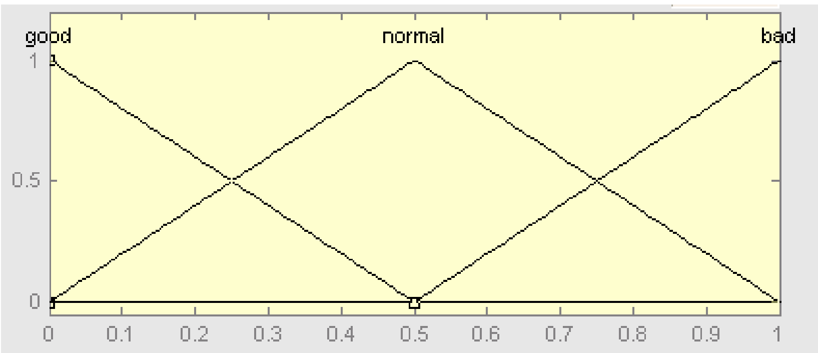

Figure 7 Triangular membership functions for the input subspaces

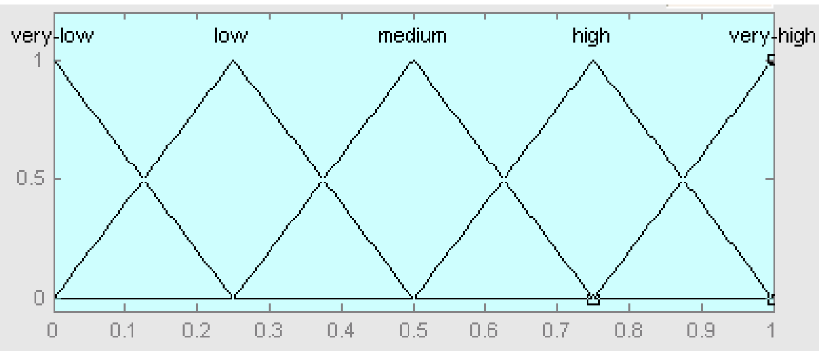

Figure 8 Triangular membership functions for the output subspaces

As one can notice, triangular functions have a peek which has the highest value, 1, only in one point and they have sharp edges at transitions. For these reasons the final transitions and changes in the output are not smooth (Figure 11).

To improve this system the following smooth membership functions were selected:

- "Z-shaped curve" – asymmetrical polynomial curve open to the left
- "S-shaped curve" – asymmetrical polynomial curve open to the right
- "Gaussian – curve" – simple Gaussian distribution curve

The input space partitions were assigned the following membership functions: ***good*** – "Z-shaped curve" membership function, ***normal*** – "Gaussian curve" membership function and ***bad*** – "S-shaped curve" membership function (see Figure 9). In a similar way membership functions for the output space were assigned: ***very-low*** with "Z-shaped curve" membership function; ***low***, ***medium***, and ***high*** with "Gaussian curve" membership function; and ***very-high*** (with "S-shaped curve" membership function (see Figure 10).

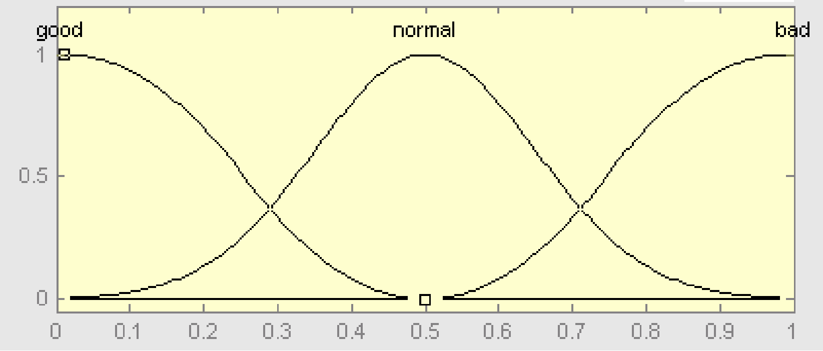

Figure 9 Smooth membership functions for the input subspaces

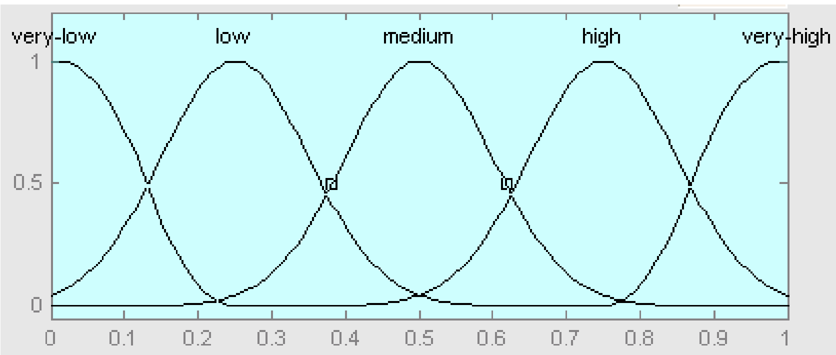

Figure 10 Smooth membership functions for the output subspaces

Results that were achieved with these membership functions were better than those with triangular membership functions (see Figure 11). It is noticeable that the fused data with triangular membership functions is not smooth and also has consistently smaller values than the one with smooth membership functions. This happens even when the values from the monitors are closer to 0, when the confidence measure should be closer to 1. The same advantages of smooth membership functions over triangular membership functions appeared in all case studies.

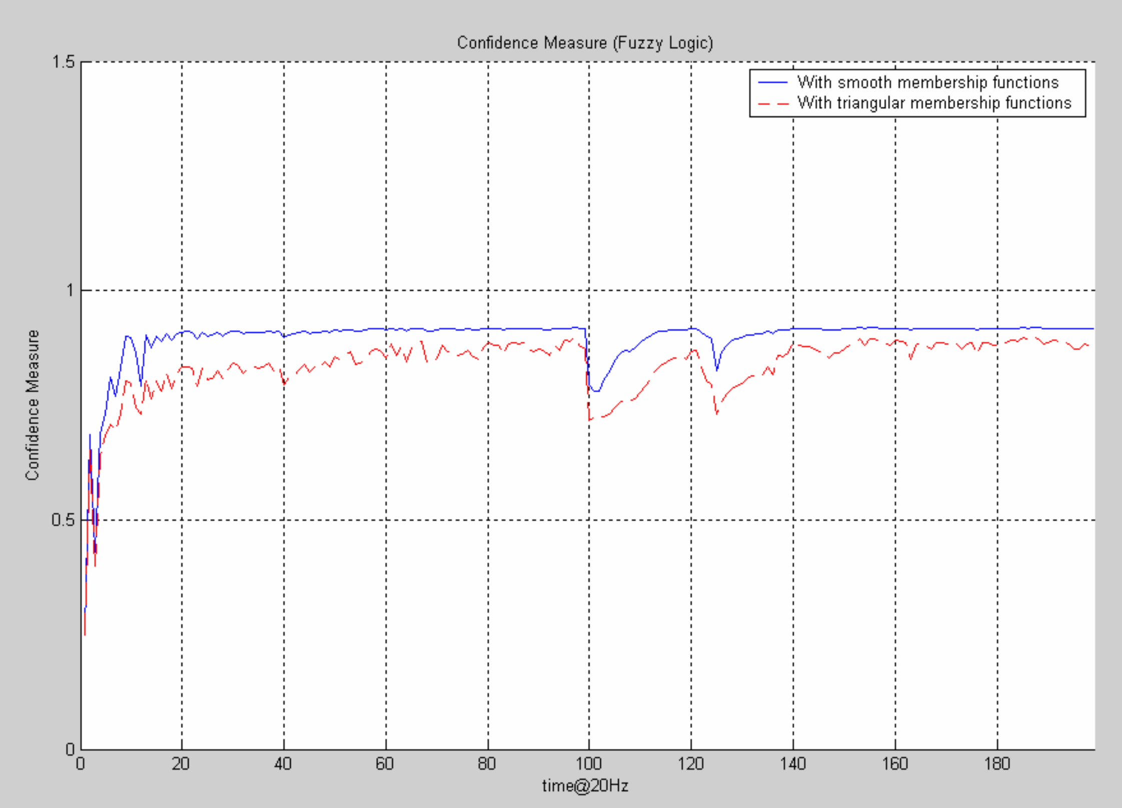

Figure 11 Comparison between triangular and smooth membership functions

3.3. Selecting Appropriate Deffuzification Method

Following the choice of membership functions and creation of the rules, the next phase is to select the appropriate defuzzification method. The most popular deffuzification methods are mean of maximum (MOM) and center of area (Centroid). Like explained in section 1.3, MOM takes into account only those output values with higher possibility degrees. Sometimes this leads into very low sensitivity on failures in the DCS network when compared with data fusion method based on Murphy's rule. In Figure 12 we can see that the MOM defuzzification method during failure time (after time frame 100) produces oscillations in the output that usually do not match up with the monitors and with output generated from the Murphy’s rule based method. Best results were achieved with centroid defuzzification. These results followed the same pattern as the results achieved in the previous chapter.

