A Formal Approach to the Design of Feature-Based Multi-Sensor Recognition Systems

Mieczyslaw M. Kokar[1] , Zbigniew Korona

Department of Electrical and Computer Engineering, Northeastern University,
360 Huntington Avenue, Boston, MA 02115

Abstract

This paper shows an example of developing a fusion system in a formal framework, i.e., through the use of formal operators in the development process. Two main concepts of formal methods are *theories* and *models*. In our approach, the development of a fusion system consists of operations on theories and models. We show, on a simple example, how theories and models are combined in the process of designing a fusion system. We also compare the performance of a system developed according to our approach with a more traditional system.

Key words: information fusion, recognition, formal methods, wavelets, features

1 Introduction

The term *formal method* is used in the literature to refer to the use of formal logic in the process of specification, design and construction of computer systems (cf. [24,19,1]). In this approach, all facts used in the process of developing a computer system, including all assumptions and requirements, are expressed in a formal language with well-defined semantics. The development process is constrained to use a set of inferencing rules that are part of the formal system associated with the given formal language. As a consequence, all the properties of the developed system can be mathematically proven.

Formal methods for developing software systems are becoming more popular not only among researchers but also among application developers. Although

[1] Corresponding author: Tel: +1-617-373-4849; Fax: +1-617-373-8970; E-mail address: kokar@coe.neu.edu; URL: http://www.coe.neu.edu/˜kokar

many people still believe that formal methods are expensive and thus should be used only in *critical systems*, some others (cf. [19,1]) claim that, considering the overall system development cycle, especially the cost of testing systems, formal methods actually can reduce the overall cost of system development and also improve the quality of software (reliability and robustness). It is also expected that due to the great progress in the research on formal methods, they will become more cost efficient and consequently will become common practice for all kind of applications.

In spite of great progress in the area of formal methods, they still draw criticism from various angles. For one, the applications of formal methods are still limited, while the research literature presents mainly rather simple examples of applications. This paper is no exception. Another criticism is that they are not “user-friendly”, i.e., they require that the user and the developer possess highly sophisticated mathematical knowledge. The application example presented in this paper is also a simple one. However, since the concept it is used to explain is rather complex, we felt that it was necessary to use an example that can explain just the concept without adding the complexity of the application itself.

The goal of this paper is to show an example of the development of a fusion system in a formal framework, i.e., through the use of formal operators in the development process. Two basic concepts of formal methods are *theories* and *models*. These concepts are briefly discussed in Section 2. A theory is a collection of sentences in a formally defined language. A model is an interpretation of a theory in a mathematical structure. To use formal methods we need to deal with these two concepts. The main idea of this paper is that of fusion as an operation that is performed on both theories and classes of their models, and not just on one of these two components of a formal system. In other words, we view fusion as an operation that takes as its inputs theories and their models and produces a fused theory and a class of models of the fused theory. This approach differs from the approaches presented in the fusion literature where fusion is treated as an operation on either data (data fusion) or decisions (decision fusion). To shed more light on this subtle distinction, we stress that we do not introduce here a different classification; we still accept two kinds of fusion - data fusion and decision fusion. However, the point we are trying to make is that the most interesting issues of fusion are resolved at the design time, not at the run time of a fusion system. At the design time, the designer needs to make decisions on how to derive a fusion function that eventually will take sensor data and produce decisions. This is when the designer needs to consider theories and models and perform operations on these structures in order to derive a fusion function. We believe that fusion system designers perform this kind of operations anyway, but this is not said explicitly. In our approach, on the other hand, these structures (theories and models) are used and manipulated in an explicit manner.

Formal methods can be used in the development of any type of fusion system. In further discussion we assume that we know theories of targets (formal symbolic descriptions of targets). The following example explains the importance of considering both theories and models in the process of fusion.

Example 1 *Consider 2D sensors (e.g., vision sensors) operating in a 3D world whose goal is to recognize whether an object they observe is an ellipse or a circle. The sensors extract a set of edge points (x, y), in their own coordinates, and check whether all edge points satisfy either the theory of ellipse or circle:*

$$circle(x, y) \Longleftrightarrow x^2/r^2 + y^2/r^2 = 1$$

$$ellipse(x, y) \Longleftrightarrow x^2/a^2 + y^2/b^2 = 1, \;\; a \neq b$$

Assume the sensors' locations and orientations with respect to the object are such that their recognition decisions are always correct (no error due to skewing). For instance, the sensors S_1 and S_2 in Figure 1 satisfy such a condition. In such a case the fusion operation should be the union of these two theories, i.e., both sensors should conclude *circle* whenever the world contains a circle, and *ellipse*, whenever the world contains an ellipse. However, if the two sensors are placed differently, like sensors S_2 and S_3 in Figure 1, we cannot use such a simple operation. It's easy to see that the fusion operation must take into consideration the orientations and locations of the two sensors. In other words, we need to model the world in order to derive a theory that should be checked in the recognition process; the theories themselves are not sufficient for such a design decision.

The central contribution of this paper is an example of a formal method based design of a fusion system. The design is composed of a sequence of formal operators. We present a step-by-step analysis of the formal aspects, especially of the issue of consistency, involved in the design of a fusion algorithm using formal methods. Although it is not the focus of this paper, we also compare the performance of a system implemented using the approach described in this paper, with a system developed by a more traditional approach. Towards this aim, we compare the quality of recognition of our system with a system that uses an entropy-based feature selection [20] and a neural network for recognition. The results of the comparison for the scenario described in this paper are shown in Section 8. In [17], we used this approach to the domain of lung sound recognition.

In the next section we briefly review the essence of the formal method approach and then, in Section 3, we describe the formulation of the design problem of automatic target recognition and fusion using formal methods. Section 4 shows the complexity of the feature selection problem and the need for fusion. In Section 5 we describe the structure of a target recognition system based on this approach. This is followed by the discussion of the issue of consistent

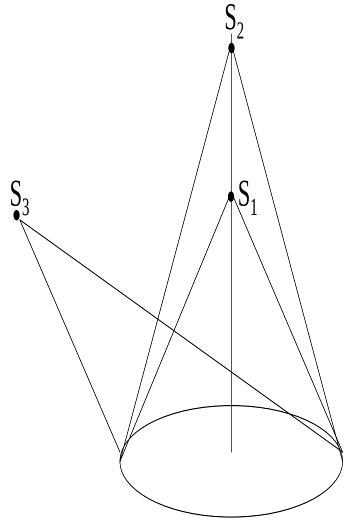


Fig. 1. Example: Circle or Ellipse?

fusion of theories and their models (Section 6). In Section 8 we describe the results of experimental evaluation of the resulting system. In Section 9 we present our conclusions.

