FEATURE EXTRACTION AND SENSOR OPTIMIZATION FOR CONDITION MONITORING OF RECIRCULATION FANS AND FILTERS

M. Gerdes, D. Scholz
Hamburg University of Applied Sciences
Aero - Aircraft Design and Systems Group
Berliner Tor 9, 20099 Hamburg, Germany

Abstract

Complex systems are commonly monitored by many sensors. When one of the sensors fails, the current state of the system can not be calculated or the information about the current state is not complete. For that reason sensor failures are one of the main error sources of a system. Thus sensors that deliver significant information about the system state need to be redundant. This paper shows how to calculate the significance of the information that a sensor gives about a system by using modern signal processing and artificial intelligence technologies. It also shows how significant features can be extracted, evaluated from a set of data samples, how difficult it is to find an optimal parameter and sensor set and that it is possible to reduce the size of needed data by 97%. The paper concludes analyzing the results of experiments that show that the methods can classify different errors with a 75% probability.

1. INTRODUCTION

New and better monitoring approaches are required for condition monitoring, because systems become complexer. This is especially true for aircraft systems, where a high availability is required. The air conditioning system is interesting to monitor, because it not only supplies the passengers with fresh and tempered air; it also cools most of the electronic systems in the aircraft. Recirculation fans and filters are used for removing and cleaning a part of the used air from the cabiny and direct the used air into the air mixing chamber, where it is mixed with fresh air and returned into the cabin. The recirculation filters can get clogged and the fans can get broken. If this happens the comfort of the passengers is reduced, because clogged filters smell and a broken fan causes smoke. It is difficult to predict, when a filter gets clogged, because the clogging of the filter depends on the usage of the aircraft. If a filter gets clogged then the fans might get broken faster and thus cause more costs and damage. Monitoring both components can save a lot of trouble. The recirculation fan and filter system consists of active (fans) and passive (filter) parts and is one of the few systems in the aircraft that uses condition monitoring.
Condition monitoring requires reliable sensors. If a sensor fails the information about the system is incomplete and thus it is recommended to install significant sensors redundantly. One problem of current condition monitoring technologies is, that it is difficult to calculate which sensors give significant data about the condition of a system and thus have to be installed redundantly. Sensor data needs to be processed to gain more information, than just time domain values. Frequency data and information about patterns in the sensor data can be useful to gain information about the condition of a system.

2. CONDITION MONITORING OF RECIRCULATION FANS AND FILTERS

Today only a limited number of methods for monitoring the condition of recirculation fans and filters is used on aircraft. Older aircraft monitor only very rudimentary the air conditioning system, while newer aircraft sometimes use trend monitoring for the air conditioning. Most of the technologies that are used for fan and filter monitoring have disadvantages that make them not suited for complex condition monitoring. Some of the most common technologies, for recirculation fan and filter monitoring, will be explained below.

Pressure Switch: On many aircraft pressure switches are used to monitor recirculation fans or filters. These switches generate a failure message, if the pressure reaches a defined threshold [1].

Differential Pressure Sensor: Newer aircraft use differential pressure sensor to monitor filters and fans. The pressure difference is used as an indicator for the clogging of the filter. To get a clogging level and a forecast, measured values are compared against a curve, which relates the pressure difference to clogging. A filter clogging can be detected up to 100 hours in advance [1].

Vibration Sensor: Boeing uses on the 767 and 747 [2] vibration sensing systems to monitor fans. These systems are mounted on top of a fan and shut the fan off, if a certain vibration intensity is exceed. This concept tries to prevent serious damage to the air

conditioning subsystems.

All presented monitoring concepts use the raw sensor data without additional signal processing and can only perform simple monitoring tasks. They operate without significant knowledge of the system and often only differentiate between a no fault and faulty state. To be able to get more knowledge about the system, more and different sensors have to be applied and the sensor data needs to be processed further.

3. BACKGROUND

The methods that are presented and used in this paper are from the field of information theory and artificial intelligence. This section will show the basic principles that are used for the concept in Section 4.