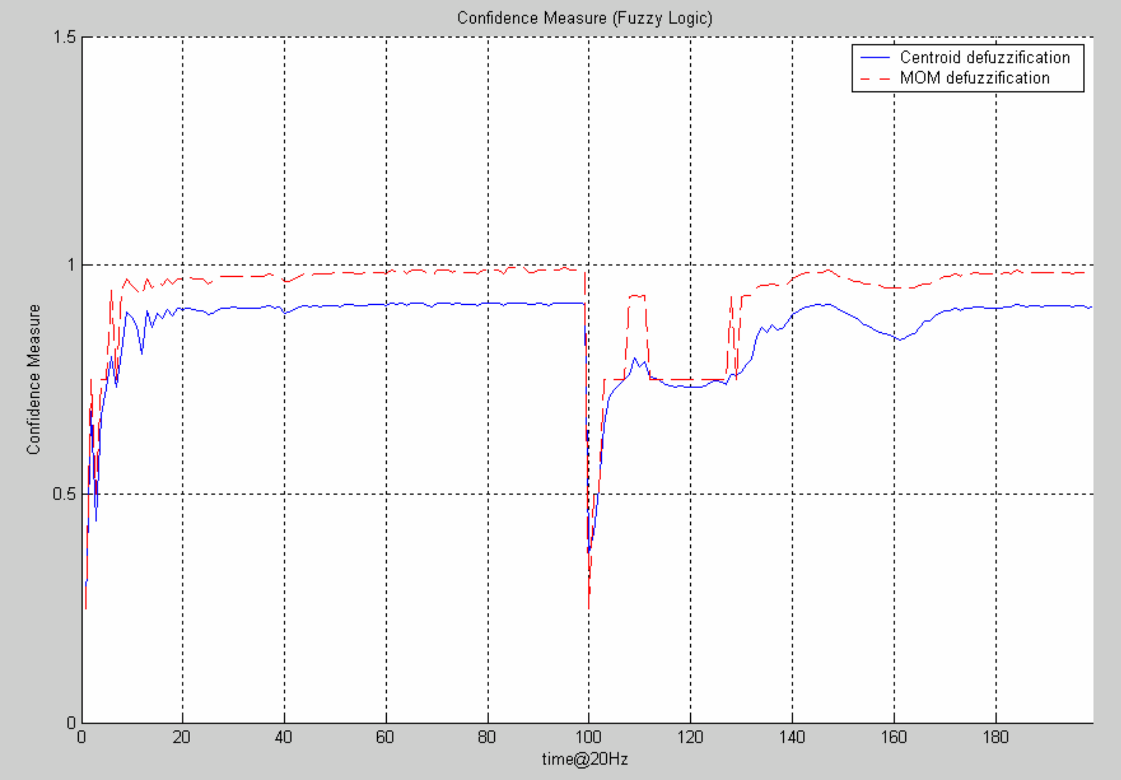

Figure 12 Comparison between centroid and MOM defuzzification

This method, based on fuzzy logic, shows very good results (see case studies). The fused data from the four monitors is simple. It indicates how much we can trust the outputs from the DCS network. Based on this simple measure decisions can be based on some thresholds. These thresholds will depend on the system where this method will be applied. Since Mamdani fuzzy models are widely used, their software or hardware implementations are easy. The IFCS system requires the data fusion method to be fast so that it will not waste processor power. The method proposed in this chapter satisfies this condition.

However, in a case when a numeric confidence measure is not required and when a linguistic output is sufficient, the defuzzification method can be excluded. Instead of that, the output with the highest possibility will be selected. This linguistic output can be used as a “traffic” light that will indicate whether there is an unexpected behavior of the DCS network, or the DCS network receives unexpected inputs. For instance, the linguistic constants can be used to signal the following lights:

- *very high* – green light (network outputs are highly trusted)
- *high* – green and yellow light
- *medium* – yellow light
- *low* – yellow and red light
- *very low* – red light (network outputs are not trusted)

CHAPTER 4: CASE STUDIES

The online DCS neural network was trained with the data collected from the high fidelity NASA-WVU F-15 flight simulator. This simulator corresponds to the architecture and the functionality of the IFCS system. The data from the simulator is sampled with a frequency of 20Hz. We have nine experiments (see Table 3), where two of them represent no-failure modes of the aircraft and the other seven represent failure modes. For every experiment the data set consist of 800 data frames and all failures happen at the 600th data frame. The DCS network is trained between the 500th and the 700th data frame, since for the monitors to react on the failure there was no need to train it on all 600 data frames before the failure. One data frame at a time was passed to the DCS network. After training the network to that input, the four monitors provide data that is fused with the two described methods. Thus, the whole system was working in real-time, providing confidence measure for the DCS network at every moment. The time on the graphs is in the range [1-200], which corresponds for the time frames [500-700] from the data set. So, the failure is shown at the time frame 100.

Mode	Failed Surface	Failure Description
1	Locked left stabilator	Locked at angle of 0 Deg
2	Locked left stabilator	Locked at angle of +3 Deg
3	Locked left stabilator	Locked at angle of -3 Deg
4	Locked right aileron	Locked at angle of +3 Deg
5	Locked right aileron	Locked at angle of -3 Deg
6	Loss of left stabilator	50% missing surface
7	Loss of right aileron	50% missing surface
8	No-failure mode 1	none
9	No-failure mode 2	none

Table 3 Failure and no-failure modes

4.1. Mode 1

The normalized monitor values in Figure 13 show spikes when the failure happens at a time frame 100. Monitors 1 and 4 show higher spikes than the other two monitors. Both, the minimum and the maximum confidence (Figure 14) from the method based on Murphy's rule fell down, which indicates that there is a failure. Also their difference increases. From the fuzzy logic method (Figure 15) we can see that after time frame 100, there is a similar pattern in the confidence level.

