Automotive Signal Diagnostics Using Wavelets and Machine Learning

Hong Guo[1], Jacob A. Crossman[1], Yi Lu Murphey[1], Mark Coleman[2]
[1]Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI 48128-1491

[2]Advanced Diagnostic Design
Diagnostic Service Planning
Ford Motor Company
Allen Park, MI 48101

Please address all correspondence to
Yi Lu Murphey
Department of Electrical and Computer Engineering
The University of Michigan-Dearborn
Dearborn, MI 48128-1491
yilu@umich.edu
313-593-5420(phone)
313-593-9967(fax)

Automotive Signal Diagnostics Using Wavelets and Machine Learning

Hong Guo, Jacob A. Crossman, Yi Lu Murphey, Senior Member, IEEE, Mark Coleman •

Abstract

In this paper, we describe an intelligent signal analysis system employing the wavelet transformation towards solving vehicle engine diagnosis problems. Vehicle engine diagnosis often involves multiple signal analysis. The developed system first partitions a leading signal into small segments representing physical events or stateds based on wavelet mutli-resolution analysis. Second, by applying the segmentation result of the leading signal to the other signals, the detailed properties of each segment, including inter-signal relationships, are extracted to form a feature vector. Finally a fuzzy intelligent system is used to learn diagnostic features from a training set containing feature vectors extracted from signal segments at various vehicle states. The fuzzy system applies its diagnostic knowledge to classify signals as abnormal or normal. The implementation of the system is described and experiment results are presented.

1. Introduction

Today's vehicles are becoming more and more complex with increased reliability on electronics and on-board computers. As a result, fault diagnosis on these vehicles has become increasingly challenging with a greater number of parts and controllers interacting in a large number of complex and, sometimes, poorly understood ways. Correspondingly, the job of vehicle diagnosis has become more difficult, especially for non-routine faults. Often, technicians cannot even pinpoint the root cause of a difficult fault and find themselves replacing parts in the hope that the given part is the source of the problem. This "throwing parts at the vehicle" approach increases car manufacturer warranty costs and leads to dissatisfied customers. Therefore car manufacturers are finding it necessary to develop a new breed of electronic diagnostic technology that can help lead quickly to the root cause of a vehicle fault.

During the1980s, the rapid introduction of electronic engine management techniques greatly improved the performance of the vehicle engine, while, conversely, making engine diagnosis the most difficult part of vehicle diagnosis. Vehicle diagnosis tools and techniques can be divided into three classes [7]:

1. On-board diagnostic software and self test routines. An Electronic Control Unit's (ECU) software may incorporate self-test routines that can store the fault code when a fault is detected.
2. On-board diagnostic data accessed using an off-board diagnostic tool. When a vehicle is inspected, a scanner or a scan tool can be connected to the diagnostic terminal of the on-board computer. These tools can either simply collect fault codes from the

• H. Guo, J. A. Crossman and Y. L. Murphey are with [1]Department of Electrical and Computer Engineering The University of Michigan-Dearborn, Dearborn, MI 48128-1491. Yilu@umichedu
M. Coleman is with [2]Advanced Diagnostic Design, Diagnostic Service Planning, Ford Motor Company

ECU self test, or they can record continuous signal outputs from the on-board vehicle sensors during driving.

3. Off-board diagnostic stations. These tools combine data downloaded from the vehicle ECU and sensors with off-board diagnostic sensors more sophisticated than are available on the vehicle. Again, the technician is left with the full responsibility of interpreting the data.

There are several limitations to on-board diagnostics. First, on-board software must be integrated with vehicle specific hardware, which means different vehicles cannot share the same software or diagnostic methods. Second, the error codes provided by the on-board software do not provide enough details regarding the fault to allow diagnosis. Third, the knowledge stored in the system is fixed unless the manufacturer updates it with costly replacements. Finally, because of the limited computing resources of a vehicle (slow processor and less information storage space), it's difficult to do much more than limit checking type diagnostics. Advanced signal analysis techniques such as signal transformations or machine learning techniques are not available. With the rapid development of the CPU and signal processing, off-board diagnostic techniques are more promising than on-board diagnostics. Standards are in place (ISO 9141) for the data link from the on-board computer to the off-board unit, so data can be collected from the ECU and analyzed off-line by powerful computers. Unfortunately, at this time, diagnostic techniques lag far behind data collection techniques.

Vehicle diagnosis techniques can be divided into two classes: model-based and model-free. Model-based techniques employ mathematical models of the dynamics of the vehicle components to analyze the behavior of vehicle systems [2, 10, 12]. While these models may useful for examining simplified versions of each of the engine components, we do not have accurate models for a real vehicle with many interactive components.

Model-free systems are knowledge-based, incorporating professional knowledge from engineers without exact information regarding the details of system dynamics. The rationale behind this approach is that many experienced technicians can find faults even though they do not have extensive knowledge of the mechanical or electrical dynamics of the vehicle. Examples of such system include Strategy Engine (HP), TestBench (Carnegie Group), IDEA (Fiat Research Centre) and MDS (Daimler-Benz Research) [25].

In this paper, we describe an off-board and model-free diagnostic system for identifying faulty vehicle behavior through analysis of ECU signals. The signals discussed here come from the Powertrain Control Module (PCM) of the ECU, however, the methods developed are sufficiently general to allow for use in other multi-signal fault diagnosis problems.

In a tightly coupled system such as a vehicle powertrain, inputs and outputs from every component effect most of the other components of the system. For example, the driver pressing the throttle causes an increase in the airflow to the engine. The PCM changes the control strategy modifying fuel delivery and spark timing. Increased fuel and air increases RPM, which, in turn, dramatically changes the behavior of the transmission and other components such as alternator output. Furthermore, there are feedback loops in the system. The on-board controller monitors output exhaust quality, gear changes and airflow changes further modifying system behavior to keep performance at a maximum and exhaust pollution at a minimum. Outside factors such as road quality, road gradient, vehicle weight, active accessories, etc provide physical feedback to the system further altering behavior.

In our system, we capture these physical events and dependencies through the powertrain signals. For example, figure 1(a) demonstrates a simplification of the relationship between the TP (Throttle Position) and RPM (Revolution Per Minute) signals. TP makes a sudden rise and fall while RPM mimics this behavior but more smoothly. This simplification is not completely accurate but demonstrates the key point that important physical relationships can be seen through the vehicle signals. Figure 1(b) shows a more typical set of relationships between four different signals. Each circle is a signal and each edge indicates a feature that the tail signal influences in the head signal. These relationships are often complex, include 5-10 different important signals, and have many cyclic dependencies between signals.

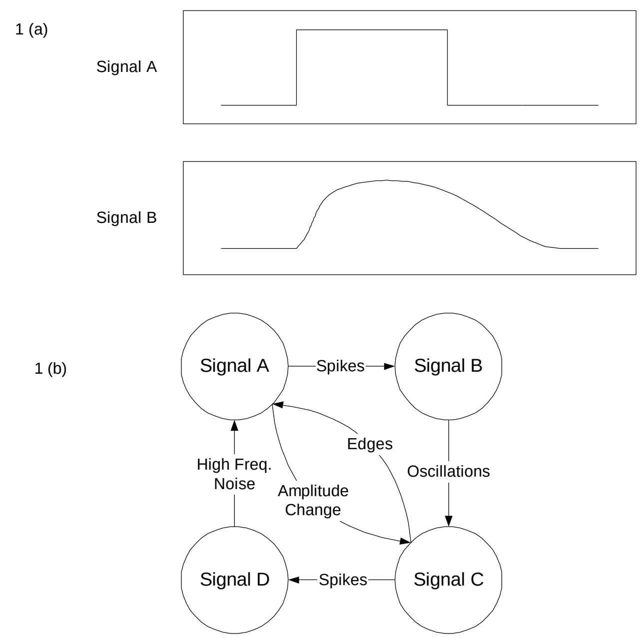


Figure 1: In multiple signal systems, changes in one signal generally result in changes in one or more other signals. In (a) signal B is the smoothed version of signal A, a simple relationship. In (b) each edge is indicating the feature the signal at the tail end is causing in the signal at the head end. Notice that each signal effects and is effected by multiple different signals.