3.1. Information Gain

Information entropy is the knowledge that is contained in an answer depending on one's prior knowledge. The less is known, the more information is provided. In information theory information entropy is measured in bits. One bit of information entropy is enough to answer a yes/no question about which one has no data [3]. The information entropy is also called information and is calculated as shown below in equation 1. $P(v_i)$ is the probability of the answer v_i.

$$(1) \quad I(P(v_1),\ldots,P(v_n)) = \sum_{i=1}^{n} -P(v_i)log_2P(v_i)$$

The information gain from an attribute test (setting the value of a node in a tree, see Figure 3.2 for an example) is the difference between the total information entropy requirement (the amount of information entropy that was needed before the test) and the new information entropy requirement. p is the number of positive answers and n is the number of negative answers [3].

$$(2) \quad Gain(X) = I(\frac{p}{p+n},\frac{n}{p+n}) - \sum_{i=1}^{v}\frac{p_i+n_i}{p+n}\cdot I(\frac{p_i}{p_i+n_i},\frac{n_i}{p_i+n_i})$$

3.2. Decision Trees

Decision trees are a method from the area of artificial intelligence and are used for machine learning. They are often binary trees, where each node has an if-then-else function on an attribute of the sample data. The ID3 algorithm (Iterative Dichotomiser 3, published by J. Ross Quinlan in 1986, used to generate decision trees [5]) was the first algorithm to construct decision trees. ID3 had some problems and was improved. The improved version of ID3 is C4.5 [4]. It enhances the ID3 algorithm with the ability to handle both discrete and continues attributes, it can handle samples with missing attributes and supports pruning of the tree at the end of the algorithm (removing branches from the tree). The algorithm uses the concept of information gain to choose attributes from the data and build a decision tree. The algorithm in pseudo code is:

1. Check for base cases
2. For each attribute a
   a) Find the normalized information gain from splitting on a (see below)
3. Let a_best be the attribute with the highest normalized information gain
4. Create a decision node that splits on a_best
5. Recurse on the sub-lists obtained by splitting on a_best, and add those nodes as children of node

The result of the algorithm is a binary decision tree, where the root of the tree is the attribute with the highest normalized information gain. Nodes in the following levels of the tree represent attributes with lower normalized information gain. If pure information gain is used for splitting, then classes with the most cases are favored [4]. C4.5 uses the normalized information gain or the gain ratio. Split info is the information that is gained from choosing the attribute to split the samples.

$$(3) \quad Split\ Info(X) = -\sum_{i=1}^{n}\frac{p_i+n_i}{p+n}log_2(\frac{p_i+n_i}{p+n})$$

Gain ratio is the normalized information gain and is defined as shown in equation 4 [4].

$$(4) \quad Gain\ Ratio(X) = \frac{Gain(X)}{Split\ Info(X)}$$

A resulting decision tree (for the decision if you can play outside) may look like the tree in figure 3.2.

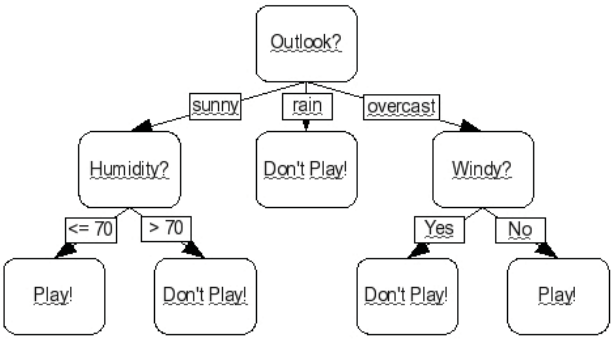

Figure 1: Example Decision Tree

3.3. Feature Extraction

The standard application of decision trees is pattern recognition (classification), learning and decision making. The tree is used to make a decision for new data, based on old knowledge. Either a sample is classified based on prior analyzed data or a decision is made based on prior 'experience'.
Using a decision tree for feature extraction works a bit different. The goal is not to classify new data, instead the goal is to build a tree, classify given data and find the attributes with the highest information entropy. During testing, some attributes can be disabled or deleted and their influence on the condition monitoring can be tested.

4. SENSOR OPTIMIZATION BY FEATURE EXTRACTION FOR PAHMIR

This section shows how the concepts of the preceding sections are applied for high pressure recirculation fan and filter monitoring within the PAHMIR project (Preventive Aircraft Health Monitoring for Integrated Reconfiguration [7]). Sensor optimization is a very wide topic and includes a number of different definitions. A few different meanings for sensor optimization are:

- Optimizing the position of sensors [8].
- Optimizing the processing of sensor data [9].
- Optimizing the information gain of sensors.