2 Formal Methods

Model theory is a branch of mathematical logic which deals with the relation between a *language* and its *interpretations*, or *models* [5]. A first-order *language* L is a collection of symbols consisting of relation symbols $(P_1, \ldots, P_R)$, function symbols $(f_1, \ldots, f_F)$, and constant symbols $(C_1, \ldots, C_C)$

$$L = \{P_0, \ldots, P_R,\ f_0, \ldots, f_F,\ C_0, \ldots, C_C\}. \tag{1}$$

Each relation symbol P_i represents an n_i-placed relation, where $n_i \geq 1$. Similarly, each function symbol f_j represents an m_j-placed function, where $m_j \geq 1$. To formalize the meaning of the symbols in the language L, we use other symbols, like parentheses (")","("), variables $(y_0, \ldots, y_i, \ldots)$, logical connectives ($\wedge$ (and), $\Longleftrightarrow$ (equivalence), $=$ (identity), $\neg$ (negation), quantifier ($\forall$)) [5]. In our application, we also use some special symbols, like the linear order relation ($\leq$) and addition operation ($+$). Similarly as in [14], we treat these symbols as logical symbols, having always the usual intuitive meaning. A *grammar* for L defines how these symbols can be combined to form *well formed formulas (wffs)*, or simply *formulas*, of L. *Wffs* with no free variables are called *sentences*.

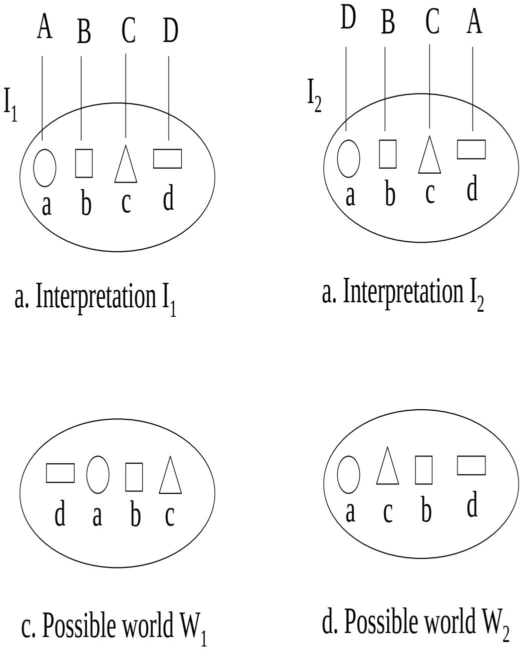
A language can be made into a *formal system* by adding *logical axioms* and *rules of inference* (cf. [5]). This allows us to define *proofs*, i.e., finite sequences of *wffs* of L in which each *wff* is either an axiom or a result of the application of the inference rules to preceding *wffs*. All *wffs* in such a proof are called *theorems*. The fact that a *wff* Q is a theorem, i.e., that it can be proved from the axioms of the formal system, is written as $\vdash Q$. If a set of *wffs* Q' is required to prove Q, then we write $Q' \vdash Q$. A set of sentences T of L is called a *theory* if it is closed under $\vdash$. Typically, theories are *presented* by *axioms*, i.e., sentences of T such that the consequences of $\vdash$ for these sentences are the same as for T.

Sentences of theories are intended to state facts about a domain (a world). In order to establish the relationship between theories (languages) and domains the *satisfaction* relation is defined (cf. [5]). This relation is defined inductively by establishing an *interpretation* of all relational symbols as elementary relations among the elements of the domain, functional symbols as functions on the elements of the domain and constant symbols as elements of the domain. More specifically, a *model* M for the language L is a pair $< A, I >$, where A is the *universe* (domain) of the model and I is the *interpretation function*, or simply an *interpretation*. Therefore, each n_i-placed symbol P_i from the language L corresponds to an n_i-placed relation $R_i \subset A^{n_i}$ on A, each m_j-placed f_j (symbol) corresponds to an m_j-placed function $f_j : A^{m_j} \rightarrow A$ and each constant symbol C_k corresponds to a constant $x_k \in A$. Whenever all the relations represented by a sentence Q hold in M, we say that Q is *satisfied* in M and write it as $M \models Q$. Whenever all sentences of a theory T are *satisfied* in M, we write $M \models T$ and say that M is a model for T.

For a simple example, consider a domain (see Figure 2) consisting of objects of four types: circle a, square b, triangle c, and rectangle d. Consider also a language consisting of one binary relation symbol $left(x, y)$ and four constants A, B, C, D

$$L = \{left, A, B, C, D\}$$

The symbols A, B, C, D don't have any particular meaning until we assign them to particular objects in the domain through an interpretation function I. First, assume that the predicate $left(x, y)$ has the usual interpretation: "the object denoted by x is located to the left (not necessarily immediately) of the object denoted by y". The symbols a, b, c, d, on the other hand, are not part of the language L. They are meta-symbols that we use to define the meaning. The symbols of the language can be assigned in many different ways to the objects in the domain, but we need to have unique identifiers of the objects



a. Interpretation I_1

a. Interpretation I_2

c. Possible world W_1

d. Possible world W_2

Fig. 2. Example: Models vs. Theories.

in the domain (in this case these are a, b, c, d) in order to define the meaning uniquely.

In this example we consider two interpretation functions, I_1 and I_2. We assume that both of them assign the same meaning to the relation symbol $left$, as specified above. However, they differ in the assignment of objects to constants. I_1 assigns circle to A, square to B, triangle to C, and rectangle to D. I_2 assigns circle to D, square to B, triangle to C, and rectangle to A. Consider the sentence (theory)

$$left(A, B)$$

and two possible worlds as in Figure 2. We can see that this theory is satisfied in both W_1 and W_2 under the interpretation I_1. However, under I_2 it is satisfied only in W_1. Consequently, W_1 would satisfy that theory under both interpretations (and thus would be a model of that theory), while W_2 would satisfy the theory only under the first interpretation.

3 The Recognition Problem

To formally specify the object recognition problem we need the following components:

- A language to express the recognition goal, the features used for recognition and the theories describing targets in terms of the features.
- Target theories to be used for recognition
- The classes of models for the target theories

Additionally, to give the context of the recognition problem, we need to specify the signals used for feature extraction and the features. We provide all of this information in the following sections.

3.1 Example Scenario

To introduce formal notation we use the following example scenario. Consider a 2D world consisting of objects, sensors and a light source (see Figures 3 and 4). There are two 1D sensors: a vision sensor and a range sensor. The vision sensor measures intensity of the light reflected by the object and the range sensor measures the distance to the object. Both sensors scan the world by sliding along the line parallel to and at distance h from the ground and provide a 1D array of data points (one frame at a time. The goal in this recognition problem is to distinguish triangles from rectangles using complementary information provided by the two sensors.