We note several important issues related to using signals to diagnose a vehicle. First, we must differentiate between a bad signal and bad vehicle behavior reflected in the signal. A bad signal is generally caused by a bad PCM or a bad sensor. Bad vehicle behavior can be caused by any of a number of factors, physical or electronic. Our system detects signal features that indicate bad vehicle behavior, whether it is caused by bad electronic parts or physical faults. Second, we note that not all of the physical dependencies present in the actual vehicle can be modeled with corresponding signals. For instance, there is no signal to indicate road bumpiness, a physical factor that can effect vehicle and, therefore, signal behavior. To handle these unknown conditions we train with vehicle data in several conditions while avoiding extreme driving conditions (e.g. off-road racing). Finally, the same signals are not available from all vehicles. When considering behavior that depends on signal relationships, this can lead to an inability diagnose certain faults that depend on information present in the missing signal.

In this paper we focus on developing techniques of decomposing multiple signals, diagnostic feature extraction, and intelligent diagnosis. The paper is organized as follows. In section 2 we briefly introduce the diagnostic system. In section 3, an automatic segmentation algorithm based on wavelet multi-resolution analysis is introduced. In section 4 we discuss how we can process and combine feature and segment information to form feature vectors suitable for input into a machine learning system. Section 5 describes how a fuzzy-based machine learning system can be used to learn good and bad signal behavior. Section 6 describes the implemented diagnostic system and the encouraging experimental results we've obtained. Finally, section 7 discusses the impact of our work thus far and our future goals for this research.

2. System overview

The system we have developed is a multi-layered diagnostic system (see Figure 2). Here we present a brief overview of each layer and its goals. Layers 2, 3 and 4 are discussed in more detail in later sections. Layers 1 and 5 are discussed only briefly.

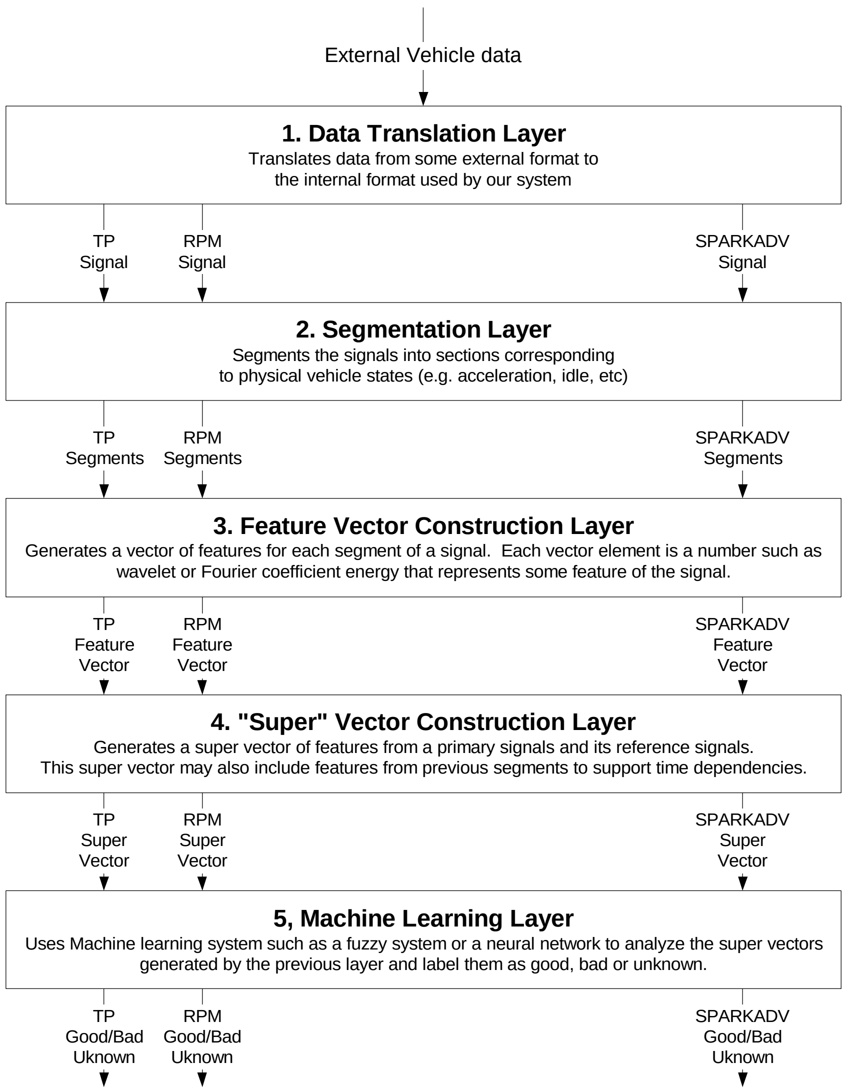


Figure 2: Diagram of the proposed diagnostic system. Each layer handles different signals individually thus enabling the diagnosis of each signal to be customized.

The first layer translates the data into a format suitable for processing. This layer is relatively simple and is not discussed further.

The second layer automatically partitions the signal into segments using either wavelet features from that signal or the segments of another signal. Figure 3 shows a TP signal that has been segmented using this module. These segments have three purposes. First, they divide the signal into regions that relate to some physical vehicle state, for example, acceleration or idle. If we know the general physical state of the vehicle we can eliminate many possible faults and behaviors that we know cannot occur in the given physical state. Second, segmentation leads to a natural clustering of the signal data. Signal behavior within a given segment is generally very similar to behavior in other segments of the same state. This leads to more consistent training and test data. Third, using the segments we can isolate fault location within a signal. This can lead to easier fault identification. Finally, the segmentation strikes a nice balance between analyzing the original signal as a whole, which would result in enormous amounts of superfluous data, and analyzing the signal in one piece, which would result in a very complex feature vector. Segmentation allows us to examine important details of the signals without overloading the system with data.

The third layer extracts features from each segment and combines these features into a feature vector compatible with the machine learning system. These feature vectors

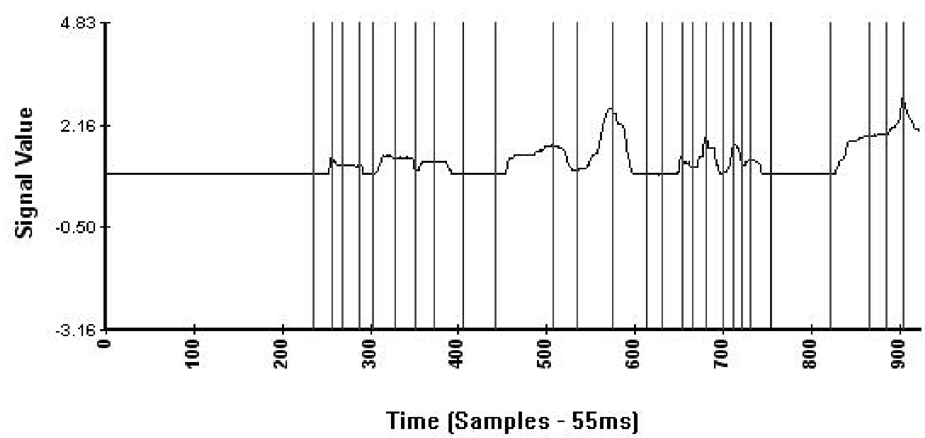


Figure 3: An example of segmentation on the Throttle Position (TP) signal. In ADSAS, the segment lines are color-coded to indicate the beginning of acceleration (rising), cruise (steady high), deceleration (falling) and idle (steady low) segments. The segmentation was done automatically by our system, ADSAS (see Experiments).

highlight important aspects of the segments that can be used to define good and bad behavior. For example, we may want to look at the “noise” in the signal, the rate of change within a segment or the motion pattern. To extract this information in a compact form we use advanced signal processing techniques including wavelet and Fourier transforms as well as statistical data regarding the signal itself. From the transformation coefficients and the statistical data we choose, for each signal, the data elements that best represent the given signal’s features. Finally, we compose our feature vector from this data. Figure 4 shows examples of feature vectors extracted from the TP signal shown in Figure 3.