In this paper sensor optimization has the meaning of sensor optimization, where identifing significant sensors in a number of available sensors that give the most information about a system, and thus increasing the information gain, is the goal. Signal processing methods are also counting as sensors in this context (e.g. frequency information and average values). Optimization in this paper is finding the sensors with the highest information gain of the data. The data can be raw sensor data, like amplitudes or processed data like frequency data. The calculation of the *information gain* and the sorting of the data is done by the *C4.5* algorithm that is used to construct decision trees for classification problems. Data samples are first processed, then the C4.5 algorithm is applied and finally the data is analyzed. Raw data samples are normally processed before they are used as an input. This means that the data is transformed into the correct input format for algorithms, it is enhanced with additional information and it is even possible to compress the data as it will be shown. Processing is done with Matlab and does not use any special toolboxes.
Data samples can generally be divided up into two categories of data: high frequency data and low frequency data. Low frequency data is defined as data with a sampling frequency less then 20 Hz. High frequency data is any data with a higher sampling rate then 20 Hz.
The low frequency data will not be processed beside bringing the data into the correct data format for the algorithm. There is too little data to perform significant frequency analysis and compression.
The high frequency data will be processed with the following steps: First the data is transformed into the frequency domain, then noise reduction is applied to the data, after that the frequency data is partitioned into small blocks (the size of the blocks depend on the data see Section 6.1.2)and finally every block is enhanced with additional information.

4.1. Fast Fourier Transformation and Partitioning

The fast Fourier transformation takes a number of time-domain samples and transforms them into the frequency domain. Basis of the FFT algorithm is the discrete Fourier transformation, which is defined as shown in equation 5 with $x_n, \ldots, X_{N-1}$ as complex numbers.

$$X_k = \sum_{n=o}^{N-1} x_n e^{-i2\pi k \frac{n}{N}} \quad k = 0, \ldots, N-1 \tag{5}$$

A fast Fourier transformation is performed in O(N log N) operations [6]. The fast Fourier transformation (FFT) is done with the Matlab function fft. A full transformation with the sampling frequency is done. After the fast Fourier transformation is done, the frequencies are divided up into blocks. The number of the frequencies that are grouped in one block is determined by the calculation parameter *Block Width*. If less then *Block Width* frequencies are available, then all frequencies are treated as one block. After partitioning all blocks are transformed back into the time domain, to get information about the behavior of the block-signal over the time.

4.2. Noise Reduction

Noise reduction is applied to the signal to remove random data from the samples in order to improve the feature detection of the undisturbed signal. The maximum frequency power is calculated and then every frequency signal that is below a defined fraction of the maximum frequency power is reduced to zero to remove noise from the sample. The exact fraction of the maximum frequency power for noise reduction is a parameter of the experiments (*Noise Reduction Factor*). Noise reduction is done as shown in the Matlab code 1.

```
Y = fft(y);
x = mean(abs(Y))*NoiseReductionFactor;
Y = Y .* (abs(Y)>x);
```

Listing 1: Noise Reduction

4.3. Additional Information and Data Compression

Every block of the sampled data is enhanced with additional information. This information is added to give the following algorithm more information about the signal in the time and the frequency domain. The added information is for the time domain:

- The maximum amplitude of each block
- The mean amplitude of each block

In the frequency domain the following information is added:

- The mean frequency power of each block
- The maximum frequency power of each block
- The frequency with the highest power of each block
- The number of peaks that are higher then a defined magnitude of the mean frequency power

The Matlab code 2 shows how the peaks are calculated. *peakBorder* is the parameter that can be varied and it defines, when a spike counts as a peak.

```
currPeakNum = 0;
for X = 1:blockWidth
    if (Y_block(X) >= meanPower*peakBorder)
        peaks_block = peaks_block+1;
    end
end
```