For this scenario, we made several simplifying assumptions: there are only two types of objects in this world – isosceles triangles and rectangles; the objects are illuminated by a parallel light source; at any time only one object exists in the world; objects are stationary; there is no scattering; triangles are always positioned with their base on the ground; the length of the bases of triangles and rectangles are equal; there are no shadows on the ground.

The two sensors provide complementary information about this simple world. This complementary information is useful because, under some conditions, it is difficult or even impossible to distinguish a rectangle from a triangle using either a range sensor or a vision sensor alone. Figures 3 and 4 show two extreme cases when it is almost impossible to make a recognition decision based only on range data or based only on intensity data. In Figure 3, an isosceles triangle and a rectangle illuminated with vertical light produce identical intensity signals. But the range sensor gives two distinct signals. Figure 4 shows a scenario in which it is much easier to make a correct recognition decision based on the intensity data than based on range data, especially when measurement noise is present. In the following, we describe how our design methodology is used for this scenario.

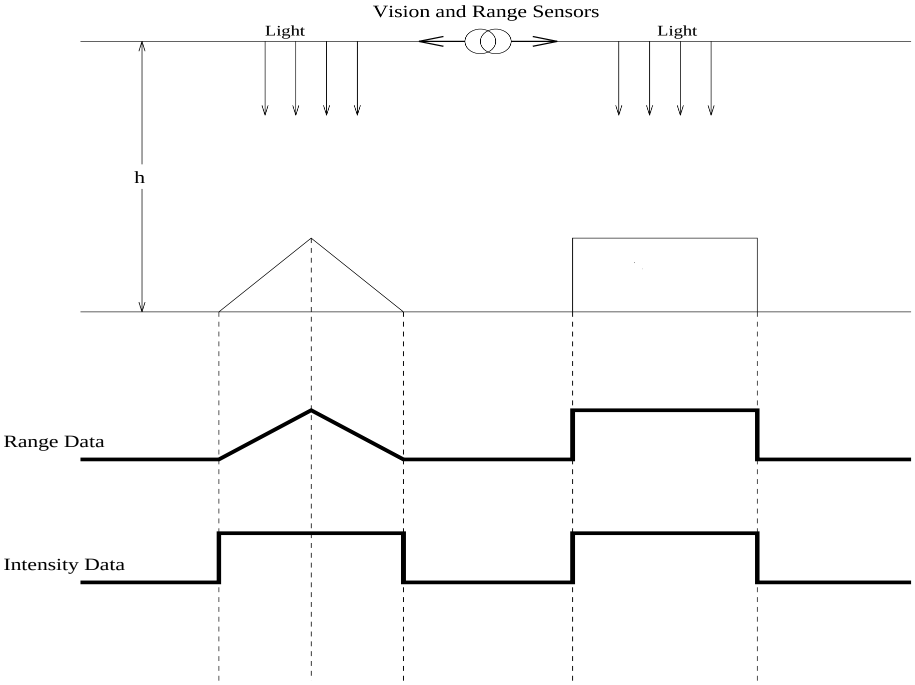


Fig. 3. Examples of Targets and Signals

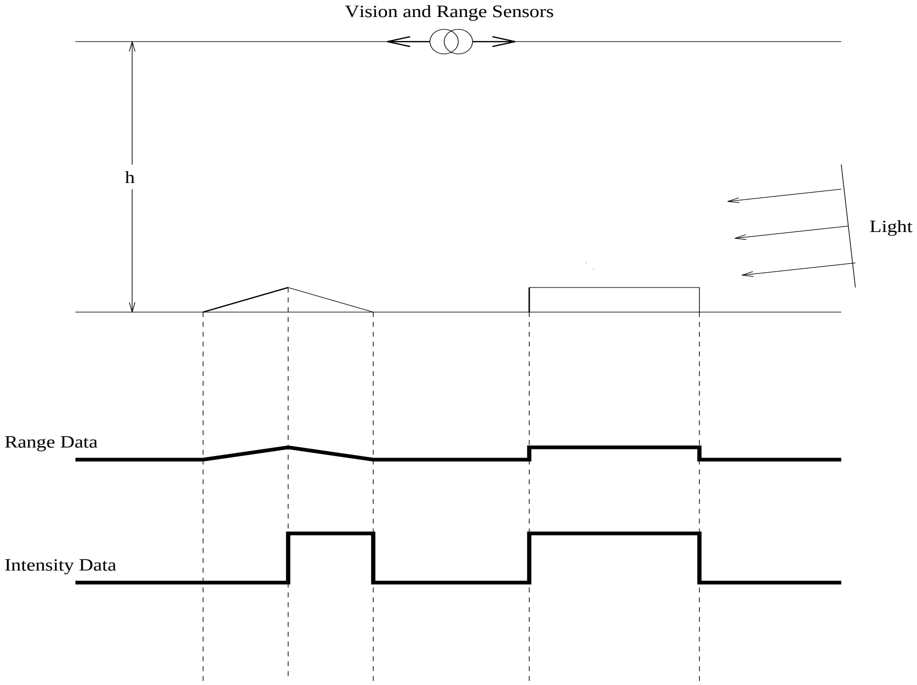


Fig. 4. Examples of Targets and Signals

3.2 Features

Examples of features are edges and corners of objects or the shape of main transients in signals. A variety of approaches to feature extraction from measurement data are known. Commonly used techniques include the Fourier transform, moment feature space, the Hough transform, Wigner distribution feature space, orthogonal polynomials, and Gabor functions [8]. In this study we used the wavelet transform for feature selection. The advantage of wavelets is that often the important target features are expressed by combination of edges, spikes and transients in an input signal. Therefore, these features are characterized by local information in both the time and frequency domains. The wavelet processing approach allows extraction of features in both these domains simultaneously [20].

A signal $s(n) \in R^N$, $n \in \mathbf{Z}$, where $N = 2^J$, can be recursively decomposed into lower-resolution signals at the decomposition levels $j = 0, 1, \ldots, J$. We used the decomposition scheme called *discrete wavelet packet decomposition* (DWPD) [7]. According to this scheme, at the decomposition level j, there are 2^j frequency bands. At the level 0 there are 2^J wavelet coefficients. At the level 1, the signal is decomposed into two frequency bands with $\frac{2^J}{2} = 2^{J-1}$ wavelet coefficients in each band. The total number of coefficients at level 1 is 2^J, i.e., the same as in the original signal. Similarly, at any level $j+1$, the number of bands is twice as in level j, each containing twice less wavelet coefficients, resulting in the same total number of coefficients at every level. Consequently, the DWPD coefficients can be represented as a two-dimensional matrix, where the first row contains the original signal, the second row contains the coefficients of the two bands of the decomposition level 1, and so on. For a signal s from the sensor r, we denote such a matrix of wavelet coefficients as $W_{sr}(j, b, n)$, where j represents the decomposition level, b - the frequency band, and n is the coefficient index (in the case of time signals it corresponds to time). Therefore, the DWPD transforms a sensor signal in the time domain into the time-frequency (wavelet) domain.