Seg#	Start	End	State	FE [4.00; 20.0]	Minimum	Maximum	Average	WE_DB1 L1	WE_DB1 L2
1	0	236	1	8.69	0.849	0.868	0.860	8.92e-006	1.18e-005
2	236	256	2	7.07	0.854	1.25	0.895	1.40e-003	8.34e-003
3	256	269	4	10.6	1.09	1.26	1.15	2.22e-004	9.83e-004
4	269	288	3	7.30	1.07	1.09	1.08	4.47e-006	6.81e-004
5	288	302	4	10.1	0.855	1.07	0.920	1.21e-003	4.14e-003

Figure 4: Example of a few of the features extracted from the TP signal shown in figure 3. Start and end indicate the beginning and ending sample of the segment. States are 0-3 indicating idle, acceleration, cruise and deceleration respectively. F: [4.00; 20.0] is the energy of the Fourier coefficients over the frequency range 4-20 Hz for the given segment. Minimum, maximum and average are calculated on the original signal. The WE_DB1 Ln columns are the energy of the DB1 (i.e. Haar) wavelet detail coefficients at the nth level over the given segment. (Taken from Adsas – see Experiments)

The fourth layer introduces the notion of time and signal dependency into the system. In this layer, we choose a primary signal for analysis and select a set of reference signals that have some causal relationship with a given primary signal. We then combine the features of our primary signal with selected features from the reference signals to form a “super feature vector.” This super feature vector may also include features from previous segments, thus incorporating time dependency into the system. For example, as mentioned earlier, an RPM rise is generally caused by a similar rise in TP. Furthermore, because of physical inertia, behavior of RPM in one segment is usually very closely tied to the behavior in the immediately previous segment. Thus we create a super feature vector for RPM that contains RPM’s features for the current and previous segment along with the state of the TP signal (increasing -acceleration, decreasing - deceleration, stable – cruise/idle, etc). A machine learning system can use this information to differentiate between RPM signals that react normally to TP and those that do not.

The fifth and final layer consists of a machine learning system. The machine learning system is trained to recognize faults from one type of signal at a time, which results in separate knowledge bases for each signal type. Currently, we use a fuzzy learning system, but this layer can be generalized to neural networks or other suitable machine learning systems. Each of the super vectors produced in layer 4 is fed into the learning system for training. At this time the training is supervised so we provide a target output that the system attempts to match. We note that, as with many complex diagnosis problems, we often don’t have many verified “bad” data samples so we mainly train with good samples.

The following three sections discuss each of the layers 2-4 in more detail.

3. Signal Segmentation

The signal segmentation algorithm we developed partitions a signal into time slices representing different vehicle states. The vehicle states we consider are idle, cruise, acceleration and deceleration. The TP signal in particular is a good candidate for segmentation because its behavior closely mimics these four states of the vehicle. In the TP signal, signal rises occur during periods of acceleration, declines occur during periods of deceleration and the relatively flat TP signal indicates cruise or idle. From TP we can make a good estimate of the vehicle's state at any given time and partition the signal into segments representing time periods when the vehicle state is consistent. Figure 3 above (section 2) shows an example of a typical TP signal segmented using our segmentation algorithm.

Using TP as the leading signal, we apply the segments obtained from automatically segmenting the TP signal to other signals from the same vehicle recording in order to label the corresponding vehicle states in those signals. In Figure 5 we show the RPM and SPARKADV (Spark Advance) signals overlaid with the TP segments from figure 3, demonstrating how the TP segments map to other signals. The issue then becomes how to best segment the TP signal. We propose an approach based on multi-resolution analysis (MRA) below that uses wavelet coefficients to help find segment boundaries.

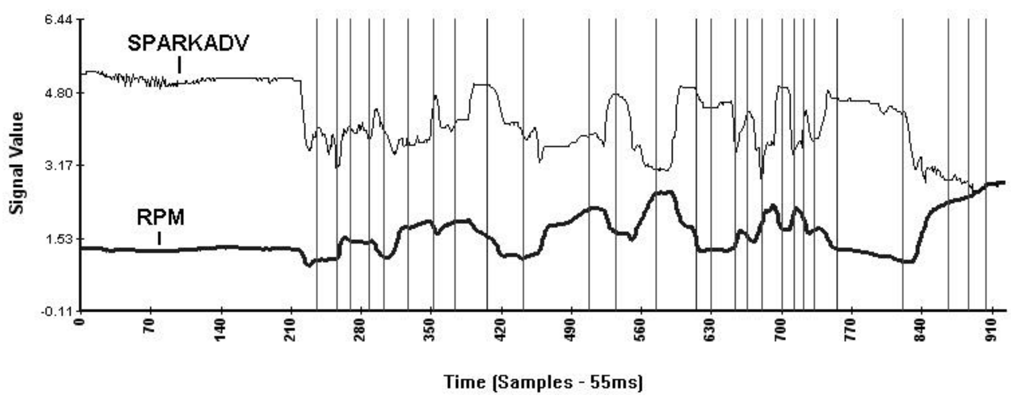


Figure 5: Segments from the TP signal in figure 3 applied to the RPM and SPARKADV signals.

3.1 Using Wavelets and Multi-resolution Analysis in Signal Segmentation

Our automatic segmentation algorithm is based on MRA and uses the discrete wavelet transform (DWT) [9, 18, 19] to isolate features in multiple scales. The DWT has been used recently for many other applications including image compression [23], pattern recognition [4], speech processing [13], signal detection [17], astronomy [3] and model estimation [5].

Finding segment boundaries is a type of edge detection problem. In particular, the states *acceleration* and *deceleration* correspond to rising and falling edges in the TP signal respectively. With the correct choice of mother wavelet function, the wavelet coefficient values associated with a signal can be used to identify these edges. [18].

Furthermore, the wavelet coefficients tend to isolate signal features in (such as edges) by scale. This allows us to tune the segmentation to avoid certain edges occurring from random noise or very smooth changes.

We implement the DWT using the FWT (Fast Wavelet Transformation) algorithm [18, 19]. The first stage of FWT starts with the original signal. By passing the signal through special low and high pass filters associated with the mother wavelet function, we obtain detail wavelet coefficients corresponding to the high frequency details of the signal and approximation coefficients corresponding to the signal minus the details. These two sets of coefficients are then downsampled by 2 before the algorithm continues by repeating the process on the approximation coefficients as we show in figure 6 below.

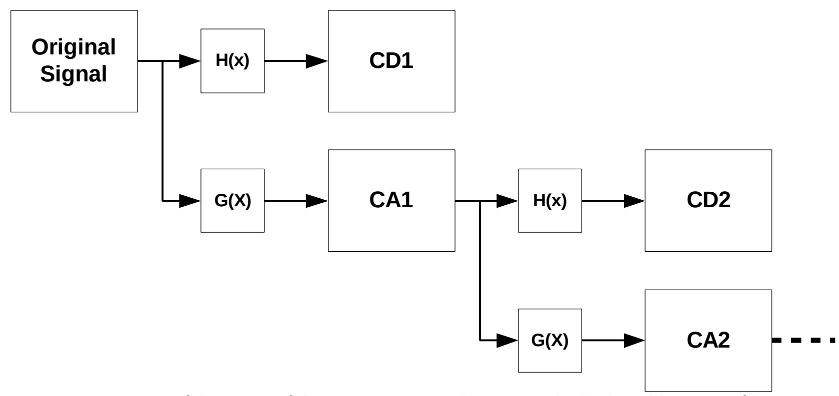


Figure 6: Diagram of the steps of the FWT. H(x) and G(x) are the high and low pass filters respectively associated with the chosen mother wavelet funciton. CDn are the detail coefficients at level n and CAn are the approximation coefficients at level n.

We label the detail coefficients as CD_k where k is the level of decomposition (number of iterations of the filtering and downsampling steps). The approximation coefficients are similarly labeled as CA_k. As k becomes larger, the corresponding detail coefficients, CD_k indicate coarser (larger scale) details in the signal. High frequency details can be found in the fine detail levels (e.g. 1-3), while low frequency details can be found in the courser detail levels (e.g. 4-6). Specifically what the detail coefficients indicate is dependent on the mother wavelet function. In our case we have chosen to use the mother wavelet *DB1* (Daubauchies One) because its detail coefficients indicate sharp changes in a signal indicating transition states (*acceleration* or *deceleration*). Specifically, using *DB1* sharp changes correspond to peaks and valleys in the detail coefficients obtained from the FWT.

3.2 Segmentation algorithm

The segmentation has four major steps we describe below. Details of each of these steps follows.

1. *Create approximate segment boundaries.* This step uses the wavelet coefficients from a range, [1, K], of detail coefficient levels to place segment boundaries very close to the correct location in the signal. We use a recursive,

multi-scale algorithm to divide large segments into multiple smaller segments for further detailed analysis.

2. *Combine same state segments*: After step 1, some adjacent segments may have the same state and this step joins the segments together.
3. *Fine tune segment boundaries*: This step looks at a small neighborhood, typically one or two samples, around the segment boundaries and shifts the boundaries to more optimal locations. This step also removes any steady states (*idle* or *cruise*) that are too short to be significant.
4. *Combine same state segments again*: The process in step (3) may create adjacent same state segments, so we re-run step (2).

The first step uses a recursive procedure to examine increasingly finer levels of detail in the signal. Segments begin as large, inexact boundaries and are fine-tuned as finer levels of detail coefficients are examined. We have found that if we ignore the coarse level coefficients, we cannot identify certain smooth changes in the signal. On the other hand, if we ignore the finer level coefficients, we miss small changes. The following parameters are used to control the segmentation results.