Listing 2: Peak Calculation

The additional information is also calculated for the complete signal sample. Experiments showed that the added information is more useful for the algorithm, then the actual raw data. This allows to compress the data. The information of 100 frequencies (100 frequencies are used to simpler convert the result into percent values) is thus reduced down to four attributes (maximum and mean power, the frequency with he maximum power and the number of peaks). A similar result is achieved in the time domain. Instead of calculating the amplitude for each frequency in the time domain, only two attributes (maximum and mean amplitude) are calculated for 100 frequencies.

$$Freq\_Info = 4 \cdot \frac{Frequencies}{BlockWidth} \tag{6}$$

$$Time\_Info = 2 \cdot \frac{Frequencies}{BlockWidth} \tag{7}$$

$$\begin{aligned} Total\_Info &= Freq\_Info \\ &\quad + Time\_Info \\ &= 6 \cdot \frac{Frequencies}{BlockWidth} \end{aligned} \tag{8}$$

$$Normal\_Info = 2 \cdot Frequencies \tag{9}$$

$$\begin{aligned} Compression &= \frac{Total\_Info}{Normal\_Info} \\ &= \frac{3}{BlockWidth} \end{aligned} \tag{10}$$

Thus the needed data is reduced to 3 % if $BlockWidth = 100$ and $Frequencies = 11000$. The results of the analysis of the data with and without ($BlockWidth = 1$) compression will be shown in the section 'Result Analysis'.

5. EXPERIMENT

To show the performance and concepts of the algorithm, experiments were performed with different parameters. The data for the experiments and the feature extraction was sampled with an autonomous box, that contained sensors and logic to save the data on a SD card. As a basis for the data collection a test rig was used. Vibration data with a sampling rate of 44 kHz of a simple PC fan was collected. A PC fan was used to show the principals of the method. Data is saved in a raw wave format onto a SD card and then transferred onto a PC. In addition to the raw sensor data the condition of the component was saved. The fan is operated with standard speed, but three different conditions were sampled. Data from the following conditions was collected:

- No additional weight
- A very small weight (less then one gramm) is added to one blade
- A small coin (one Eurocent) is added to one blade

For each case 900 samples were collected. Every sample contains the vibration data of one second. Ten minutes passed between the individual samples. Samples were collected during office work hours and so a variety of noise is contained in the samples. In the experiment 900 “No weight” (no additional weight), 450 “Small weight” (a very small weight) and 450 “Big weight” (a small coin) samples were used. The decision tree of the J48 algorithm (an implemen-

tation of C4.5) in WEKA was validated with a 3-fold-cross-validation (all samples are used for testing and training and the cross-validation process is repeated 3 times).

5.1. Calculating the Decision Tree

The decision tree is calculated with the open source Java software WEKA [10]. WEKA allows the user to test different algorithms and shows the classification errors that occurred. The correct data format is generated by using a Java program that transforms the output files from Matlab into input files for WEKA. For classification J48 is chosen, which is an implementation of the C4.5 decision tree algorithm, and a confidence factor of 0.0001. The confidence factor defines how much pruning is done to the resulting decision tree.
The complete processed data is used as training data. After the generation of the decision tree the same data is used for testing the decision tree. In general the training and the testing data should not be the same, but in this case it is exactly what is wanted. The goal is not to classify new objects correctly, but to check how good the available data is classified and what part of the data gives us the most information about the system.

5.2. Experiment Parameters

Calculations with the same input data, but different parameter values, were performed to show the influence of the parameters on the results; Table 1 shows the available parameters with their possible values. All "Yes/No"-parameters are Boolean parameters, that toggle the calculation of that parameter during the processing. Default parameters are the values that are used, when the effect of a parameter onto the algorithm is tested. Only one value per test varies, while all other parameters keep their default value. The data processing with Matlab generates a number of different input sets for the J48 algorithm. For every input set a decision tree is generated and the influence of the modified parameter is then evaluated.

6. RESULT ANALYSIS

This section analyzes the results of the data processing of the previous section. First the different experiments with the different parameters are analyzed and evaluated in the next step the results of the best parameter configuration are taken and analyzed more closely.

6.1. Parameter Evaluation

This section shows the processing results of the different input sets, based on the parameter variation. The influence of a parameter is judge by the number of correct classified samples for every input set. Finding an optimal set of all parameters for the given samples, that give the lowest overall false classification rate, is a very complex problem. The complexity of the problem is so high that it is not possible to solve the problem in a fixed time, instead heuristic methods have to be used to find an optimal solution.