DWPD can be considered as a recursive decomposition of a vector space into two mutually orthogonal subspaces, with two subbases for the frequency bands $2b$ and $2b+1$ at level $j+1$ representing the same vector subspace as one base in the frequency band b at level j. Since the related bases at consecutive levels are not independent, the DWPD scheme generates a number of dependent subbases out of which a base for the whole space can be chosen in many different ways. In our approach we used the Best Discriminant Basis Algorithm (BDBA) [7,20], which selects a complete orthonormal basis that is best for representing a signal, i.e., such that minimizes some "information cost" measure. For classification problems, where the goal is to find a basis in which a class of signatures is best discriminated from all other classes of signatures, this measure must capture a "statistical distance" among classes. Saito [20] built such a measure using the notion of *relative entropy.*

The BDBA algorithm [20,7] starting with the highest level of decomposition, i.e., with the leaves of the tree, prunes the tree by replacing two frequency bands $2b$ and $2b+1$ at the level $j+1$ with one frequency band b at level j, whenever this substitution gives more discriminant power to the representation as measured by the chosen discriminant measure. The application of this algorithm results in a best (with respect to a given signature database) orthonormal basis consisting of selected subbases (frequency bands) for each decomposition level j. This basis is called *Most Discriminant Basis (MDB).* It is used for feature extraction, i.e., for a given signal s from sensor r, the pairs $((j,b,n), W_{sr}(j,b,n))$ are used as features (where the second elements are coefficients of the signal in the best basis and the first elements are their location in the DWPD). The total number of signal features is equal to the number N of samples of the measurement signal. In our case, the total number of components (wavelet coefficients) in this basis is equal to 128, i.e., it is equal to the number of samples in the object signature.

3.3 Languages

In our example we assume that we have two languages, L_r and L_i, for a range sensor and an intensity sensor, respectively.

The language L_r for the range sensor is:

$$L_r = \{rect_r, trian_r, f_r, C_{r_0}, C_{r_1}, \ldots, C_{r_7}, C_{r_8}\}, \tag{2}$$

with the intended interpretation:

- $rect_r$, $trian_r$ are 9-placed relation symbols (rectangle and triangle objects to be recognized using the range data). These two relations are used to express the goal of target recognition.
- f_r is an 1-placed function symbol – a function that maps range feature indices into feature values,
- $C_{r_0}, \ldots, C_{r_8}$ are constant symbols (range feature indices).
- Pairs of constants and functions $(C_{ri}, f_r(C_{ri}))$ are called *symbolic features,* or simply *features.*

The language L_i for the intensity sensor is:

$$L_i = \{rect_i, trian_i, f_i, 0, 1, C_{i_0}, C_{i_1}, \ldots, C_{i_7}, C_{i_8}\}, \tag{3}$$

The intended interpretation of the elements of this language is similar to that of L_r.

3.4 Target theories

For a given target, we have one target theory for each sensor. Each theory is a collection of sentences expressed in terms of relation symbols, constant symbols and function symbols. In this paper, for simplicity, we describe two target theories as versions of a sensor theory. The feature-level theory T_r for the range sensor consists of the following sentences that state the fact that the constants are linearly ordered.

$$C_{r_0} \leq C_{r_1} \leq C_{r_2} \leq C_{r_3} \leq C_{r_4} \leq C_{r_5} \leq C_{r_6} \leq C_{r_7} \leq C_{r_8}. \tag{4}$$

The following two formulas represent the two versions of T_r. If the formulas defined by Equation 5 are included this becomes the theory for rectangles, while if the formulas defined by Equation 6 are included this becomes the theory for triangles. Note that the symbols $rect_r$ and $trian_r$ are not necessary, from the logical point of view. They are introduced into the theory by definition in order to enhance the readability of the theories. They are a shorthand notation for the formulas defined by Equations 5 and 6.

$$\begin{aligned} rect_r(C_{r_0}, \ldots, C_{r_8}) &\equiv f_r(C_{r_0}) = \ldots = f_r(C_{r_8}) && (5)\\ trian_r(C_{r_0}, \ldots, C_{r_8}) &\equiv f_r(C_{r_0}) \leq f_r(C_{r_1}) \leq f_r(C_{r_2}) \leq f_r(C_{r_3}) \leq f_r(C_{r_4}) \wedge \\ & \quad f_r(C_{r_0}) = f_r(C_{r_8}) \wedge f_r(C_{r_1}) = f_r(C_{r_7}) \wedge \\ & \quad f_r(C_{r_2}) = f_r(C_{r_6}) \wedge f_r(C_{r_3}) = f_r(C_{r_5}), && (6) \end{aligned}$$

The recognition problem is to decide whether one of these sets of relations (Eq. 5 or 6) is fulfilled in the world.

The feature-level theory T_i for the intensity sensor consists of the following sentences of which Equation 8 describes the sentences that represent the rectangle version and Equation 9 represents the sentences that represent the triangle version.

$$C_{i_0} \leq C_{i_1} \leq C_{i_2} \leq C_{i_3} \leq C_{i_4} \leq C_{i_5} \leq C_{i_6} \leq C_{i_7} \leq C_{i_8}. \tag{7}$$

$$\begin{aligned} rect_i(C_{i_0}, \ldots, C_{i_8}) &\equiv 1 \leq f_i(C_{i_0}) = \ldots = f_i(C_{i_8}), && (8)\\ trian_i(C_{i_0}, \ldots, C_{i_8}) &\equiv f_i(C_{i_0}) = \ldots = f_i(C_{i_8}) = 0, && (9) \end{aligned}$$

3.5 Models

To connect the theories to the world, we need to construct models for the languages. For each model we need a domain that represents the constants of the theory, and an interpretation function that maps particular constants into elements of the domain.

The model M_r of the language L_r is:

$$M_r =< A; rect_r, trian_r, W_{sr}, 0, 1, \ldots, 8; I_r >, \qquad (10)$$

where $A = \{0, \ldots, 8\}$ is a universe of the model M_r. In our case, these numbers are indices of the nine wavelet coefficients selected out of the complete Discrete Wavelet Packet Decomposition (DWPD) of a given signal [7] using the BDBA algorithm outlined in Section 3.2. The function

$$I_r : \{C_{r_0}, \ldots, C_{r_8}\} \rightarrow \{(j, b, n)\} \qquad (11)$$

is an interpretation function that maps symbols of the language L_r to appropriate relations, functions, and constants in the universe A. I_r assigns constants $0, \ldots, 8$ in the model M_r to the constant symbols $C_{r_0}, \ldots, C_{r_8}$ in the language L_r respectively. Moreover, I_r assigns the function $W_{sr} : A \rightarrow A$ in the model M_r to the symbol f_r in the language L_r. W_{sr} is the wavelet decomposition the meaning of this inequality is not obvious. In our case, this symbol was interpreted as if j were the most significant digit in a three-digit number and n were the least significant digit. The interpretation function for constants was constructed by first ordering the selected wavelet coefficients according to the inequality relation and then assigning the number 0 to the first coefficients, 1 to the second, and so on.