(1) b_1, the zero threshold. This is used to determine if the absolute value of a CD_1 coefficient is small enough to be called 0. $b_1 > 0$. The zero threshold at level k, b_k, is calculated using the equation $b_k = b_1 + (k-1) * b_1 * l$, where l is a constant that accounts for the natural scaling of the coefficients across different levels. Thresholds that are too high result in some steady states with high fluctuation, while thresholds that are too low result in many, tiny segments.

(2) l, the coefficient level-biasing factor (see (1)). If $l > 1$ then smooth transitions in the signal are more likely to be labeled as STEADY (e.g coarser coefficients are more likely to be ignored). If $l < 1$ smooth transitions are more likely to be labeled as transition states, such as ACCEL or DECEL (e.g. coarser level coefficients have greater impact).

(3) h, the boundary pinpointing neighborhood. This is the number of the neighboring coefficients that are searched during step 3 of the algorithm. This parameter has the greatest impact on very short and sharp transition states. If it is too high the boundary may jump out of a local maximum or minimum and result in overlapping segments.

(4) m_0, the length of acceptable steady states. This parameter primarily impacts the amount of padding added to transition states. Generally, we set this to be about 1 second in the time domain.

(5) L_S and L_E, the starting coefficient detail level and ending coefficient detail level respectively. We have set $L_E = 0$, and $L_s > L_E$. If L_S is too low, we may miss some smooth transition information. If L_S is too high, we get too much padding on transition states while other aspects of the segmentation do not improve.

Each recursive instantiation of the algorithm focuses on one section of the signal represented as the indexes, [*begin*, *end*), into the signal's detail wavelet coefficients at a given level. The assumed state of the segment when the algorithm begins, is also passed into the algorithm.

The algorithm iterates through every coefficient at the given detail level, tracking consecutive sequences of 0s (determined using b_k), positive values or negative values. These sequences represent the states cruise, deceleration, and acceleration respectively if the mother wavelet function used is DB1. We use STEADY, DECEL and ACCEL to denote the three states, respectively. Note that we do not differentiate between idle and cruise, calling them both STEADY. Idle states are determined later using information from another signal.

Every time the value of the coefficients changes between 0, positive, and negative, we have a state change and we locate the end point of the previous state. The endpoint is found using the location and state of a segment to optimize the endpoint location according to the following rule: pad transition states and shrink cruise states. This rule is based on the fact that TP transitions tend to indicate the beginning of a vehicle state change. There is always some time delay between when TP changes and the vehicle reacts to this change. The padding at the end of transition states leaves some room for the reaction time of the vehicle.

Steady states are handled slightly differently than both acceleration and deceleration states. If the value of a coefficient is 0, we increment a counter. If the number of consecutive 0s is over a given limit, the state is considered STEADY. This eliminates short cruise segments (e.g. "steps") and further pads transition states.

Once we have identified a state's begin and end boundaries, we recursively call the *Segment* procedure to further sub-divide the segment as necessary. This is necessary because the wavelet transform tends to isolate features of different scales at different levels of the detail coefficients. It is not uncommon for a transition state to appear in one level of detail coefficients and be totally absent in another. Moving to finer levels of coefficients guarantees we find all significant vehicle states. Furthermore, we can more precisely locate segment boundaries.

The base case of the recursion occurs at level L_E, here L_E=0. It indicates that we are operating on the original signal instead of the coefficients. Here we simply create the segment. Because we use depth-first recursion, the segments are created in order, from the first signal sample to the last.

The recursive partitioning of the *Segment* procedure will sometimes segment one large state into a few smaller states. Therefore, we run a simple algorithm to combine adjacent same-state segments after this procedure is over. Lastly we need to fine tune segment boundaries. We use the original signal to check the neighborhood, usually two samples (h =2) around each segment boundary and move them slightly according to these rules:

1. Move the beginning of accelerations and the end of decelerations to the local minimum farthest to the right.
2. Move the end of accelerations and the beginning of decelerations to the local maximum farthest to the right.

When we finish this process we may have produced steady states that are too short, so we eliminate those based on m_0. We also must remove all adjacent same state boundaries again in case they were created when we removed a steady state.

We have found that the parameter values, L_S=3 or 4, L_E=0, b_1=0.015, l =0.60, h =2 And m_0=20 give good results most of the time. However, it is difficult to optimize the parameters because “good,” in this case, is subjective. One example was shown in figure 3. For signals with relatively little noise (small fluctuations) we get nearly 100% accuracy. When the signals are noisier, we get 98% accuracy. What we desire is correctly positioned segments that don’t include behavior that should occur in other states, e.g. acceleration states with no significant decreases. Currently, we manually examine the segments before using them in training. Any segments that are poorly located are removed from the training set.

4. Feature Extraction

The goal of this module is to extract significant diagnostic features from the segments obtained in the segmentation module. We define a feature to be any property of the signal within a segment that is useful for describing normal or abnormal signal behavior within the given segment.

Based on vehicle diagnostic engineering and signal processing knowledge, we have found the following features to be useful:

1. *Segment state*. Either idle (0), acceleration (1), cruise (2) or deceleration (3).
2. *Segment Length*. Generally, signal behavior within a segment is dependent on the length of the segment. For example, an long acceleration state will likely result in a much greater change in RPM than a very short one.
3. *Signal Max and Min.* The maximum and minimum values of the signal within a segment. These features are valuable for detecting out of bounds conditions such as TP dropping below its idle threshold.
4. *Signal Range*. The range of a signal in a segment is defined as y_{max}-y_{min} where y_{max} and y_{min} are the maximum and minimum values of the signal within the segment.
5. *CD_k Energy*. The energy of the CD_k wavelet coefficients within a segment. It is calculated using:

$$e = \frac{\sum_i y_i^2}{L}$$

   where L is the length of the segment, y_i is the CD_k coefficient, and e is the energy. The energy reflects the average noise level of the signal in the segment. It is especially useful to detect faults indicated by noise.
6. *CD_k Average*. The average value of CD_k wavelet coefficients within a segment. This is similar to (5) above; however, it retains sign information and can be useful for indicating the direction of motion within a segment.

The features listed above pertain to a single signal. As we have shown earlier, a single signal from a vehicle is often not sufficient for detecting abnormalities. Therefore, we may also include features that are combinations of features from other signals to form a more complete feature vector. Typical ways of combining the features include:

a. S*imple copy*: Copy a feature from another signal within the same segment.
b. *Difference*: Take the difference between a feature of one signal from another signal within that segment.
c. *Ratio*: Take the ratio of a feature from two different signals within the segment.

In our experiments with TP and TPCT (see section 6) we chose to use the feature $TP_{min} – TPCT_{max}$ as a feature because certain values of this distance indicate a bad TP signal. Even additional signal features may not be sufficient to diagnose some types of faults. Many signal such as RPM are effected greatly by the vehicle inertia and the values of features in a segment may depend on the behavior in previous segments. Therefore we will sometimes place features from previous segments into a feature vector to account as a time delay factor. We use this approach in our TP/RPM experiments when we placed the previous state and length of previous segment into our feature vector.

Each signal has a different set of features to account for their different behavioral patterns, signal dependencies, time delays, and common faults. Choice of which features to use is based partly on engineering knowledge of vehicle engineers and partly on experimentation. We begin by examining typical behaviors of a signal with an expert who describes the significance of specific features within the signal, dependencies for the signal, and typical fault cases. From this expert information, we create a feature vector from the set of possible features listed above, that we believe to be the best for capturing the information explained by the vehicle expert. Finally, we run experiments using the features and analyze the data for consistency and ability to highlight common faults. If necessary, we modify the feature vector based on experiment results or additional expert information and continue until satisfactory results (95-100% identification of bad segments) are obtained.

5. Signal Fault Diagnosis Using Fuzzy Logic

The theory of fuzzy logic is aimed at the development of a set of concepts and techniques for dealing with sources of uncertainty, imprecision and incompleteness. Fuzzy systems have been used successful in many applications including control theory where gradual adjustments are necessary, control systems, business and even the stock exchanges [1, 20, 22, 24, 21, 14, 6, 11,16]. The nature of fuzzy rules and the relationship between fuzzy sets of differing shapes provides a powerful capability for incrementally modeling a system whose complexity makes traditional expert systems, mathematical models, and statistical approaches very difficult. The most challenging problem in signal diagnostic problem is that knowledge of most signal faults is incomplete and vague due to the complexity of modern vehicles. This uncertainty leads us to seek a solution using fuzzy diagnostic methods.