Table 1: Processing Parameters

Parameters	Possible Values	Default Value
Block Width (see 4.1)	5/50/ 100/200	100
Noise Reduction Factor (see 4.2)	0/1/2/5	0
Maximum Amplitude	Yes/No	Yes
Mean Amplitude	Yes/No	Yes
Maximum Power	Yes/No	Yes
Maximum Frequency	Yes/No	Yes
Mean Power	Yes/No	Yes
Number of Peaks	Yes/No	Yes
Peak Border	1/2/5	2
Global Maximum Amplitude	Yes/No	Yes
Global Mean Amplitude	Yes/No	Yes
Global Maximum Power	Yes/No	Yes
Global Maximum Frequency	Yes/No	Yes
Global Mean Power	Yes/No	Yes
Global Number of Peaks	Yes/No	Yes
Confidence Factor	0.0001/ 0.001/ 0.01/ 0.1/ 1	0.0001

6.1.1. Default Parameters

The first calculation was performed using the default parameters (see Table 1). The following results in Table 2 were gained.

Table 2: Results for the Default Parameter Set

Correct Classified	False Classified
73.4 %	26.6 %

The numbers imply that about three quarter of the test cases are correctly classified. When looking in more detail at the classification distribution Table 3 results.

Table 3: Distribution of Wrongly Classified Samples

Sample Class	Classified as No	Classified as Small	Classified as Big
No	755	103	42
Small	175	218	57
Big	41	61	348

The majority of the samples was correctly classified. For samples with no additional weight and a big additional weigth the classification was very good, while samples with a small additional weigth were often classified as samples with no additional weigth. The results are still good, because the small attached weight was really light and sensing accuracy was not

very high. When only no additional weigth and big additional weigth samples are used then the number of wrongly classified samples goes down to 5 %.

Table 4: Results for the Default Parameter Set with no Additional small Weight Samples

Correct Classified	False Classified
92.7 %	7.3 %

Table 5: Distribution of Wrongly Classified Samples with no Small Weight Samples

Sample Class	Classified as No	Classified as Big
No	862	38
Big	60	390

6.1.2. Block Width

Table 6 shows the results, when the block width varies.

Table 6: Results for Block Width

Block Width	False Classified
5	43.3 %
50	27.4 %
100	26.6 %
200	24.3 %

The decreasing numbers imply that at some point an optimal block width can be reached, at which a minimum of false classified samples is obtained.

6.1.3. Noise Reduction

Table 7 shows the experimental results for a varying noise reduction.

Table 7: Noise Reduction

Noise Reduction	False Classified
0	26.6 %
1	24.2 %
2	27.6 %
5	42.6 %

6.1.4. Maximum Amplitude

The calculation of the maximum amplitude can be turned on or off. The Tables 8 and Table 9 show the results.

Table 8: Results for Maximum Amplitude per Block

Maximum Amplitude	False Classified
Yes	26.6 %
No	26.5 %

Table 9: Results for Global Maximum Amplitude

Global Maximum Amplitude	False Classified
Yes	26.6 %
No	26.6 %

These results show that the maximum amplitude does not have a big influence on the classification in this problem. This is even more interesting to notice, because amplitude is the value that the vibration sensors record and that can be taken as an input without additional processing.

6.1.5. Mean Amplitude

The Tables 10 and 11 show the influence of the mean amplitude values.

Table 10: Results for Mean Amplitude per Block

Mean Amplitude	False Classified
Yes	26.6 %
No	27.7 %

Table 11: Results for Global Mean Amplitude

Global Mean Amplitude	False Classified
Yes	26.6111 %
No	26.6111 %

The mean amplitude seams to have little influence on the accuracy.

6.1.6. Maximum Frequency Power

Table 12 and Table 13 show the results of the parameter variations for the maximum frequency power.

Table 12: Results for Maximum Frequency Power per block

Maximum Frequency Power	False Classified
Yes	26.6 %
No	25.0 %

Table 13: Results for Global Maximum Frequency Power

Maximum Frequency Power	False Classified
Yes	26.6 %
No	26.6 %

The maximum power seams to have a small influence on the classification.