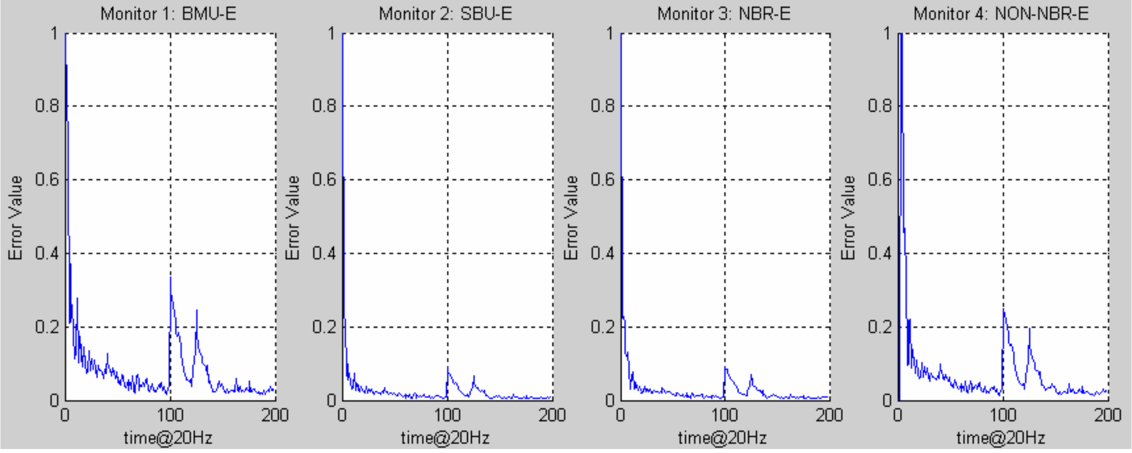

Figure 13 Normalized values from the monitors for Mode 1

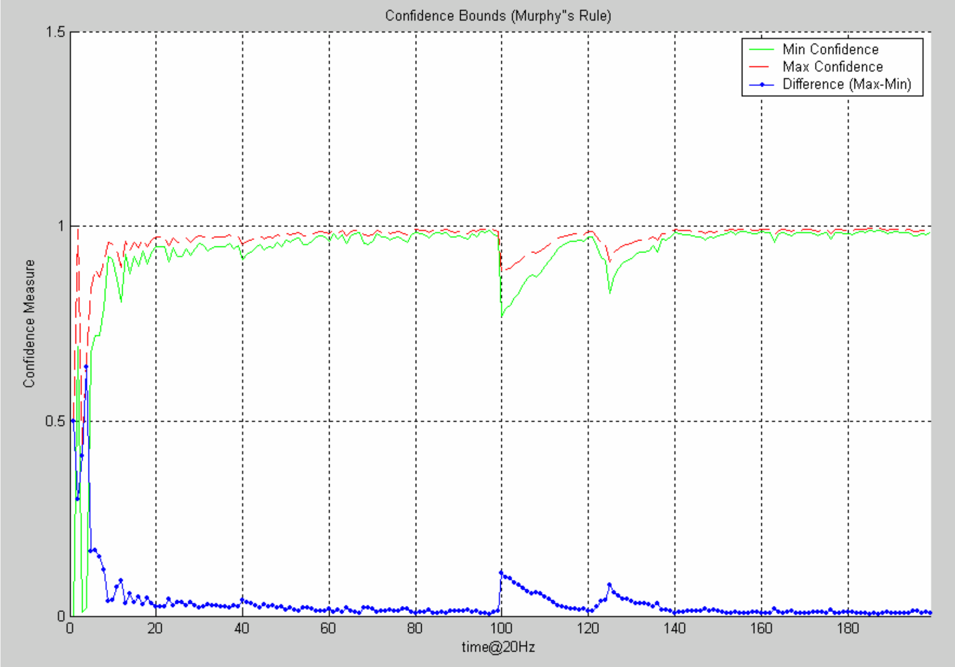

Figure 14 Data fusion using Murphy's rule for Mode 1

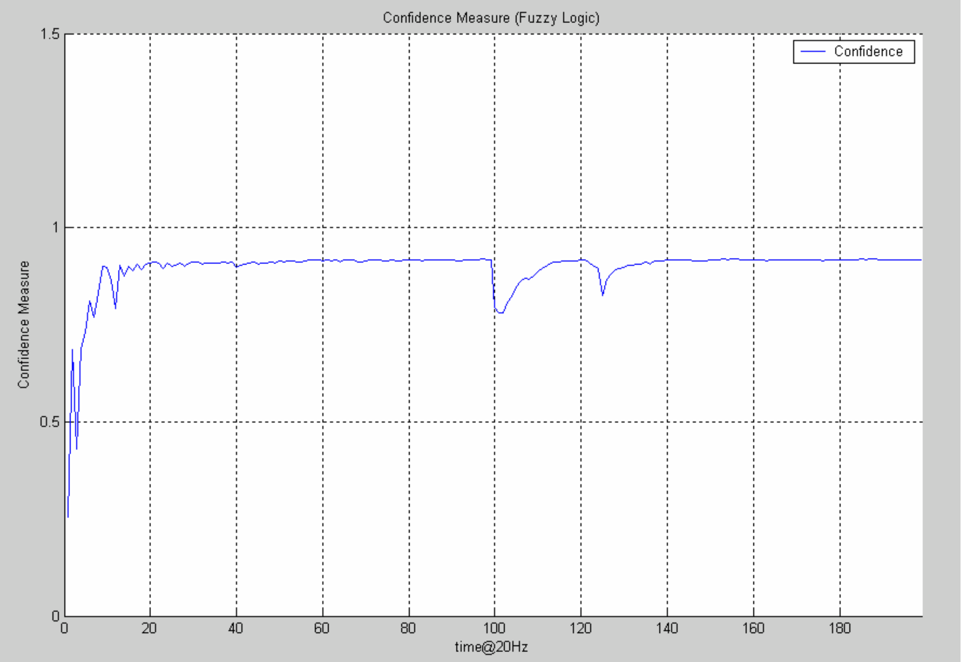

Figure 15 Data fusion using fuzzy logic for Mode 1

4.2. Mode 2

In this failure mode, the normalized monitors (Figure 16) show significant spikes when the failure happens. The highest spikes are from monitors 1 and 4, but also the spikes from the monitors 2 and 3 are significant. Figure 17 shows the results from the data fusion method based on Murphy's rule, where the minimum and the maximum confidence have a very significant drop to lower values and their distance increases at the time of failure. Also from this figure we can see that there is a period 60 time frames until the network stabilizes again. From the fuzzy logic method (Figure 18) we can notice that the similar pattern is shown. At time frame 100, there is a very low value for the confidence measure, and it is restored to a high confidence value after 60 time frames.

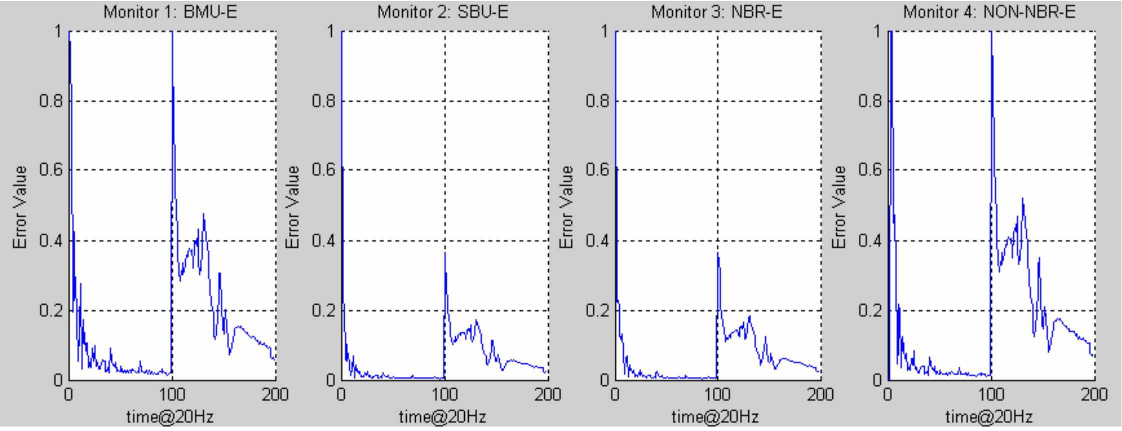

Figure 16 Normalized values from the monitors for Mode 2

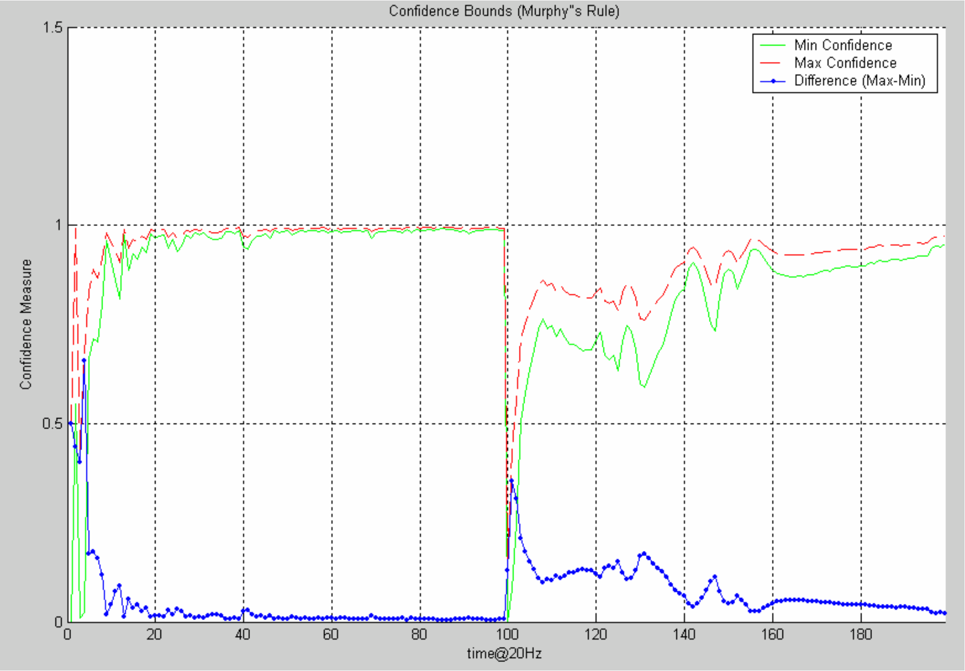

Figure 17 Data fusion using Murphy’s rule for Mode 2

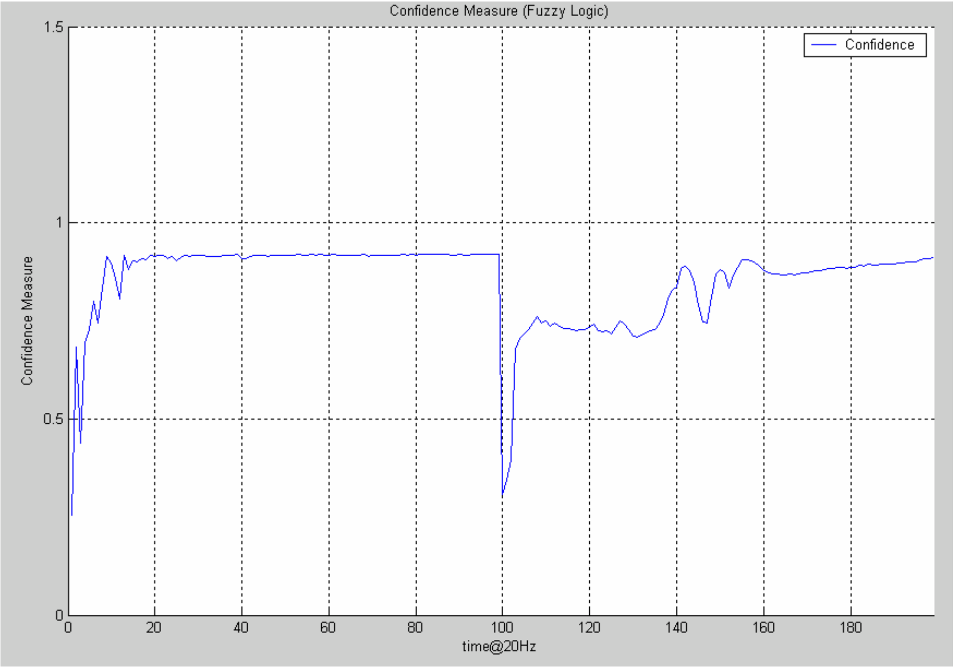

Figure 18 Data fusion using fuzzy logic for Mode 2

4.3. Mode 3

Following a similar conclusion from the previous two failure modes, monitor 1 has the highest value (Figure 19), followed by monitor 4. Monitors 2 and 3 do not show significant spikes. Murphy's rule based data fusion method (Figure 20) at the failure time for both boundaries shows a drop to a confidence level around 0.7 with their difference around 0.2 meaning that the network's outputs can not be trusted. A similar reaction of a drop to 0.7 confidence level is noticed from the fuzzy logic method (Figure 21).