Fuzzy reasoning is performed within the context of a fuzzy system model, which consists of control, solution variables, fuzzy sets, proposition (rule) statements, and the underlying control mechanisms that tie all these together into a cohesive reasoning environment. The fuzzy rules can be completely characterized by a set of control variables, $X = \{x_1, x_2, ..., x_n\}$ and solution variables, $y_1, y_2, \ldots, y_k$. In this application, we have one solution variable y, which represents the fault status of the current signal segment, i.e. a value of GOOD indicates a normal segment while a value of BAD or UNKNOWN indicates a faulty or suspect segment. The control variables are the features from segment feature vectors as described in section 4. Each control variable x_i is associated with a set of fuzzy terms $\Sigma_i = \{a_i^1, \ldots a_i^{p_i}\}$, and the solution variable has its own fuzzy terms $\Gamma = \{t_1, ..., t_q\}$. For example Γ = {normal, likely normal, not likely normal, abnormal}. Each fuzzy variable, control or solution variable, is associated with a set of fuzzy membership functions corresponding to the fuzzy terms of the variable. A fuzzy membership function of a control variable can be interpreted as a control surface that responds to a set of expected data points. The fuzzy membership functions associated with a fuzzy variable can be collectively defined by a set of **critical parameters** that uniquely describe the characteristics of the membership functions, and the characteristic of an inference engine is largely affected by these critical parameters. The possible values of these critical parameters form a hyper space and the system response to the control parameters form a control surface. The optimization of fuzzy membership functions requires finding a point in the hyper space of the critical parameters that makes the control surface (the system response) react correctly to a set of training data. A fuzzy rule k has the following format:

IF (x_{k1} is a_i^{k1}) AND (x_{k2} is a_i^{k2}) AND ...(x_{km} is a_i^{km}), **THEN** *y* is t_i^k

where $\{x_{k1}, x_{k2}, ..., x_{km}\} \subset X$, $\{a_i^{k1}, a_i^{k2}, ..., a_i^{km}\} \subset \Sigma_i$, and $t_i^k \in \Gamma$.

Fuzzy rule generation and membership functions are critical to the performance of a fuzzy system.

In this paper, we present a fuzzy intelligent system that can automatically learn new knowledge from either data or engineering experts. Figure 7 gives an overview of the fuzzy intelligent system. Figure 7 (a) shows the fuzzy learning component and (b) shows the fuzzy inference component.

The fuzzy learning component has functions to acquire knowledge either from engineering experts or from machine learning through training data. The output of the fuzzy learning component is a fuzzy knowledge base that is composed of the fuzzy rules and the fuzzy membership functions (MSFs). Signal diagnostic knowledge can be generated either from engineering experts or training data by using a machine learning algorithm.

The fuzzy inference component has the ability to apply signal diagnostic knowledge to signal segment and derive the diagnostic result. The most challenging portion of the research is generating an effective knowledge base.

If we assume each fuzzy variable has the same number of fuzzy terms, a complete set of fuzzy rules contains m^n fuzzy rules, where *n* is the number of fuzzy variables and *m* is the number of fuzzy terms for each variable. When *m* and *n* become large fuzzy rule generation can be computationally expensive [1, 14, 20, 22]. The fuzzy rule generation algorithm we used for fuzzy learning efficiently generates a compact and optimal set of fuzzy rules. In this algorithm, fuzzy rule generation is combined with fuzzy membership optimization. At the beginning of the algorithm, the critical parameters of the fuzzy membership functions are initialized at arbitrary points, and are further optimized during and after the generation of the fuzzy rules. The fuzzy rules and the membership functions are generated and optimized in an iterative fashion. The details of the fuzzy rule generation can be found in [LCH98]. The fuzzy rules generated by the algorithm are a compact subset of the complete fuzzy rule set, which is determined by the control and solution variables and their associated fuzzy terms. Since the fuzzy rule pruning process eliminates unreliable fuzzy rules, and the rule merging process combines several rules into one, the resulting compact fuzzy rule knowledge base allow the system to perform robust and efficient detection. However, during fuzzy inference, it is possible that an input data sample fires no rule in the knowledge base. In order to deal with this problem, we developed an inference scheme that fires the nearest rule to the input sample.

Generally, a fuzzy rule can be considered as a fuzzy cluster in input space. For example, a fuzzy rule written as

if $\boldsymbol{x_1}$ *is* ***LOW*** *and* $\boldsymbol{x_2}$ *is* ***HIGH***, *then* **y** *is* ***MEDIUM***

represents a fuzzy cluster centered at the ***center point*** of the fuzzy membership functions of "$\boldsymbol{x_1}$ is **LOW**" and "$\boldsymbol{x_2}$ is **HIGH**". Based on this concept, we define the following distance measure between a data sample and a fuzzy rule. Let an input data sample be I = {a_1, …, a_n}, where a_i is the instantaneous value of fuzzy variable x_i, i = 1, …n, $\Sigma = \{a_1, a_2, \ldots, a_p, f\}$ is a set of fuzzy terms associated with each control variable in X, f is a symbol that serves as "*don't care*". With the introduction of f, a fuzzy rule can always be written in the following general form:

if $\boldsymbol{x_1}$ is a_{1k} and $\boldsymbol{x_2}$ is a_{2k} and…and x_n is a_{nk} then **z** is b

where $a_{ik} \in \Sigma$, for i = 1, …, n, z is a solution variable and b is a fuzzy term. For example, for a system has three control variables, {x_1, x_2, x_3}, and the fuzzy terms are {LOW, MEDIUM, HIGH} if we have a fuzzy rule:

if $\boldsymbol{x_1}$ *is* ***LOW*** *and* $\boldsymbol{x_3}$ *is* ***HIGH***, *then* **y** *is* ***MEDIUM,***

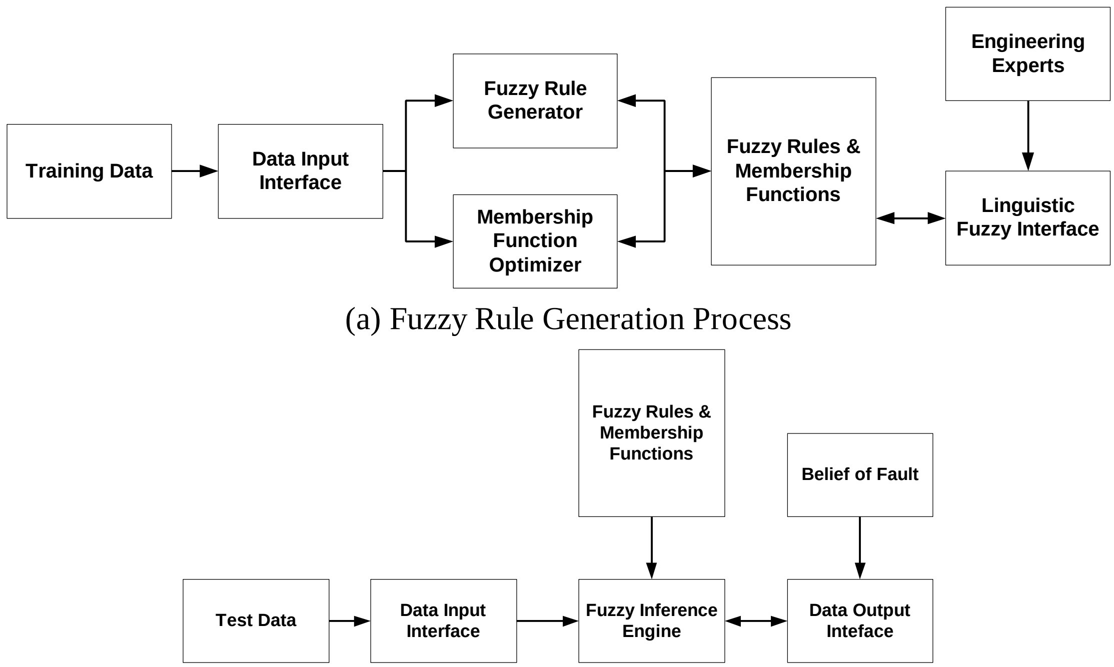
we can rewrite the rule equivalently as

if $\boldsymbol{x_1}$ *is* ***LOW*** *and* $\boldsymbol{x_2}$ *is* f *and* $\boldsymbol{x_3}$ *is* ***HIGH***, *then* **y** *is* ***MEDIUM***

The *distance* between an input vector $a = \langle a_1, a_2, \ldots, a_n \rangle$ and a fuzzy rule k is defined as

$$d = \sqrt{\sum_{i=1}^{n} (a_i - c_{ik})^2},$$

where c_{ik} is the center point of the fuzzy membership function of fuzzy variable x_i for fuzzy term $a_{ik} \neq f$, otherwise $c_{ik} = a_i$, which sets $(a_i - c_{ik}) = 0$. The fuzzy rule that has the shortest distance to the input data a is fired.