6.1.7. Frequency with Highest Power

Table 14 and Table 15 show the results of the parameter variations for the frequency with the maximum power.

Table 14: Results for Frequency with Highest Power per Block

Frequency with Highest Power	False Classified
Yes	26.6 %
No	26.3 %

Table 15: Results for Global Frequency with Highest Power

Frequency with Highest Power	False Classified
Yes	26.6 %
No	26.6 %

The maximum power seams to have a small influence on the classification.

6.1.8. Mean Frequency Power

Table 12 and Table 13 show the influence of the parameter variations for the mean frequency power.

Table 16: Results for Mean Frequency Power per Block

Mean Frequency Power	False Classified
Yes	26.6 %
No	22.8 %

Table 17: Results for Global Mean Frequency Power

Mean Frequency Power	False Classified
Yes	26.6 %
No	26.6 %

Mean frequency power is a big factor and can improve the classification by nearly 4 %. The global mean values give no information about the condition of the fan.

6.1.9. Number of Peaks

Table 18 and Table 19 show the influence of the number of peaks for the calculation.

Table 18: Results for Number of Peaks per Block

Number of Peaks	False Classified
Yes	26.6 %
No	21.8 %

Table 19: Results for Global Number of Peaks

Number of Peaks	False Classified
Yes	26.6 %
No	26.6 %

The number of peaks does have an even bigger influence on the classification then the mean frequency power and the false classification rate can be improved by nearly 5 %.

6.1.10. Peak Border

The peak border (the value that defines what is a peak) also influences the calculation. Table 20 shows this.

Table 20: Results for Peak Border

Peak Border	False Classified
1	24.3 %
2	26.6 %
5	22.3 %

Results for the peak border show no clear trend, but the numbers suggest that an optimum exists.

6.1.11. Confidence Factor

The confidence factor determines how much the decision tree is pruned and also has an influence on the classification accuracy.

Table 21: Results for Confidence Factor

Peak Border	False Classified	Tree Size
1	27.4 %	275 Nodes
0.1	26.7 %	225 Nodes
0.01	26.2 %	185 Nodes
0.001	26.0 %	163 Nodes
0.0001	26.6 %	109 Nodes

The results of the variation of the confidence factor shows the problem of over-fitting. The less pruning is performed the more samples are wrongly classified. Over-fitting is reduced when pruning is used.

6.2. Sensor Optimization

To be able to define the set of sensors that give the most information about the fan condition a decision tree has to be generated and evaluated. To find the attributes, which contain the most information about the system, a decision tree with a low false classification rate is needed.

6.2.1. Feature Extraction

To extract significant attributes from the desicion tree a parameter set is needed that generates a tree with a low false classification rate. However it is not possible to take a parameter set of only the best attribute settings that were calculated in the experiments, because the parameters influence each other in a highly complex way. For example, if we deactivate the calculation of the mean frequency power and the number of peaks (two values which improved the false classification rate if turned off) then we get the classification results in Table 22 and Table 23.

Table 22: Results for the Modified Parameter Set

Correct Classified	False Classified
73.0 %	27.0 %

Table 23: Distribution of Wrongly Classified Samples

Sample Class	Classified as No	Classified as Small	Classified as Big
No	741	104	55
Small	163	225	62
Big	63	39	348

Table 22 shows that the false classification rate gets worse and indicates that both values are highly connected. Thus to find an optimal set of parameters special algorithms are needed, as it was mentioned in Section 6.1. Using the default parameters, which represent a good parameter set, and the sample data results in a decision tree that is partly as shown below.

```
Block109MeanPower <= 0.01
|   Block112PeakNum <= 11
|   |   Block106PeakNum <= 8
|   |   |   ...
|   |   Block106PeakNum > 8: small
|   Block112PeakNum > 11
|       Block123MeanPower <= 0.007: no
|       Block123MeanPower > 0.007: small
Block109MeanPower > 0.01
    Block110PeakNum <= 1
    |   Block11PeakNum <= 3
    |   |   ...
    |   Block11PeakNum > 3
    |       ...
    Block110PeakNum > 1
        Block111MeanPower <= 0.009
        |   ...
        Block111MeanPower > 0.009
            ...
```

Listing 3: Part of the Decision Tree

It is evident that the most important feature in the tree is the mean frequency power of block 109. Followed by other attributes like the number of peaks. The tree contains no nodes of the “global” attributes (e.g. Global Mean Power). That means that they give nearly no information for the classification problem. The block peak number and the block mean power are the most significant features or attributes for the given problem and the samples. It is interesting to notice here that the correct classification rate increases, when one of both features is not used. This indicates the complex problem of finding the optimum parameter set for a given problem.