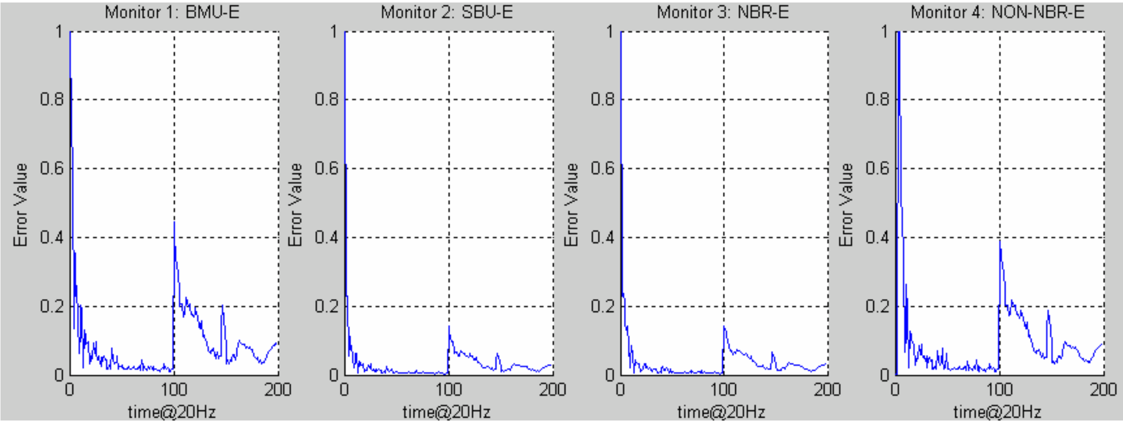

Figure 19 Normalized values from the monitors for Mode 3

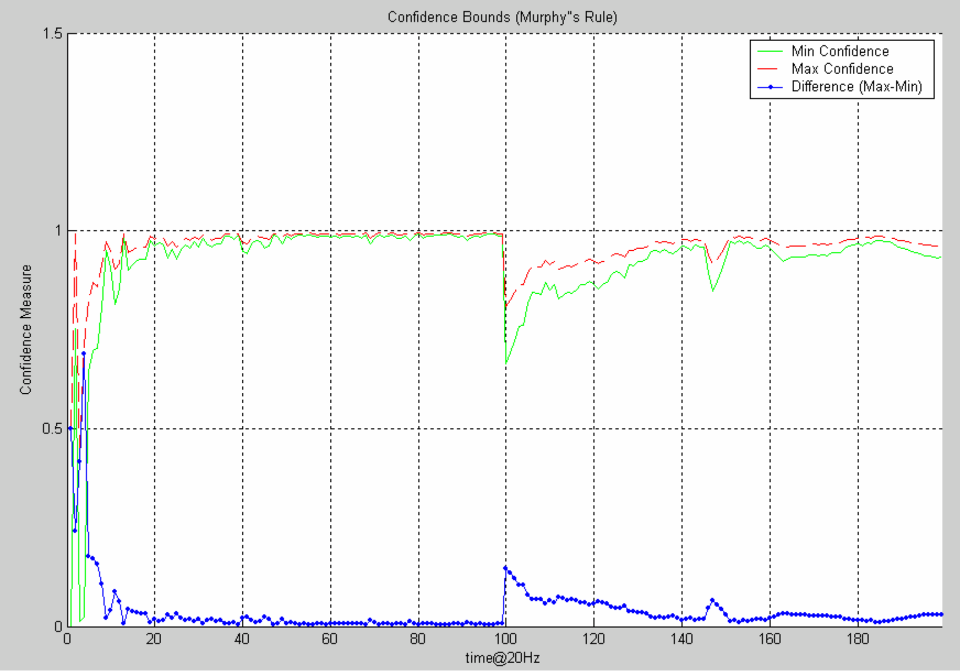

Figure 20 Data fusion using Murphy’s rule for Mode 3

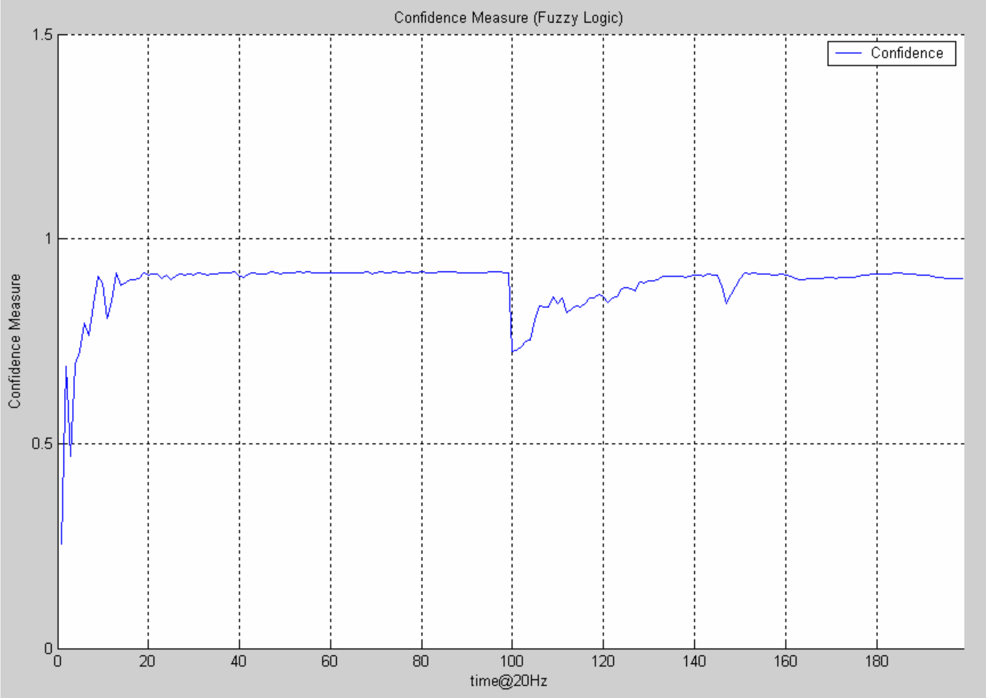

Figure 21 Data fusion using fuzzy logic for Mode 3

4.4. Mode 4

The need for different monitors can also be noticed in this failure mode, since the monitor 1 has a significantly higher value than the other 3 monitors upon the occurrence of the failure (Figure 22). This results in a drop of the minimum confidence value to around 0.7 and the maximum confidence value to around 0.9, which also affects their distance to rise to 0.2 (Figure 23). However the fuzzy logic method (Figure 24) shows only a little decrease of the confidence level.