(a) Fuzzy Rule Generation Process

(b) Fuzzy Inference Process

Figure 7: A high-level overview of the fuzzy learning process (a) and the fuzzy inference process (b).

6. Implementation and Experiments

The algorithms described in the previous sections have been integrated into a single system, ADSAS, and a number of experiments have been conducted using this system. Section 6.1 describes the functions of ADSAS and section 6.2 describes two experiments in vehicle signal diagnosis.

6.1 Advanced Diagnostic Signal Analysis System

Advanced Diagnostic Signal Analysis System (ADSAS), is an intelligent system for vehicle diagnostics and learning. The system is implemented using the Win32 API and is compatible with Windows 95/98/NT. The system has two goals. First, it provides a powerful and flexible testbed for our research and experimentation. Second, it is a foundational component of a larger system used for vehicle diagnostics by engineers and technicians in a major auto company. We present screenshots of the system in Figures 8-10. Figure 8 is a general screenshot of the system's primary windows. Figure 9 shows a close-up view of a signal including wavelet transform coefficients and segments. Figure 10 shows the fuzzy intelligence system.

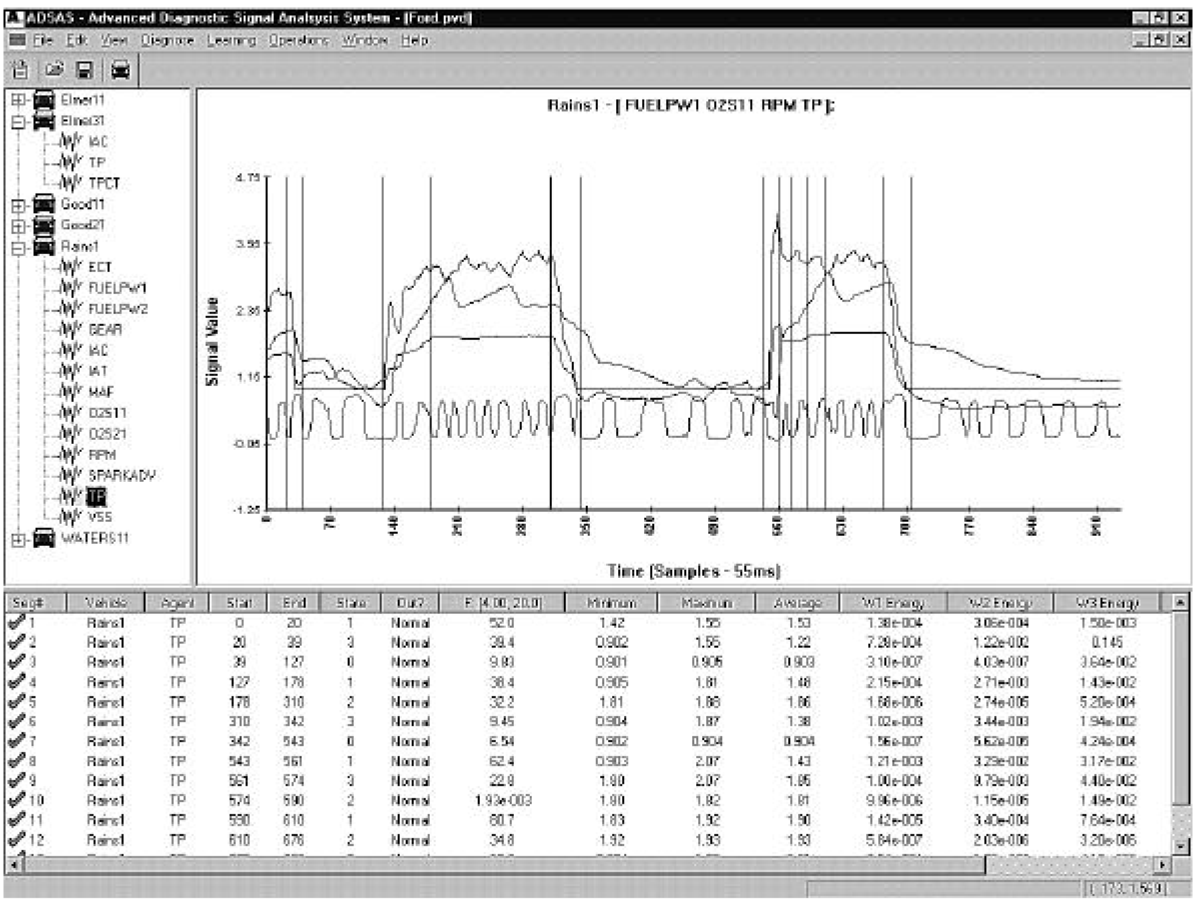


Figure 8. Screenshot of ADSAS our current prototype diagnostic system. There are three panes. The left pane is form managing the vehicle and signal data. The right pane plots signals and transformation coefficients. The bottom pane displays data collected from the signal segments.

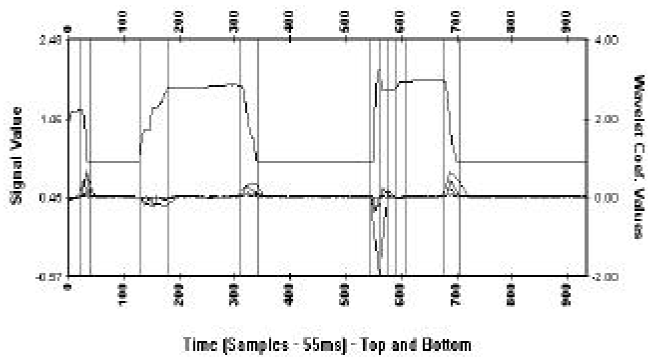


Figure 9: Closeup of plot pane showing wavelet coefficients plotted below the signal. The wavelet axis is to the upper right of the plot.

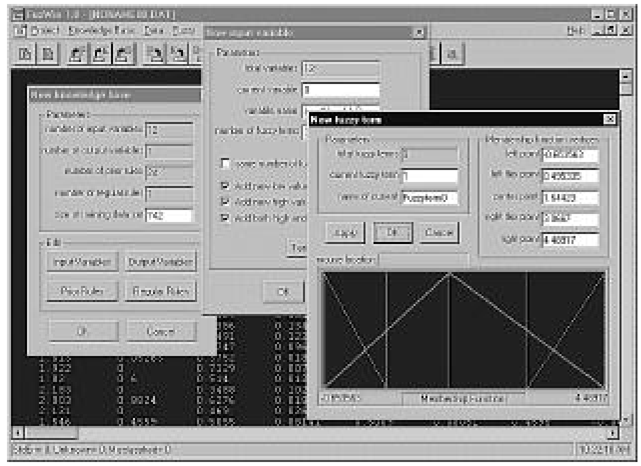


Figure 10: The fuzzy learning program used for machine training and testing as described in section 5.

In addition, ADSAS contains a number of functions supporting signal diagnosis.

- functions to load, view, and process any number of vehicle recordings and signals simultaneously.
- extensive plotting capabilities including:
    a) Multiple concurrent signal viewing
    b) Zoom out an in to within one sample
    c) Simultaneous wavelet and Fourier coefficient display using a second pair of axis
    d) Display of signals segments
    e) Ability to save plot to file
- a module that automatically segments signals as described in Section 3.
- wavelet transformation functions
- fast Fourier transformation functions
- a fully integrated fuzzy intelligent system.

The system architecture, design and implementation is object oriented (OO) with extensive reliance on data abstraction, encapsulation and polymorphism to make each component more flexible. Our system can easily add new functions, learning techniques, learning parameters, data views, file converters or other components with a minimum of re-coding.

6.2 Experimental Results

In this section we describe two groups of multiple signal diagnosis experiments performed using ADSAS. The task of ADSAS is to flag all signal segments that are abnormal according to knowledge learnt by the fuzzy intelligent system described in section 5 above.

The first group of signal diagnosis experiments is aimed at detecting faults in the TP signal using both TP and Throttle Position Closed Throttle (TPCT) signals. The second group of experiments involves the detection of RPM stumbling using RPM and TP.