6.2.2. Sensor Selection

Based on the feature extraction in the previous subsection it is possible to decide, which sensors and processing methods need to be considered for monitoring the system. For the experiments only two “sensors” need to be considered to gain a fairly good classification.

6.2.3. Application for the Recirculation Fans and Filters

The experiment shows good results. To validate the method and experiments two more complex tests are planned to be performed. A test on aircraft will be performed that collects real world operation samples. These samples of a well funtioning aircraft will contain no information about failure cases, however they will contain much noise from the operational environment of the aircraft. The additional noise is needed, because the algorithm will later also operate in the real world and therefore need to be well adapted to work with noise. Because it is difficult to collect information about failure cases in a test on aircraft a second test will be performed. This second test will be constructed and installed in a hangar, where recirculation fans and filters will be monitored and different failure cases will be simulated. This test set up allows gathering of samples in different failure conditions with low noise. Information processing (noise reduction, FFT) is well supported by state of the art digital signal processors and it is possible to evaluate the information nearly in real time.

7. SUMMARY AND RECOMMANDATION

The results of the experiment show that a fairly good failure classification can be made with the recommended algorithms and methods. While the classification is not perfect, it allows to decide, which features and sensors contain the most information about a system. The experiment used only

ACKNOWLEDGEMENT

This paper was supported by the State Ministry for Economic and Labour Affairs (Behörde für Wirtschaft und Arbeit) of the Free and Hanseatic City of Hamburg (Freie Hansestadt Hamburg). Support code (Förderkennzeichen): HH 132 B.

REFERENCES

[1] WEBER, Kirsten; Airbus Deutschland, EDYVVC: *Filter Clogging Indication of Recirculation Air Filters*. Confidential presentation, Feb. 2008.

[2] Inflight Warning Systems, Inc.: *Homepage*. – http://www.iwsus.net (01.07.2009)

[3] RUSSEL, Stuart; NORVIG, Peter: *Artificial Intelligence : A Modern Approach*. New Jersey : Pearson Education, 2003.

[4] QUINLAN, J. R.: *Programs for Machine Learning*. Morgan Kaufmann, 1993.

[5] QUINLAN, J. R.: *Induction of Decision Trees*. Springer Netherlands, 1986.

[6] BURRUS C. Sidney: *Fast Fourier Transforms*. Connexions online book, 2008. – http://cnx.org/content/col10550/1.18 (2009-07-01)

[7] STUMPP, Barbara: *Noch mehr Komfort für zukünftige Fluggäste*. In: Luft- und Raumfahrt. Oberhaching : Aviatic, Sonderbeilage zur Ausgabe 2, March/April 2009. – Copy at http://www.fzt.haw-hamburg.de/pers/Scholz/Aero/Aero_ART_LuftUndRaumfahrt_Kabinenprojekte_09-03-05.pdf (2009-07-01)

[8] SMITH, Michael J.: *Sensor Location & Optimization Tool Set* (Chemical Biological Information Systems Conference, Albuquerque, 2005) 2005. – URL: http://www.dtic.mil/ndia/2005st_cbis/wednesday/smith.pdf (2009-07-01)

[9] FARRAR, Charles. R.; PARK, Gyuhae; PUCKETT, Anthony D.; FLYNN, Eric B.; MASCARENAS, David L.; TODD, Michael D.: *Sensing and Sensor Optimization Issues for Structural Health Monitoring* (23rd Aerospace Testing Seminar, Manhatten Beach, 2006) 2006. – URL: http://www.lanl.gov/projects/ei/sensing/pubs/AeroTesting-06a.pdf (2009-07-01)

[10] URL: http://www.cs.waikato.ac.nz/ml/weka/ (2009-07-01)