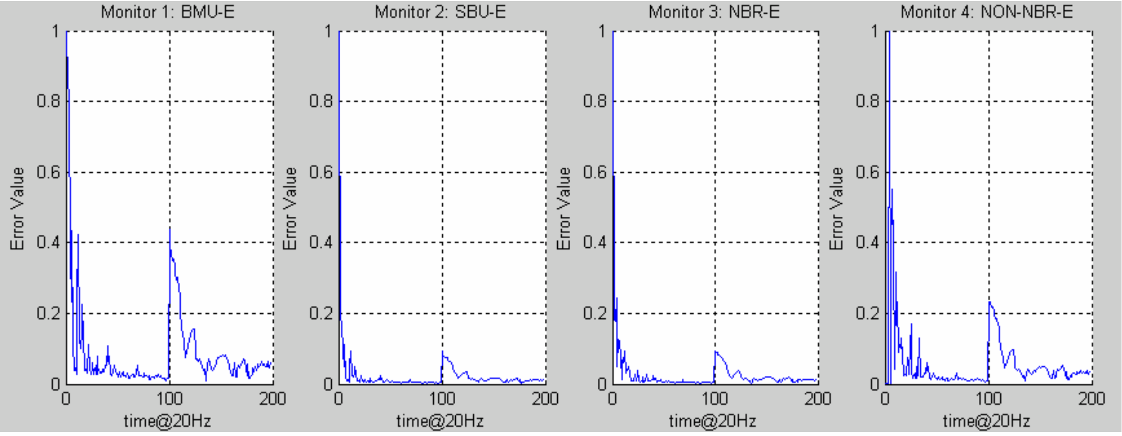

Figure 22 Normalized values from the monitors for Mode 4

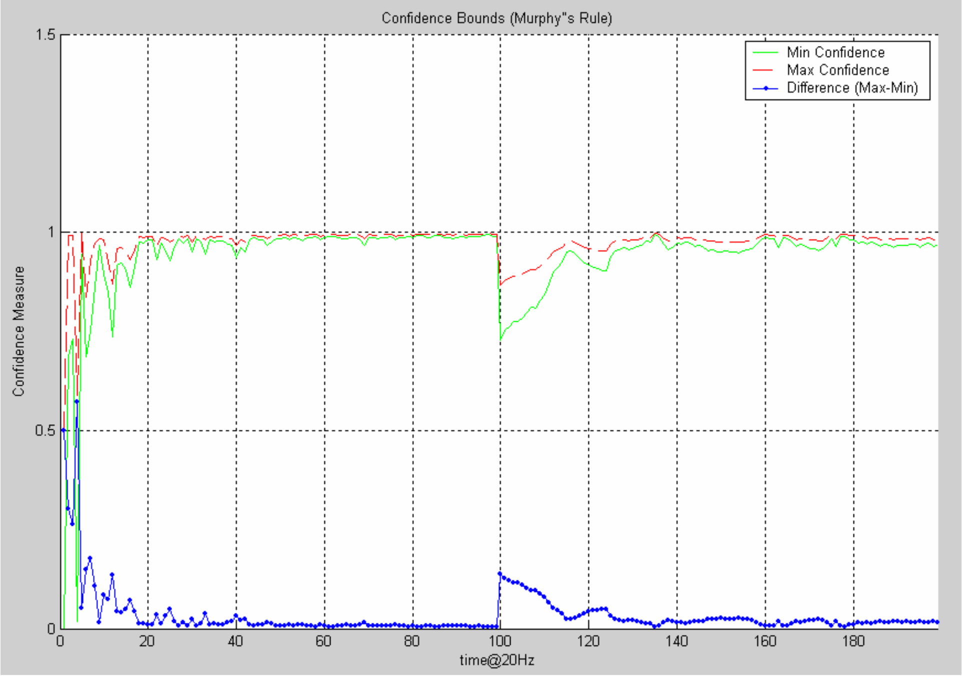

Figure 23 Data fusion using Murphy’s rule for Mode 4

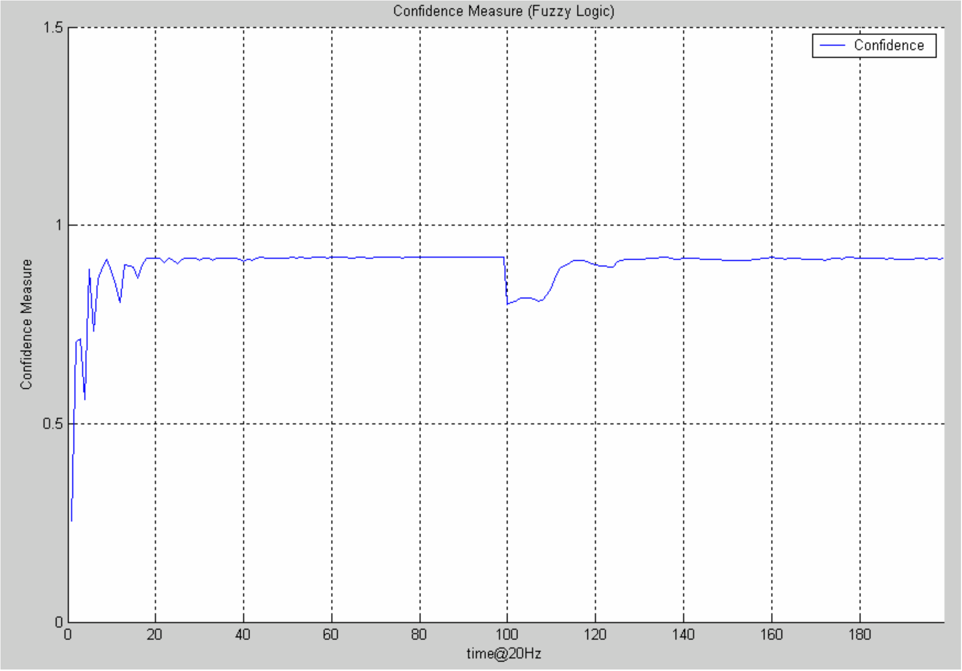

Figure 24 Data fusion using fuzzy logic for Mode 4

4.5. Mode 5

The reaction of the monitors (Figure 25) to this failure is not significant. Monitors 1 and 4 have a bit higher values than in monitors 2 and 3. When the data is fused using Murphy's rule based method (Figure 26), it shows a drop of the minimum and the maximum confidence to 0.8 and 0.9 respectively. It quickly recovers to higher values meaning that the output from the network can be ignored for a short period of time (10 time frames). Similar result is achieved using the fuzzy logic method (Figure 27).

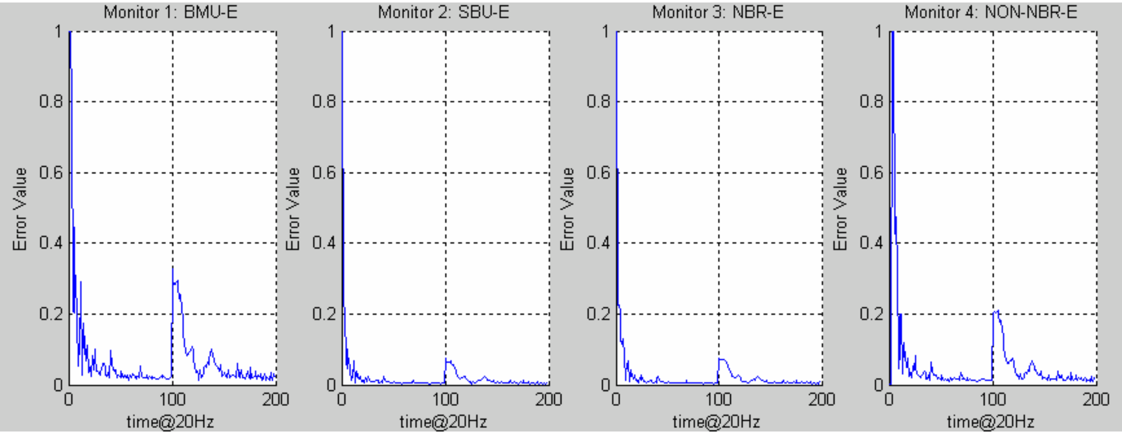

Figure 25 Normalized values from the monitors for Mode 5

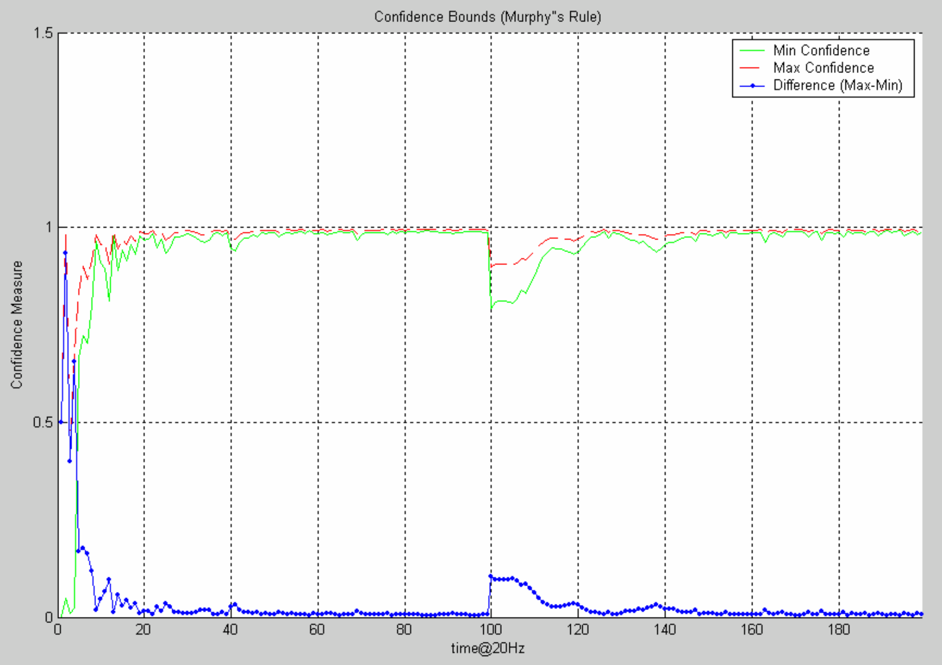

Figure 26 Data fusion using Murphy's rule for Mode 5

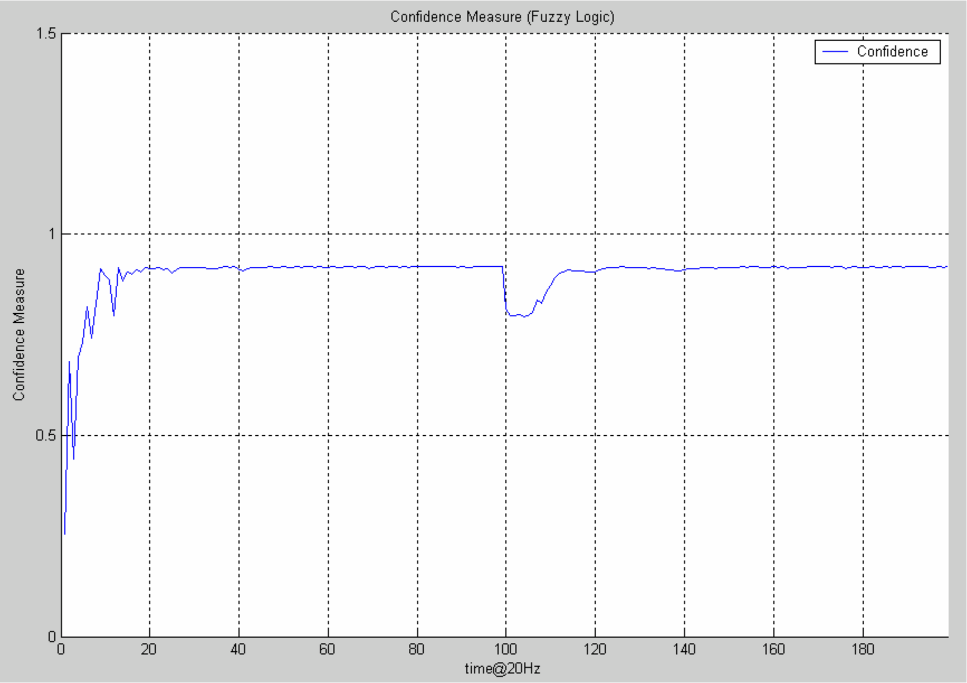

Figure 27 Data fusion using fuzzy logic for Mode 5

4.6. Mode 6

This surface failure produces very high spikes on monitors 1 and 4 and significant spikes on monitors 2 and 3 (Figure 28). Murphy's rule based method (Figure 29) shows a remarkable drop down of the confidence bounds bellow 0.4, resulting in their difference to get as high as 0.4. A significant decrease of the confidence is also produced by the fuzzy logic method (Figure 30). Both methods indicate that the network's output can not be trusted for a long period of time, approximately 60 data frames.