6.2.1 Detection of faults in TP using TP and TPCT

The TP signal indicates the openness of the vehicle throttle. It is recorded as a voltage level typically between about 0.8-1.2V for idle and 5V-6V for wide-open throttle. The TPCT signal is the baseline signal for TP and indicates the idling voltage. TPCT is set by the on-board computer every time the vehicle is started, which means the value for TPCT is calculated by the PCM is equivalent to the minimum value of TP.

The following is the typical behavior of TP and TPCT for each of the four vehicle states:

1. Idle: TP = TPCT soon after engine start. TP is very flat.
2. Acceleration: TP > TPCT. TP rises sharply or in steps.
3. Cruise: TP > TPCT. TP is either flat or is gradually sloping.
4. Deceleration: TP > TPCT. TP drops sharply, possibly with steps.

We were given the signals showing one typical fault that can be detected using the TP and TPCT signals. In this fault case, the throttle becomes shut too far near idle causing TPCT to be adjusted to a value that is too low. When the throttle springs back to its normal idle position, the PCM thinks the vehicle is at partial throttle and dumps fuel into the engine causing rough idle. We show a plot of such a case below:

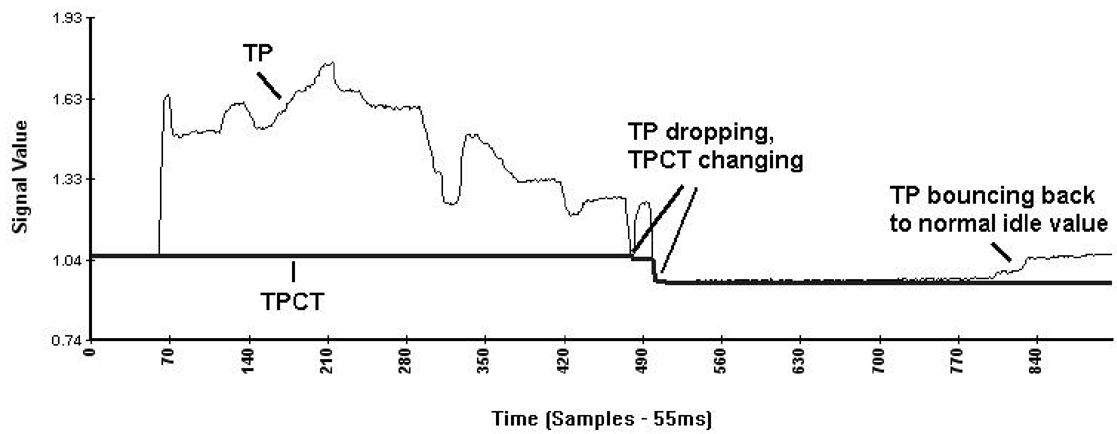


Figure 11. Typical example of TP closing too far, causing TPCT to change. When the TP moves back to its normal position, the PCM thinks the vehicle is accelerating and dumps too much fuel into the cylinder, causing rough idling.

We performed two tests using 6 recordings containing the fault and 6 recordings of normal TP behavior. In the first test, we used only CRUISE and IDLE segments because these are the segments where most of the faulty behavior is observed. Table 1 shows the fuzzy variables used and the number of fuzzy terms associated with each fuzzy variable.

The diagnostic feature used in this problem is an 8-dimensional vector, $x = <x_1, x_2, \ldots, x_n>$, with each element representing the following features:

x_1: CurState: State of the current segment.
x_2: Length: The length of the current segment.
x_3: TP_Range: The range of TP signal in the current segment.
x_4: TP_Min: The minimum value of TP in the current segment.
x_5: TPCT_Range: The range of TPCT signal in the current segment.
x_6: TPCP_Min: The minimum value of TPCT in the current segment.
x_7: TPCP_Max: The maximum value of TPCT in the current segment.
x_8: TP_Min-TPCT_Max: The difference between TP_Min and TPCT_Max .

For this experiment, we have only one solution variable, y, which represents the fault status of a segment. GOOD (low value) represents a normal segment, while a value of BAD (high value) or UNKNOWN indicates an abnormal segment.

Table 1. The number of fuzzy terms associated with each variable for the diagnosis of TP and TPCT using data of two vehicle states only.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
No of fuzzy terms	4	3	6	4	6	4	4	6

Test results from two experiments are shown in Table 2. Experiment 1 differs from experiment 2 in how the training and test sets were created.

Experiment #1: All abnormal vehicle segments were placed in the test set. The remaining normal segments from normal and abnormal vehicles were randomly chosen for the training set and the rest of the test set.
Experiment #2: Data for the test and training sets were randomly chosen from the set of all segments, both abnormal and normal.

We generated a pool of 53 normal segments from the signals shown above and 11 abnormal segments. The columns for the test results are interpreted as follows:

1. Experiment Number
2. Number of data in the training set
3. Number of training data mis-classified
4. Number of fuzzy rules generated
5. Number of data in the test set
6. Number of normal segments in the test set
7. Number of abnormal segments in the test set
8. Number of false alarms (good marked as bad)
9. Number of misfires (bad marked as good)

In both the training and test sets, ADSAS correctly flagged all bad segments and did not generate any false alarms.

Table 2. System performance on the diagnosis of TP and TPCT using data of two vehicle states only.

Exp. Number	**Train #**	**Train misfire**	**fuzzy Rules**	**Test #**	**Test normal**	**Test abnormal**	**False alarm**	**Misfire**
1	48	0	26	16	5	11	0	0
2	47	0	30	17	12	5	0	0

The second group of experiments we included segments from all four vehicle states. Table 3 shows the fuzzy variables and their associated number of fuzzy terms (note that only the number of fuzzy terms differs from the first group of experiments). Table 4 gives the results of two experiments using the same format as shown earlier.

Table 3. Fuzzy variables and the number of fuzzy terms associated with each variable for the diagnosis of TP and TPCT using data of four vehicle states.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
No of fuzzy terms	4	3	8	4	8	6	6	8

Table 4. System performance on the diagnosis of TP and TPCT using data of four vehicle states.

Exp. Number	**Train #**	**Train misfire**	**fuzzy Rules**	**Test #**	**Test normal**	**Test abnormal**	**False alarm**	**Miss fire**

3.	133	2	77	45	29	16	0	5
4.	129	1	63	47	49	7	6	0

In experiment 3, we had a training set of 133 normal segments. 2 of these were incorrectly classified by the fuzzy system indicating some rule conflicts due to a small amount of data inconsistency. The fuzzy intelligent system generated 77 fuzzy rules. When the system was tested using 29 normal segments and 16 abnormal segments, it generated no false alarms. However it missed 5 bad segments.

In experiment 4, we had a training set of 129 normal segments. Only 1 training data was mis-classified due to rule conflicts. The fuzzy intelligent system generated 63 fuzzy rules. When the system was tested using 49 normal segments and 7 abnormal segments, it correctly detected all abnormal segments, and marked only 6 normal segments as abnormal.

In all four experiments, we have only totally 3 mis-classifications of the training data. For the test set, the numbers of misfires are 0 in experiment 1 through 3 and 6 for experiment 4, which is acceptable. In experiments 1, 2 and 4, the system detected all abnormal segments, which are excellent results in engineering diagnostics.

6.2.2 Detection of stumbling using RPM and TP

The second group of signal diagnosis experiments was conducted on RPM using TP as a reference signal. We show the results of one of these experiments below. The typical behavior of TP and RPM was described earlier in section 1 and will not be repeated here.

The features we selected for this experiments are targeted at a specific fault common in vehicles, stumbling at idle (see Figure 12). To detect this fault, we used the following 10-dimensional feature vector, $x = <x_1, x_2, \ldots, x_{10}>$. The elements, x_i, are as follows:

x_1: CurState: State of the current segment.
x_2: PrevState: State of the previous segment.
x_3: Length: The length of the current segment.
x_4: PrevLength: The length of the previous segment..
x_5: TP_Range: The range of TP signal in the current segment.
x_6: RPM_Range: The range of RPM signal in the current segment.
x_7: Average of CD2 of TP signal
x_8: Average of CD3 of TP signal
x_9: Average of CD2 of RPM signal
x_{10}: Average of CD3 of RPM signal

This experiment uses the same solution variable, y, as in the first experiment, representing the fault status each segment.