The model M_i of the language L_i is:

$$M_i =< A; rect_i, trian_i, W_{si}, 0, 1, \ldots, 8; I_i > . \tag{12}$$

Again, nine features were selected. The first four features corresponded to the rising edge of a rectangle and the last five features corresponded to the vertex of a triangle.

3.6 Formal Method Based Target Recognition

The goal of an automatic target recognition system is to derive a classification decision t (in our example $t \in \{rectangle, triangle\}$), based upon the information as described above. For this, there must be a decision procedure that incorporates all of the above information, i.e., information about signals, features, targets (target theories) and their models. There are two major approaches to the development of systems using formal methods (cf. [24]): *model-based* and *algebraic*. In the model-based approach, observations can be treated as elements of a structure A that is a candidate for a model M_t of a given theory T_t. A *model checker* is then invoked to check whether the observations fulfill the relations P_t of the theory. In the algebraic approach, observations can be treated as axioms of a theory and a *theorem prover* is invoked to check whether the axioms imply that the observations come from a given target.

In our approach, we first build structures A for models of given target theories and interpretations I, and then perform model checking in order to determine the classification of a target. More specifically, we check which of the theories (or versions of a theory) is satisfied by features extracted from signals s_r, s_i coming from two sensors. To achieve this goal, we simply check whether the relation defined by some formula, e.g, Equation 5 of the theory T_r, holds in the structure A, or more specifically, among those elements of A that are assigned to constants of the theory through the interpretation function I_r.

We assume that we know target theories for two sensors. In order to be able to derive target recognition decisions based upon two sensors we need to have a fused theory that combines features and theories of both sensors.

4 The Need for Feature Fusion

While the wavelet transform provides features that have high expressive power, it also leaves us with the problem of choice – which of the features to choose for a specific application. Features extracted from measurement data are often redundant or have very limited discriminant power. We can significantly reduce the computational complexity by eliminating these features from active participation in the recognition process. For instance, if the wavelet transform generates $n = 128$ features, there are

$$C(n,k) = \begin{pmatrix} n \\ k \end{pmatrix} = \frac{n!}{(n-k)!k!} = 226,846,154,180,800 \qquad (13)$$

possible combinations of selecting a subset of $k = 10$ features out of the larger set of 128 features. These are more than $2 * 10^{14}$ possible choices!

When dealing with multi-sensor systems, we have to address one more choice: which features to choose from a particular sensor's features. For instance, if $n_1 = n_2 = 10$ features are selected from two sensors and if we want to select $n = 10$ features out of these $n_1 + n_2$ features, we have

$$C(n_1+n_2,n) = \begin{pmatrix} n_1+n_2 \\ n \end{pmatrix} = \frac{(n_1+n_2)!}{(n_1+n_2-n)!n!} = 184,756 \qquad (14)$$

possible choices.

The criterion used for selecting features needs to be based on the discriminant power of the set of features. The entropy is often used as a measure of expected discriminant power of particular features. In our work we used an entropy-based measure [20] for the process of *pre-selection* of a set of features. The final selection of features is based upon symbolic knowledge (target theories and their models).

5 Automatic Multi-Sensor Feature-based Recognition System (AMFRS)

In this section we describe the structure of our Automatic Multi-Sensor Feature-based Recognition System (AMFRS) (cf. [17]). In the next section we describe the formal aspects involved in the design of this kind of systems.

The AMFRS consists of four kinds of processing blocks: *DWPD, Feature Selection, Feature Fusion* and *Backpropagation Neural Network.* The input to the AMFRS is from two target detection systems (not covered in this paper), one for the range sensor and one for the intensity sensor. The *DWPD* (described in Section 3.2), transforms two signals s_r, s_i into the wavelet domain according to the algorithm described in [13]. The outputs of this algorithm are two matrices $W_{sr}(j,b,n)$, $W_{si}(j,b,n)$ of wavelet coefficients. Then *Feature Selection* selects some of the wavelet coefficients at certain locations, i.e., some pairs $((j,b,n), W_{sr}(j,b,n))$ for the range sensor, and similarly for the intensity sensor. These pairs are then used as *interpretations of symbolic features.* In this process, the *interpretation functions* I_r and I_i (see Eq. 11), are used to associate particular constants in target theories T_r and T_i, respectively, with elements of the domains of models M_r and M_i. The interpretation functions are constructed during the design phase using the signature databases and the target theories.

Feature Fusion combines features from both sensors into one set of fused features so that they become elements of a fused model M_f. Here again, the interpretation function I_f is used to associate constants of the fused theory T_f with the elements of the domain of the fused model M_f. Here the selection is from the set of features identified by both sensors. These features are passed to the neural-network based model checker for a recognition decision. The interpretation function I_f, the fused theory T_f and model M_f are constructed during the design phase as described in Section 6.

In the next phase, the AMFRS implements *soft model checking* using a backpropagation neural network. The network checks which of the (versions of) target theories is satisfied by the fused set of features. In other words, it checks whether the domain provided by the two signals satisfies the relations of the fused target theory associated with rectangles or with triangles. The neural network is trained using the known target theories under various noise conditions. The issue of noise can be addressed in many different ways. The main reason for selecting the neural netwok approach was to be compatible with the target recognition approach used in [20], which we used as a benchmark for our system.

6 Model-Theory Based Fusion

In this section we describe the issue of consistency involved in the design process of the *Feature Fusion* block. One way to implement this block is to fuse the decisions according to some fusion rule (decision fusion). Another approach is to fuse data (in our case *features*) and then classify targets based upon the fused data (data fusion). The question is, though, what is the theoretical basis