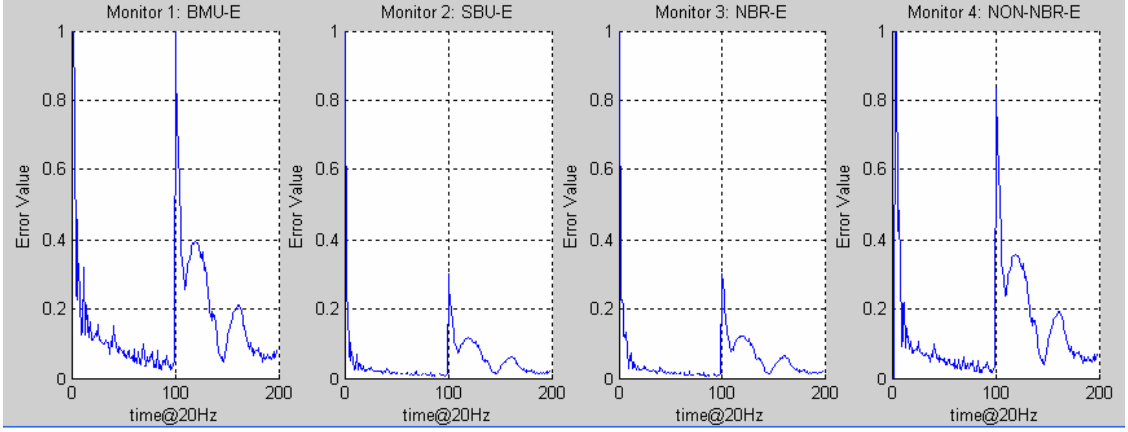

Figure 28 Normalized values from the monitors for Mode 6

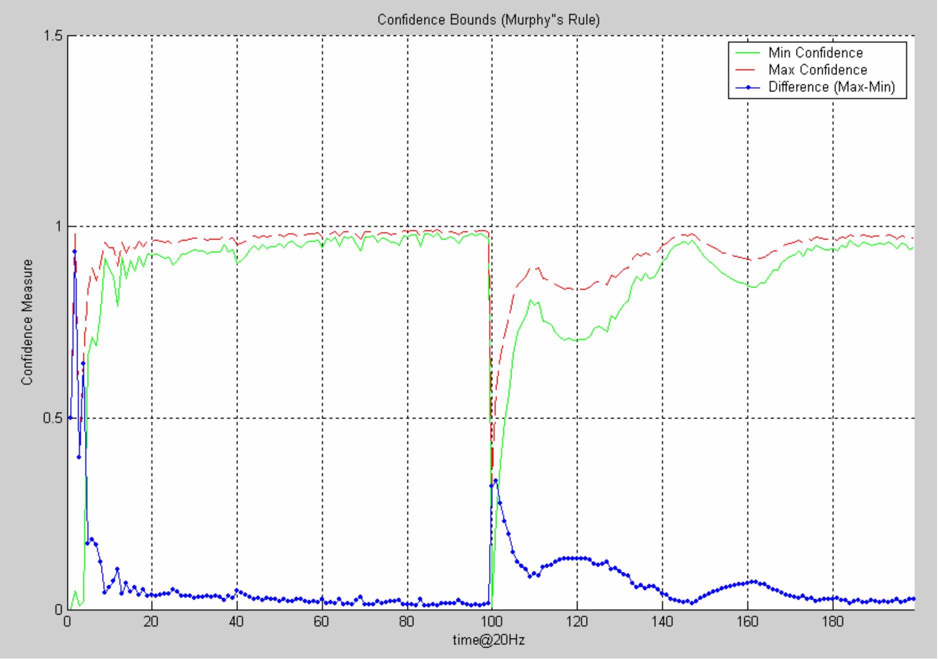

Figure 29 Data fusion using Murphy's rule for Mode 6

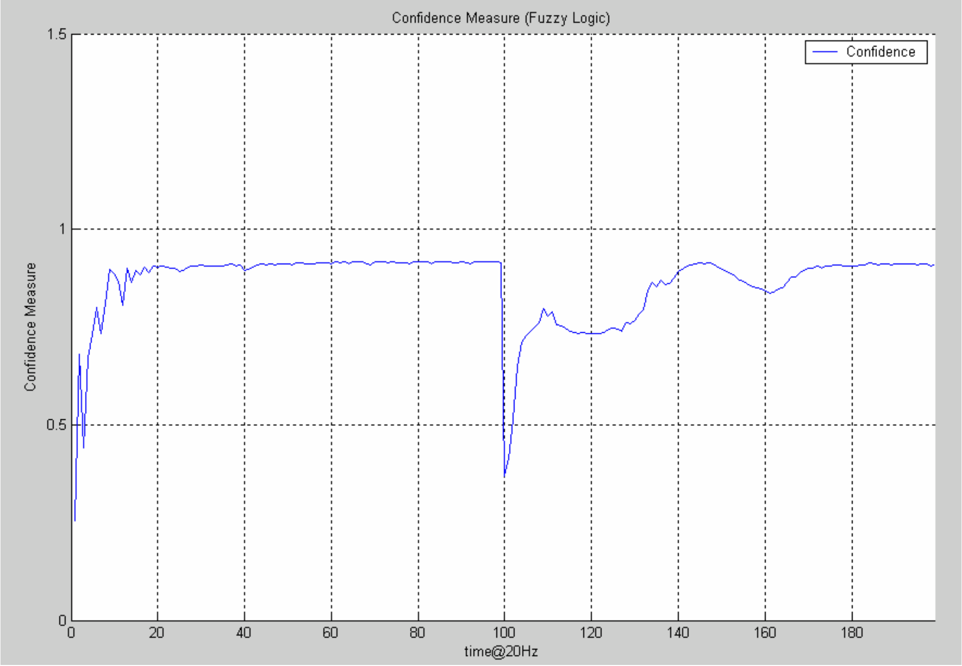

Figure 30 Data fusion using fuzzy logic for Mode 6

4.7. Mode 7

The response from the monitors (Figure 31) on this failure is up to a maximum level of 0.2. Both methods (Figure 32 and Figure 33) show a small decrease of the confidence for a short period of time after what the high confidence level is restored.

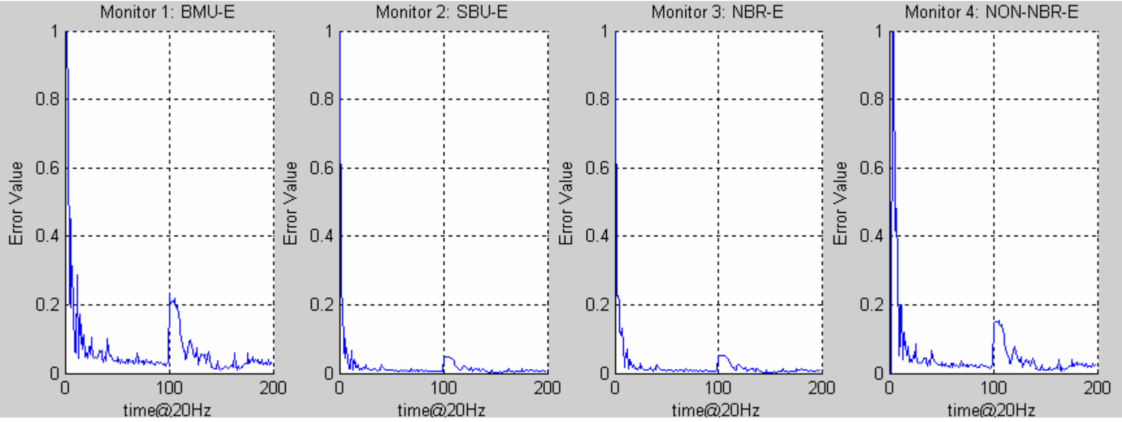

Figure 31 Normalized values from the monitors for Mode 7

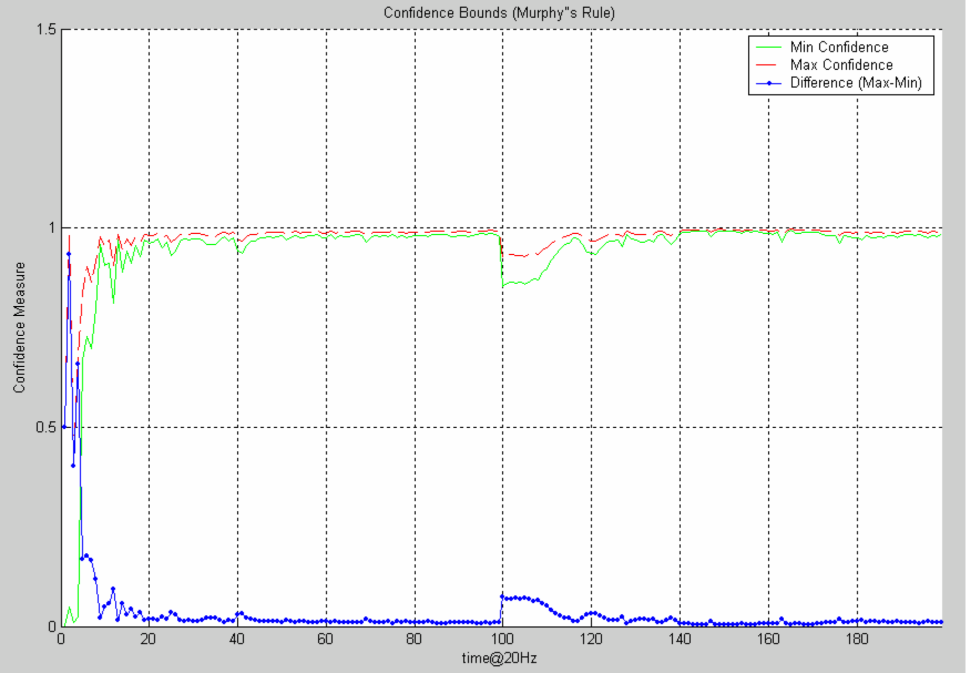

Figure 32 Data fusion using Murphy’s rule for Mode 7

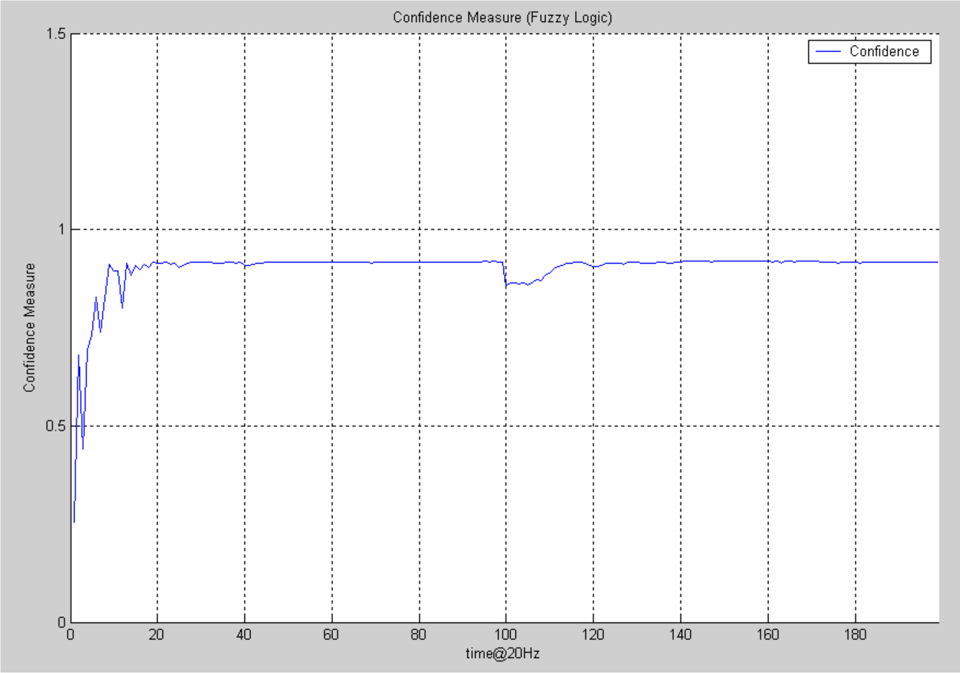

Figure 33 Data fusion using fuzzy logic for Mode 7

4.8. Mode 8

In the first no-failure mode all monitors (Figure 34) have very low values indicating that nothing bad is happening. Also the fused data (Figure 35 and Figure 36) from both methods shows a high confidence in the network's outputs meaning there are no failures. The distance between the minimum and maximum confidence is very small.