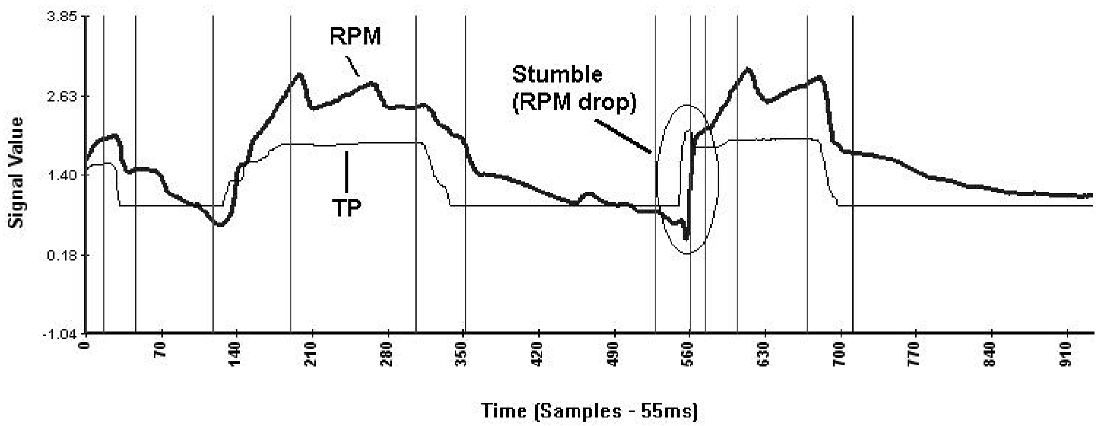


Figure 12: Example of stumbling. The location of stumbling is circled.

There were 124 normal segments and no abnormal segments in the training set. In the test set, there are 41 segments of which 25 are normal and 16 are abnormal. The data is collected from 7 recordings of bad vehicle behavior and 4 recording of good vehicle behavior. The fuzzy terms used for each element are listed in Table 5.

Table 5. Number of fuzzy terms for each element of x.

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
No. of fuzzy terms	4	4	8	8	6	6	6	6	6	6

After training, 184 fuzzy rules were generated by the fuzzy learning system. The result for training and test set is listed in Table 6. The meaning of each column in the results is the same as for the TP/TPCT experiments described earlier.

Table 6. System performance of diagnosis of stumbling by using TP and RPM

Train #	**Train misfire**	**fuzzy Rules**	**Test #**	**Test nor.**	**Test abnormal**	**False alarm**	**Miss fire**
124	0	184	41	25	16	8	1

There are no misclassifications in the training set. There were 8 false alarms about half of which we traced back to gear shifting patterns in the RPM signal not represented in the training set. We only had one misfire (normal labeled as abnormal.)

From the above experiments, we can see the diagnostic system using wavelets and fuzzy learning worked very well in the diagnosis of TP, and RPM stumbling problems. The results are encouraging, and we are currently applying the system to other more complex signal faults.

7. Conclusions

In this paper, we have described an intelligent system, ADSAS, using wavelet transform and machine learning, to solve real-world vehicle engine diagnosis problems. The experiments show encouraging results with a typical 95-100% correct classification rate for abnormal segments. In some cases, normal segments were labeled as abnormal (false alarm), usually through the effects of other system factors, such as gear changes, that were not completely represented in the training data. ADSAS is a powerful signal diagnostics tool. It is capable of carrying out major system functions including signal viewing, segmentation, feature extraction, learning, and testing. Though we only show results using signal pairs, (TP, TPCT) and (TP, RPM), our framework can be expanded to larger signal group without difficulty. In addition to its diagnostics capability, ADSAS is a good research testbed for experimenting different learning approaches, diagnostic feature selection, etc.

8. Acknowledgment

This work is supported in part by a Grant from NSF DMII-9612190, a contract from the Ford Motor Company and a CEEP grant from the College of Engineering and Computer Science at the University of Michigan. We also wish to thank Robert Mills and Shane Rachedi at the Advanced Diagnostic Design department in Ford Motor Company for their support to the project and contribution in vehicle fault diagnostics.

9. References

[1] M. Ayoubi, "Neuro-fuzzy structure for rule generation and application in the fault diagnosis of technical processes," in Proceedings of the 1995 American Control Conference, Seattle, (USA), pp. 2757-2761, 1995.

[2] A.Beschta, O.Dressler, H.Freitag, etc., A Model-based Approach to Fault Localisation in Power Transmission Networks, Intelligent Systems Engineering, 2(1), 1993, pp.3-14

[3] Bijaoui Albert , Rué, Frédéric , A multiscale vision model adapted to the astronomical images, Signal processing, Vol. 46, No. 3 (Oct. 1995), pp. 345-362.

[4] W. W. Boles abd B. Boashash, A human identification technique using images of the iris and wavelet transform, IEEE Trans. Signal Processing, vol.46, no.4, Apr. 1998, pp1185-1198

[5] M. Bsseville, et al., Modeling and estimation of multiresolution stochastic process, IEEE Trans. Information Theory, vol. 38, pp.766-784, Mar. 1992

[6] C. K. P. Chu and J. M. Mendel, "First Break Refraction Event Picking Using Fuzzy Logic Systems," IEEE Transactions on Fuzzy Systems, Vol. 2, pp. 255-266, Nov. 1994.

[7] Eric Chowanietz, Automobile electronics, Butterworth-Heinemann, 1995

[8] T. W. Cheng, D. B. Goldgof and L. O. Hall, "Fast clustering with application to fuzzy rule generation," in Procedures of the 1995 IEEE International Conference Fuzzy Systems, Yokohama, (Japan), pp. 2289-2295, 1995.

[9] Daubechies,I. *Ten Lectures on Wavelets*. Capital City Press. Montpelier,VT. 1992.

[10] D. Dvorak and B. Kuipers. Model-based Monitoring of Dynamic Systems. Proceedings of the Joint Conference on Artificial Intelligence (IICAI), August, 1989. pp. 1238-1243

[11] P. Eklund, J. Forsström, A. Holm, M. Nyström and G. Selén, "Rule generation as an alternative to knowledge acquisition: a systems architecture for medical informatics", Fuzzy Sets and Systems, vol.66, pp195-205, 1994.

[12] W.Hamscher, L.Console, J.deKleer (Eds.) - Readings in Model-based Diagnosis,Morgan, 1992

[13] S. Kadambe and G. F. Boudreaux-Bartels, Application of the wavelet transform for pitch detection of speech signal, IEEE Trans. Information Theory, vol. 38, no.2, Mar. 1992

[14] H. Kang, "An automated rule design of fuzzy logic controllers for uncertain dynamic systems", 2nd IEEE Int. Conference on Fuzzy Systems, pp261-266, 1993.

[15] Yi Lu**,** Tie Qi Chen, and Brennan Hamilton, "A Fuzzy System for Automotive Fault Diagnosis: -- Fast Rule Generation and Self-Tuning," accepted by IEEE Transaction on Vehicular, 1999.

[16] Yi Lu and Tie Qi Chen, "A Fuzzy Diagnositc Model And Its Application In Automotive Engineering Diagnosis," *J. of Applied Intelligence,* Vol. 9*, pp. 231-243, 1998*

[17] N.Lee, Q. Huynh, and S. Schwarz, New methods of linear time frequency analysis for signal detection, in Proceedings IEEE Int. Symposium Time-Frequency, Time-Scale Analysis, 1996

[18] S. Mallat, A theory for multiresolution signal decomposition: the wavelet representation. *IEEE trans. on PAMI*, 1989, 11(7), 674-693

[19] S. Mallat, Multifrequency channel decomposition of images and wavelet models, *IEEE trans. on Acoustic, speech and signal processing*, 1989, 37(12), 2091-2110

[20] H. Nomura, H. Ichihashi, T. Watanabe, "A Self-Tuning Method of Fuzzy Reasoning Control by Descent Method," Procedures of 4th IFSA Congress, Brussels, Vol. Eng. pp. 155-158, 1991.

[21] Alanna Quail and Adnan Shaout, "State-of-the-Art in Household Appliances using Fuzzy Logic," Second International Workshop on Industrial Applications of Fuzzy Control and Intelligent Systems, College Station, TX, December, 1992.

[22] F. C.-H. Rhee and R. Krishnapuram, " Fuzzy rule generation methods for high-level computer vision," Fuzzy Sets and Systems, vol. 60, pp. 245-258, 1993.

[23] J. Shapiro, Embedded image coding using zerotrees of wavelet coefficients, IEEE Trans. Signal Processing, vol. 41, pp.3445-3462, Dec. 1993

[24] T. Takagi and M. Sugeno, "Fuzzy Identification of Systems and its Applications to Modeling and Control," IEEE Trans. Systems, Man, and Cybernetics, vol. SMC-15, No.1, pp. 116-132, 1985.

[25] http://www.aber.ac.uk/~dcswww/vmbd/objectives.html