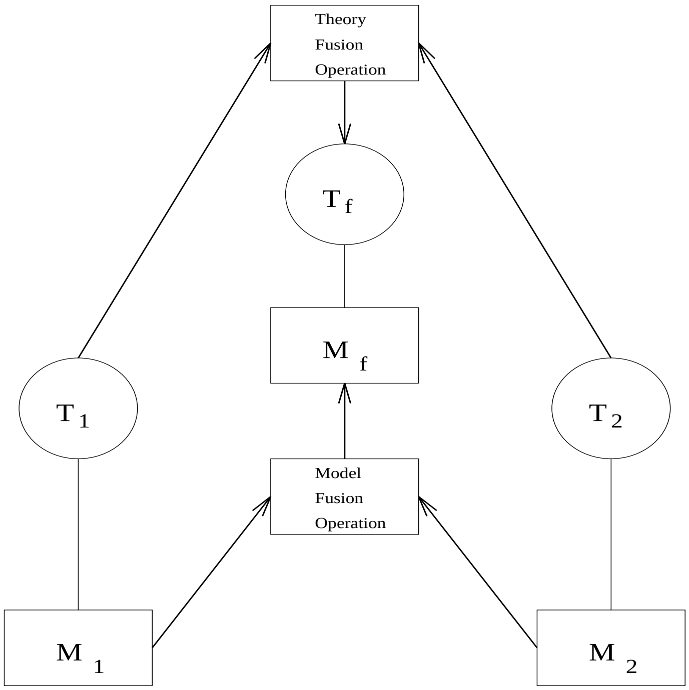


Fig. 5. Model-Theory Based Fusion Framework

for such fusion rules? Both data and decision fusion have been extensively studied and described in the fusion literature (cf. [2–4,6,9–12,15,18,21–23]). In our approach (cf. [16]), in order to derive a fusion rule we explicitly combine (fuse) theories and then let the system perform recognition based on model checking. Two theories T_1, T_2 and their classes of models M_1, M_2 are fused consistently so that the result is a fused theory T_f and its class of models M_f associated through an interpretation function I_f. A conceptual view of this framework for two sensors is shown in Figure 5.

Conceptually, fusion consists of two operations: *Theory Fusion* and *Model Fusion*. Theory fusion includes *language fusion*, the operation that combines languages (constants, functions and relational symbols) of the two theories T_r, T_i into one language (set of constant, function and relational symbols) of the fused theory T_f. It also includes *theorem fusion*, an operation that combines theorems of the two theories into one set of theorems. *Model Fusion* produces a fused model M_f (actually, this is a class of models), using M_r and M_i described in Section 3.5. The whole process must be *consistent*, i.e., M_f must be a class of models for the fused theory T_f. Therefore, fusion is a *formal system operator* that has multiple models and theories as inputs and a single theory and its model as output [16]. This interpretation of fusion differs from more traditional approaches. One of the distinguishing features of our approach [16] is that in our framework the most important issues of fusion are resolved at the design stage of system development. Additionally, since we deal with theories and models, the requirement of consistency of representations can be formally and explicitly specified. Such a formal representation is amenable to automated computer reasoning. Towards this aim, we investigated a number of operators of model theory [5], like *reduction, expansion* and *union.* These operators were used to derive fused languages, theories and models in the AMFRS design. Ideally, only *consistent* operators are used ensuring that the result of the application of an operator to two languages, theories and models constitutes a consistent formal system, i.e., the resulting structure is a model of the resulting theory. Although some of the operators known in model theory have this property, in order to fulfill the requirements of our specific applications, we had to use some other operators whose result of application needs to be checked for consistency for each case, i.e., the property of consistency is a proof obligation. The operators used in our experiments are described below. In Section 7 we show how these operators were applied in one of our scenarios.

Reduction Operator: A language L^r is a reduction of the language L if the language L can be written as

$$L = L^r \cup X, \tag{15}$$

where X is the set of symbols not included in L^r. A theory (sub-theory) T^r for the language L^r is formed as a reduction of the theory T for the language L by removing sentences from the theory T which are not legal sentences of the language L^r (i.e., those sentences that contain symbols of X). A model M^r for the language L^r is formed as a reduction of the model $M =< A, I >$ for the language L by restricting the interpretation function $I = I^r \cup I_x$ on $L = L^r \cup X$ to I^r

$$M^r =< A, I^r > \tag{16}$$

The important feature of the reduction operator is that it preserves the theorems of the original formal system, provided that they are not reduced by the operator. Given a language L_r, there is only one reduction of M_r.

Expansion Operator: A language L^e is an expansion of the language L if the language L can be written as

$$L^e = L \cup X, \tag{17}$$

where X is the set of symbols not included in L. A theory T^e for the language L^e is formed as an expansion of the theory T for the language L by adding a set of new axioms of the language L^e to the theory T. A model M^e for the language $L^e = L \cup X$ is formed as an expansion of the model $M =< A, I >$ for the language L by giving appropriate interpretation I_x to symbols in X

$$M^e =< A, I \cup I_x > . \qquad (18)$$

The expansion operator preserves all the theorems of the original theory in the expanded formal system. The expansion operator is not unique.

In our example (see Section 7), a special form of this operator was used to take advantage of some special properties of the recognition problem. Since the goal was to replace some of the constants, functions and relations with new ones, the expansion operator was used to introduce new symbols into the original language. These new symbols were interpreted using the interpretation of the original symbols, and then the original symbols were removed by the (following this step) reduction operator. Two operations were used to derive interpretations for new symbols.

(1) (Relation restriction) Given an n-placed relation $R \subset A^n$ in the model M, this model can be expanded with a n^e-placed ($n^e < n$) relation $R^e \subset (A')^{n^e}$, where $A' \subset A$. The n^e-placed relation R^e is then called a *restriction* of the relation R and is denoted as $R^e = R\mid_{(A')^{n^e}}$. This operation is a combination of projecting the relation R onto selected axes A^{n^e} and, at the same time, restricting its domain to the subset $A' \subset A$.

(2) (Product of relations) Given a n_1-placed relation $R_1(x_1, \cdots, x_{n_1})$ and a n_2-placed relation $R_2(y_1, \cdots, y_{n_2})$ in the model M, this model can be expanded with a new n-placed relation $R^e(z_1, \cdots, z_n)$, where $n = n_1 + n_2$, derived as a Cartesian product of the relations $R_1(x_1, \cdots, x_{n_1})$ and $R_2(y_1, \cdots, y_{n_2})$. Hence, $R^e(z_1, \cdots, z_n) = R^e(x_1, \cdots, x_{n_1}, y_1, \cdots, y_{n_2}) = R_1(x_1, \cdots, x_{n_1}) \times R_2(y_1, \cdots, y_{n_2})$.

In the same manner, a new function is constructed in the expanded model M^e using one of the following two procedures:

(1) (Function domain restriction) Given a function $f : A \rightarrow A$ in the model M, this model can be expanded with a function $f^e : A' \rightarrow A$, $(A' \subset A)$, where $f^e = f\mid_{A'}$ is a function whose domain has been restricted from A to $A' \subset A$. The function $f^e : A' \rightarrow A$ is then called a *restriction* of the function f.

(2) (Union of functions) Given a function $f_1 : A' \rightarrow A$, $(A' \subset A)$, and a (complementary) function $f_2 : (A \backslash A') \rightarrow A$ in the model M, this model can be expanded with a new function $f^e : A \rightarrow A$ derived as the union of the functions f_1 and f_2. Therefore, $f^e = f_1 \cup f_2$.