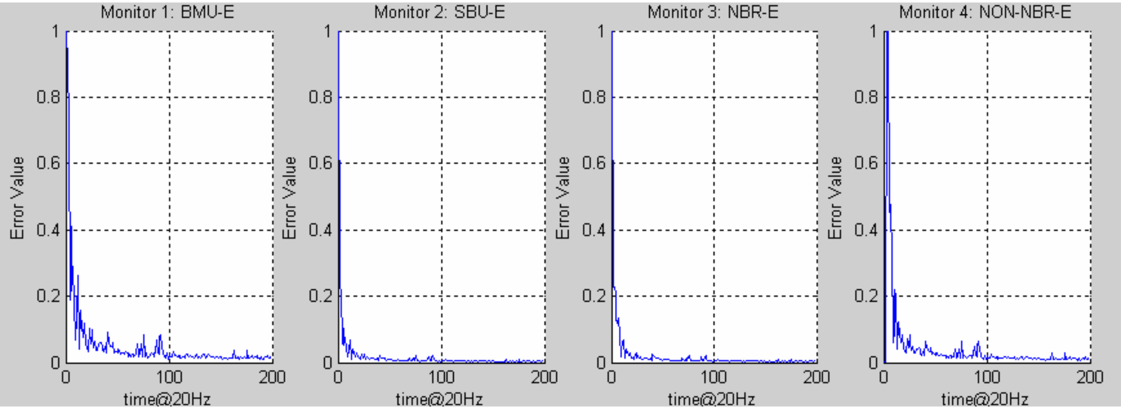

Figure 34 Normalized values from the monitors for Mode 8

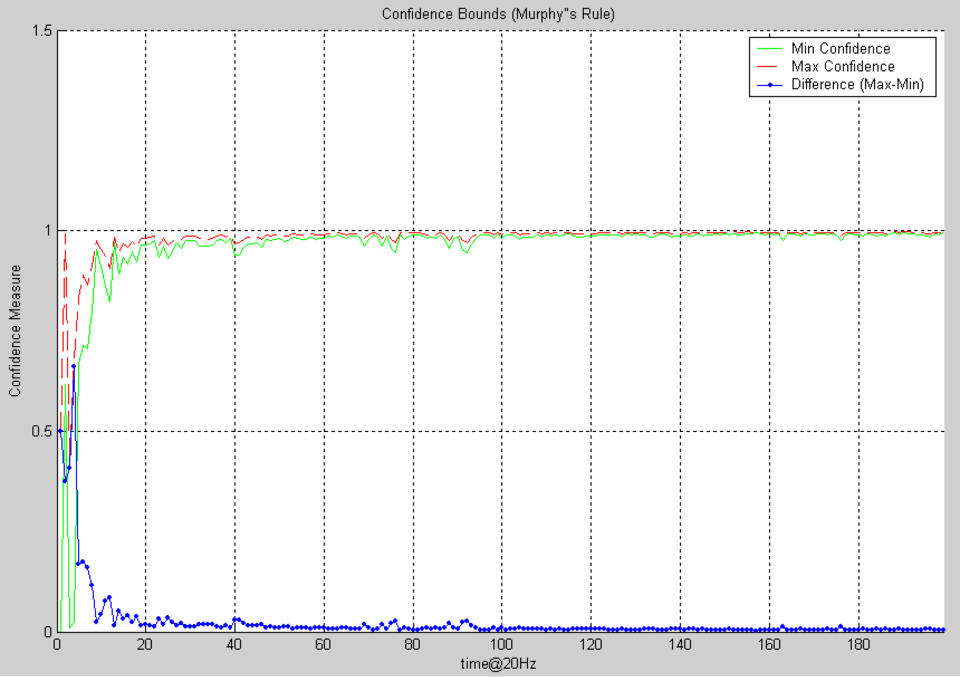

Figure 35 Data fusion using Murphy’s rule for Mode 8

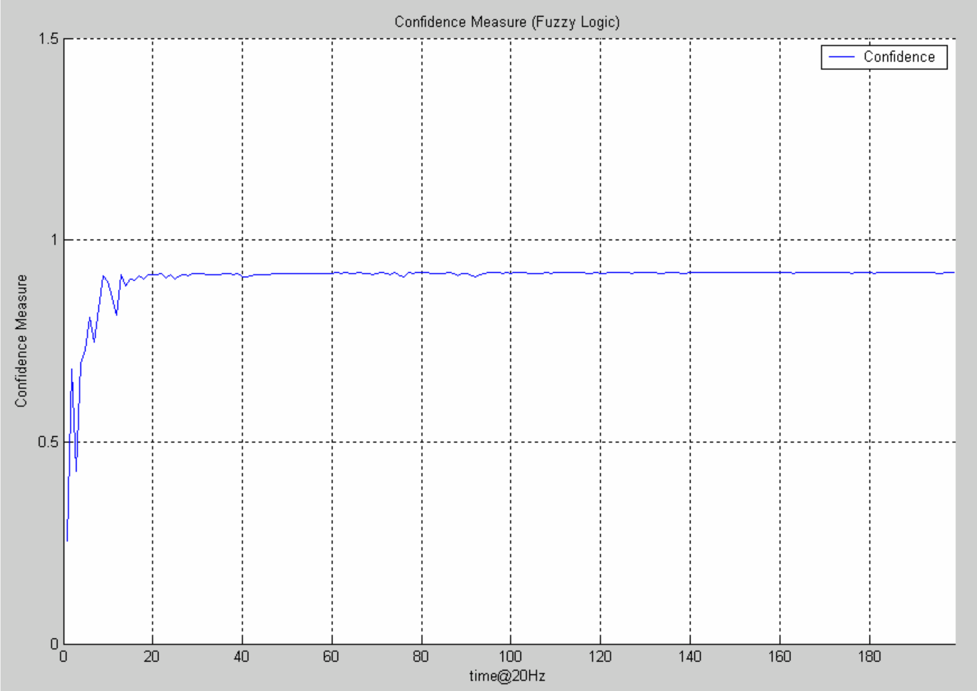

Figure 36 Data fusion using fuzzy logic for Mode 8

4.9. Mode 9

A very similar behavior like in mode 8 can be noticed in this no-failure mode. The monitors (Figure 37) show no problems with the DCS network, as well as the confidence level obtained by the two data fusion methods (Figure 38 and Figure 39).

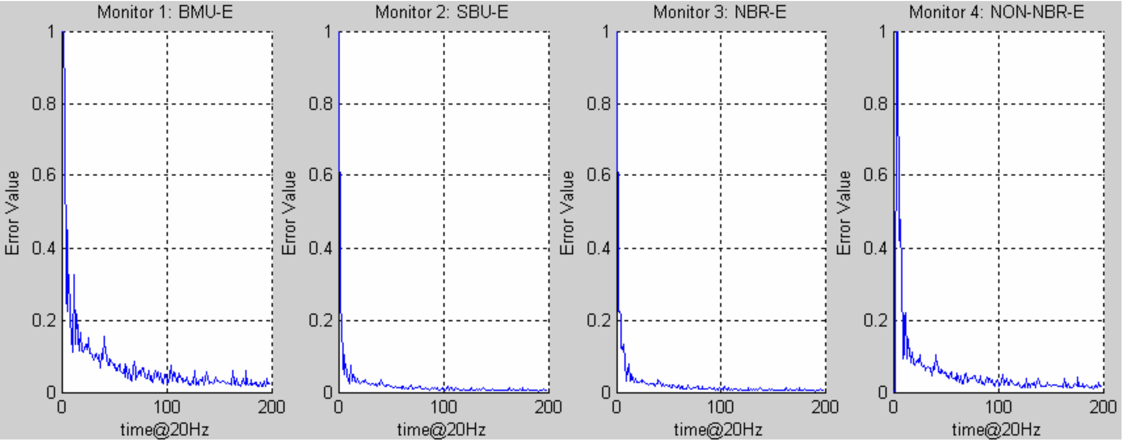

Figure 37 Normalized values from the monitors for Mode 9

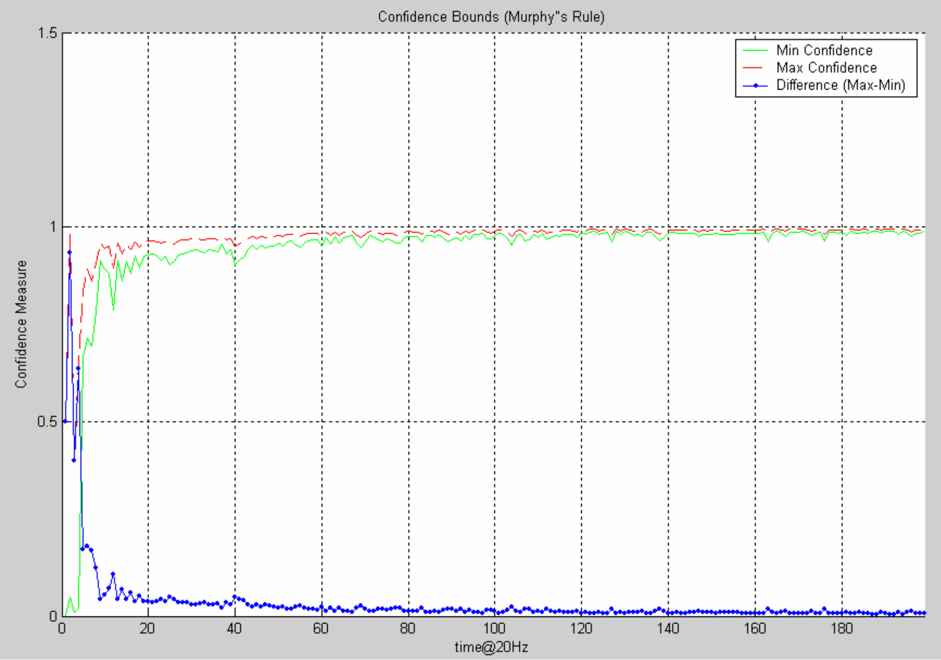

Figure 38 Data fusion using Murphy's rule for Mode 9

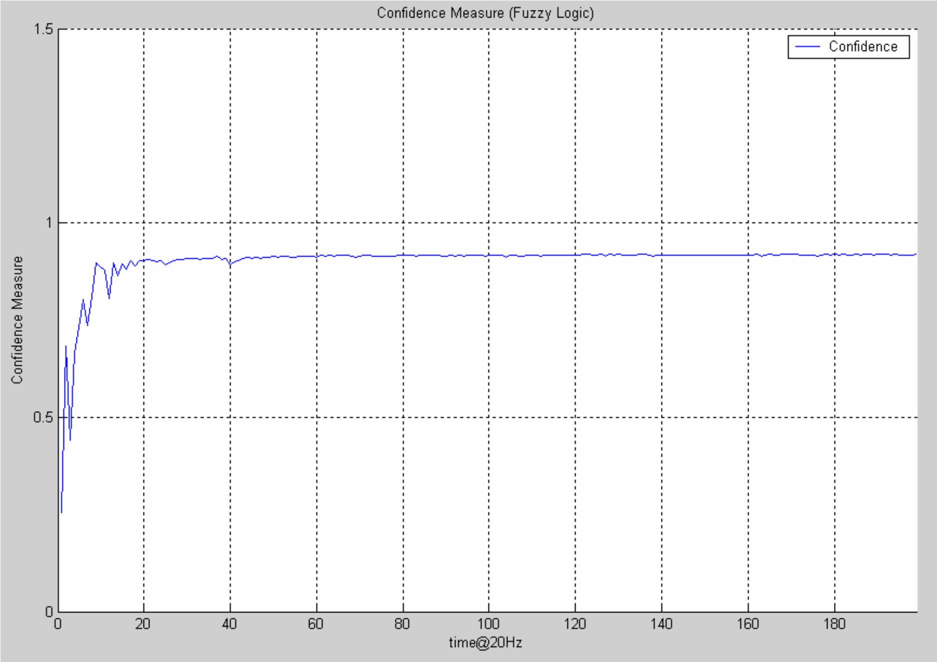

Figure 39 Data fusion using fuzzy logic for Mode 9

CONCLUSION

The importance of data fusion and its applications is enormous. Mostly it is used in complex systems that have many sensors. To be able to make conclusions or decisions based on the sensors the data needs to be fused. Such fused data is easier to understand. Data fusion is important in automated complex system, where the system has to make decisions on its own based from the data that comes from the sensors. The Intelligent Flight Control System used in our case study is an adaptive control system.

One of its important components is the online DCS neural network. Current software verification and validation methods applicable to adaptive systems are not yet scientifically mature ([15], [16]). The main contribution of our research is the development of two data fusion methods. These real time data fusion methods produce a confidence measure for the convergence trends of a DCS neural network. They use the data from four monitors, which tell how well the input data is represented by the network. The techniques elaborated in this thesis result in methods for verification and validation of a neural network based adaptive system that surpass the state-of-the-art methods.

The first method of data fusion is based on the Dempster-Shafer theory. It uses the Murphy's rule of evidence combination. Its inputs are the normalized monitor values and the output has two confidence values: the minimum and the maximum. To obtain these values we proved that if the monitor values are sorted in increasing order, we can obtain the maximum. If sorted in decreasing order, we obtain the minimum. Both values, as shown in the case studies, are good indicators of the rate of adaptation that follows the introduction of failure data into the DCS neural network. When the system is in a no-failure mode, confidence indicators stay at high values close to 1, meaning that the network reliably represents the values of actual flight parameters. In failure modes they decrease to a certain level. Furthermore, their difference provides useful information. For no-failure modes this difference is close to 0 and when a failure happens it increases.

Based on the minimum confidence, the maximum confidence and their difference an observer can decide when the output of the DCS network can/cannot be trusted.

Mamdani rule based fuzzy model was used as another data fusion method. The inputs to this method are normalized (in online fashion) monitor values and the output is a single confidence value. The input space was partitioned into three partitions: good, normal and bad; while the output space was partitioned into: very low, low, medium, high and very high. As seen from the case studies in no-failure modes the output value of this method is closer to 1 meaning that the DCS network can be trusted. When failure happens the confidence value drops down to a certain level, meaning that the network can not be trusted for that period of time.

An important fact is that both of these methods are very fast and can be easily implemented in the IFCS, providing a real time assessment. Murphy's rule based method provides more information about the state of the DCS network, than the single confidence measure provided by fuzzy logic method. However if the deffuzification method is removed and only linguistic output values are used, the fuzzy logic method can provide output as simple traffic light with green light meaning network can be trusted and red light meaning network cannot be trusted. Besides applying these two methods on the IFCS system, we can apply them in other systems where information fusion is needed.

An area of future work is to tune up the parameter n of the Murphy's rule. The fine tuning of this parameter depends on many factors. Conducting many experiments (failure and no-failure modes of the aircraft) to determine the correlation between the monitors of the DCS neural network can be the first step of adjusting this parameter. The second step could be grouping the monitors into independent groups of monitors that are highly correlated between them. Based on these correlation numbers, for each group appropriate value of n can be selected. Combining the information provided by the monitors can be done at two levels. First level will be calculating the combined value of each group of monitors. The values of each group can be combined at the second level with $n = 0.5$, since the groups are independent. Adding more monitors to the DCS neural network can increase the number of groups or the number of monitors in the groups, thus leading to better selected values of n.

Another area of future work is adjusting or choosing new membership functions for the fuzzy logic based model, which will provide better results. The output space can be divided into more partitions resulting in more precise output values. Fuzzy rules can be altered if many experiments will show that some monitors are significant or insignificant.

BIBLIOGRAPHY

[1] Institute for Scientific Research, Inc. Dynamic Cell Structure Neural Network Report for the Intelligent Flight Control Program, January 4th, 2001

[2] Sampath Yerramalla, Edgar Fuller, Martin Mladenovski, Bojan Cukic. Lyapunov Analysis of Neural Network Stability in an Adaptive Flight Control System. *Sixth Symposium of Self-Stabilization Systems (SSS)*, June 2003

[3] Sampath Yerramalla, Edgar Fuller, Bojan Cukic. A Validation Approach for Neural Network Based Online Self-adaptive Systems. April 2004 (submitted)

[4] Sampath Yerramalla, Yan Liu, Edgar Fuller, Bojan Cukic. An Approach to V&V of Embedded Adaptive Systems. *NASA-Goddard / IEEE Workshop on Formal Approaches to Agent Based Systems*, Feb 2004

[5] Yifeng Zhou, Henry Leung. Minimum Entropy Approach for Multisensor Data Fusion. *Proceedings of the 1997 IEEE Signal Processing Workshop on Higher Order Statistics (SPW-HOS '97)*, 1997

[6] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976

[7] Phillipe Smets. The Combination of Evidence in the Transferable Belief Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, No. 5, May 1990

[8] Philippe Smets. The Transferable Belief Model and other Interpretations of Dempster-Shafer's Model. *Proceedings of the Sixth Annual Conference on Uncertainty in Artificial Intelligence*, 1990, pp: 375 – 384.

[9] Robin R. Murphy. Adaptive Rule of Combinations for Observations Over Time. *Fusion and Integration for Intelligent Systems (MFI96)*, Dec. 8-11, 1996, Washington DC,pp.125-131.

[10] Robin R. Murphy. Dempster-Shafer Theory for Sensor Fusion in Autonomous Mobile Robots. *IEEE Transactions on Robotics and Automation*, Vol. 14, Issue 2, Apr 1998, pp: 197-206

[11] John Yen, Reza Langari. *Fuzzy Logic: Intelligence, Control and Information*. Prentice Hall, 1998

[12] Ludmila I. Kuncheva. How good are fuzzy if-then classifiers? *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 30 (4), 2000, 501-509.

[13] Frank Hoffmann, Oliver Nelles. Structure Identification of TSK-Fuzzy Systems using Genetic Programming. *Proceedings Information Processing and Management of Uncertainty*, IPMU'2000, Madrid, Spain.

[14] B. Solaiman, L.E. Pierce, F.T. Ulaby. Multisensor Data Fusion Using Fuzzy Concepts: Application to Land-Cover Classification Using ERS-1/JERS-1 SAR Composites. *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 37, Issue 3, May 1999

[15] Johann Schumann, Pramod Gupta, Stacy Nelson. On Verification & Validation of Neural Network Based Controllers. *In Proceedings EANN 03*, 2003

[16] Johann Schumann, Stacy D. Nelson. Toward V&V of neural network based controllers. *WOSS* 2002, pp: 67-72