Union Operator: This operator generates a language L as a union of the languages L_1 and L_2

$$L = L_1 \cup L_2 \tag{19}$$

and a theory T for the language L, as the union of the theory T_1 for the language L_1 and the theory T_2 for the language L_2

$$T = T_1 \cup T_2. \tag{20}$$

To define a union of two models, the notation is expanded by including explicitly constants, relations and functions. The union $M =< A; R, f, X; I >$ of two models $M_1 =< A_1; R_1, f_1, X_1; I_1 >$ and $M_2 =< A_2; R_2, f_2, x_2; I_2 >$ is defined as

$$M = M_1 \cup M_2 =< A_1 \cup A_2; R_1 \cup R_2, f_1 \cup f_2, X_1 \cup X_2; I_1 \cup I_2 >, \tag{21}$$

where R, R_1, R_2 are relations; f, f_1, f_2 are functions; X, X_1, X_2 are constants; and I, I_1, I_2 are interpretation functions. This operator does not guarantee that the resulting structure is a model of the union of two theories; this property is a proof obligation and needs to be checked with each specific case of the application of this operator.

In summary, we selected a number of features from each of the sets of features associated with the two sensors. This resulted in the removal of some of the features from further consideration. The consequence of this is that we had to also remove the occurrences of the symbols associated with these features from the theories. And moreover, we had to adjust the interpretation function.

The selection of the number of features (nine) was somewhat arbitrary. However, since we wanted to compare the quality of recognition using the features selected using only the Most Discriminant Wavelet Coefficients (MDWCs) and the wavelet coefficients selected by the model-theory based AMFRS, we had to select the same number of features for both cases. Since we developed target theories that contained nine constants' symbols, we selected $k_f = 9$ MDWCs.

7 Fusion of Range and Intensity Features

To derive the fusion operators we used the model/theory combination operators of *reduction*, *expansion* and *union* described in Section 6. Below we describe how these operators are used by the designer of the system to fuse two theories T_r, T_i and two models M_r, M_i into a fused theory T_f and a fused model M_f, and then how the resulting model M_f is used in the AMFRS. More specifically, these operators are used to build the fused model M_f that includes $k_f = 9$ features, some of them selected from the $k_r = 9$ range features and some from the $k_i = 9$ intensity features. As we explain below, the first three features of this fused feature vector are the same as the first three intensity features corresponding to the rising edge of the rectangle. The next three fused features are the same as the last three intensity features corresponding to the vertex of the triangle. And the last three fused features are the same as the last three range features corresponding to the falling edge of a rectangle.

Our goal was to compare the quality of recognition of the AMFRS versus the quality of recognition using the MDWCs [20] as features. In this example, the features selected as the MDWCs happened to be only the intensity features. This was because all nine intensity features have more discriminant power (as measured by the relative entropy measure) than any of the range features.

7.1 Language Fusion

First, we define the language L_r^e to be an expansion of the language L_r

$$L_r^e = L_r \cup X_r, \tag{22}$$

where X_r is the set of symbols added to L_r. As we described in Section 6, we expand a language so that the new symbols are interpreted as relation restrictions, products of relations, function restrictions, or unions of functions. In our fusion example, we chose $X_r = \{rect_r', trian_r', f_r'\}$ and therefore

$$L_r^e = \{rect_r, rect_r', trian_r, trian_r', f_r, f_r', C_{r_0}, C_{r_1}, \ldots, C_{r_8}\}, \tag{23}$$

where $rect_r'$ and $trian_r'$ are 3-placed relation symbols (restrictions of $rect_r$ and $trian_r$), f_r' is an 1-placed function symbol (restriction of f_r).

Next, we define the language L_r^{er} to be a reduction of the language L_r^e:

$$L_r^{er} = \{rect_r', trian_r', f_r', C_{r_6}, C_{r_7}, C_{r_8}\}, \tag{24}$$

which was obtained by removing the symbols $X_r^{er} = \{rect_r, trian_r, f_r, C_{r_0}, \ldots, C_{r_5}\}$ from L_r^e.

In a similar manner, we define the language L_i^e to be an expansion of the language L_i:

$$L_i^e = L_i \cup X_i, \tag{25}$$

where $X_i = \{rect'_i, trian'_i, f'_i\}$ is the set of symbols added to L_i. As a result we have

$$L^e_i = \{rect_i, rect'_i, trian_i, trian'_i, f_i, f'_i, C_{i_0}, C_{i_1}, \ldots, C_{i_7}, C_{i_8}\}, \tag{26}$$

where $rect'_i$, $trian'_i$ are 6-placed relation symbols (restrictions of $rect_i$), f'_i is an 1-placed function symbol (restriction of f_i).

Next, we define the language L^{er}_i to be a reduction of the language L^e_i:

$$L^{er}_i = \{rect'_i, trian'_i, f'_i, 0, 1, C_{i_0}, \ldots, C_{i_5}\}. \tag{27}$$

by removing the set of symbols $X^{er}_i = \{rect_i, trian_i, f_i, C_{i_6}, C_{i_7}, C_{i_8}\}$ from L^e_i.

In the following step, we create the language L_{ri}, by applying the union operator to the languages L^{er}_r and L^{er}_i,

$$L_{ri} = L^{er}_r \cup L^{er}_i = \{rect'_r, rect'_i, trian'_r, trian'_i, f'_r, f'_i, 0, 1, C_{r_6}, C_{r_7}, C_{r_8}, C_{i_0}, \ldots, C_{i_5}\}. \tag{28}$$

Next, we create the language L^e_{ri}, as an expansion of L_{ri} by $X_{ri} = \{rectangle, triangle, f, C_0, C_1, \ldots, C_8\}$.

$$\begin{aligned} L^e_{ri} &= L_{ri} \cup \{rectangle, triangle, f, C_0, \ldots, C_8\} \\ &= \{rectangle, rect'_r, rect'_i, triangle, trian'_r, trian'_i, f, f'_r, f'_i, \\ &\qquad 0, 1, C_0, \ldots, C_8, C_{r_6}, C_{r_7}, C_{r_8}, C_{i_0}, \ldots, C_{i_5}\}, \end{aligned} \tag{29}$$

where $rectangle$, $triangle$ are 9-placed relational symbols (products of $rect_r$ with $rect_i$, and $trian_r$ with $trian_i$, respectively), f is an 1-placed function symbol (union of f'_r, f'_i), and $C_0, \ldots, C_8$ are constant symbols (renamed constants C_{i_j}, C_{r_j}).

And finally, we create the fused language L as a reduction of the language L^e_{ri}:

$$L = \{rectangle, triangle, f, 0, 1, C_0, \ldots, C_8\}. \tag{30}$$

7.2 Theory Fusion

Theory fusion parallels language fusion. We denote T^e_r to be a theory of the language L^e_r. T^e_r is an expansion of T_r. In addition to the axioms of T_r, it contains the following two axioms:

$$rect'_r(C_{r_6}, C_{r_7}, C_{r_8}) \equiv f_r(C_{r_6}) = f_r(C_{r_7}) = f_r(C_{r_8}) \qquad (31)$$
$$trian'_r(C_{r_6}, C_{r_7}, C_{r_8}) \equiv f_r(C_{r_8}) \leq f_r(C_{r_7}) \leq f_r(C_{r_6}) \qquad (32)$$

Axioms 31 and 32 are derived from the axioms 5 and 6 respectively by considering only last three features (C_{r_6}, C_{r_7}, and C_{r_8}).

Next, we create the theory T_r^{er} as a sub-theory of T_r^e. This theory contains the above two axioms and, additionally, the following axiom that was obtained from the axiom 4 by removing from it constants that are not part of the language anymore:

$$C_{r_6} \leq C_{r_7} \leq C_{r_8}. \qquad (33)$$

Similarly, by applying the expansion and the reduction operators to the axioms 8, 9, 7, we derive the theory T_i^{er} for the language L_i^{er}:

$$rect'_i(C_{i_0}, \ldots, C_{i_5}) \equiv 1 \leq f_i(C_{i_0}) = \ldots = f_i(C_{i_5}), \qquad (34)$$
$$trian'_i(C_{i_0}, \ldots, C_{i_5}) \equiv f_i(C_{i_0}) = \ldots = f_i(C_{i_5}) = 0, \qquad (35)$$

$$C_{i_0} \leq C_{i_1} \leq C_{i_2} \leq C_{i_3} \leq C_{i_4} \leq C_{i_5}. \qquad (36)$$

Then, we create the theory T_{ri} by applying the union operator to the theories T_r^{er} and T_i^{er}. This theory includes the axioms 31, 32, 33, 34, 35, and 36.

In the next step, we create the theory T_{ri}^e as an expansion of the theory T_{ri} for the language L_{ri}. For this, we create axioms for the additional new constants of L_{ri}. Since the intended interpretation of these new constants is as products of relations, we conjoin pairs of axioms with renamed constants: 31 with 34, 32 with 35, and 33 with 36.

$$rectangle(C_0, \ldots, C_8) \equiv 1 \leq f(C_0) = \ldots = f(C_5) \wedge f(C_6) = f(C_7) = f(C_8) \qquad (37)$$
$$triangle(C_0, \ldots, C_8) \equiv f(C_0) = \ldots = f(C_5) = 0 \wedge f(C_8) \leq f(C_7) \leq f(C_6) \qquad (38)$$

$$C_0 \leq C_1 \leq C_2 \leq C_3 \leq C_4 \leq C_5, \qquad (39)$$

$$C_6 \leq C_7 \leq C_8. \qquad (40)$$

The above four axioms constitute the axioms of the theory T_f, which is a sub-theory of T_{ri}^e obtained by removing from it the axioms which use the symbols that are not in L_f.

7.3 Model Fusion

In this process, models for the fused theories are created out of the models of the theories that have been fused. In the first step, we create the model M_r^e for the language L_r^e (L_r^e is an expansion of L_r)

$$M_r^e =< A; rect_r, rect'_r, trian_r, trian'_r, W_{sr}, W_{sr'}, 0, \ldots, 8; I_r^e > \tag{41}$$

by expanding the interpretation function I_r on L_r to the interpretation function I_r^e on $L_r \cup X_r$. Let I_{x_r} be an interpretation function on X_r. Since the language X_r is disjoint from the language L_r, we can write $M_r =< A, I_r \cup I_{x_r} >$. The interpretation of the new symbols, i.e., $rect'_r$, $trian'_r$ and f'_r, is:

$$rect'_r = rect_r \mid_{(A\cap\{6,7,8\})^3}, \tag{42}$$

$$trian'_r = trian_r \mid_{(A\cap\{6,7,8\})^3}, \tag{43}$$

$$W_{sr'} = W_{sr} \mid_{A\cap\{6,7,8\}} . \tag{44}$$

Next, we create M_r^{er}, a model for the language L_r^{er}, by restricting the interpretation function I_r^e, so that the symbols $rect_r, trian_r, f_r$ are removed from the interpretation.

In a similar manner, we create M_i^e for the language L_i^e by expanding the interpretation I_i:

$$rect'_i = rect_i \mid_{(A\cap\{0,1,2,3,4,5\})^6}, \tag{45}$$

$$trian'_i = trian_i \mid_{(A\cap\{0,1,2,3,4,5\})^6}, \tag{46}$$

$$W_{si'} = W_{si} \mid_{A\cap\{0,1,2,3,4,5\}} . \tag{47}$$

Next, we create M_i^{er}, a model for the language L_i^{er}, by removing the symbols $rect_i, trian_i, f_i$ from the interpretation I_i^e.

Then, we create a model M_{ri} for the language L_{ri} by applying the union operator to the model M_r^{er} for the language L_r^{er} and to the model M_i^{er} for the language L_i^{er}

$$M_{ri} =< A, I_{ri} >= M_r^{er} \cup M_i^{er} \tag{48}$$

Next, we create the model M_{ri}^e for the language L_{ri}^e by expanding the model M_{ri} (adding interpretation to $\{rectangle, triangle, f, C_0, C_1, \ldots, C_8\}$):

$$rectangle = rect'_r \times rect'_i, \tag{49}$$

$$triangle = trian'_r \times trian'_i, \quad (50)$$

$$W_{sf} = W_{sr'} \cup W_{si'}, \quad (51)$$

$$C_0 = C_{r_6}, C_1 = C_{r_7}, C_2 = C_{r_8}, C_3 = C_{i_0}, \ldots, C_8 = C_{i_5} \quad (52)$$

Note that it is possible to apply the union operator to functions $W_{sr'}$ and $W_{si'}$ because their domains are disjoint ($W_{sr'}$ is restricted to $\{6, 7, 8\}$ and $W_{si'}$ is restricted to $\{0, 1, 2, 3, 4, 5\}$.

And finally, we create the fused model M_f for the language L_f by removing from the interpretation the symbols that are not needed:

$$M_f =< A; rectangle, triangle, W_f, 0, \ldots, 8; I_f > . \quad (53)$$

8 Simulation Results

As we mentioned earlier, the main advantages of the proposed methodology are all the same as of any formal method. Additionally, in this section we present results of experiments using simulated range and intensity signals for the rectangle/triangle world described in Section 3.1. The goal was to compare the AMFRS versus both a single sensor system (using our methodology of system design) and versus an MDWC-based system. The quality of recognition was measured in terms of misclassification rate. In order to assess the impact of noise, noisy signals were simulated with the additive Gaussian noise varied at 11 levels (with step 2.5) within the deviation range of 0 through 25. For each level of noise, 100 different runs were generated.

The resulting misclassification rates for the single sensor case are shown in Figure 6. From this figure it can be seen that for one (range) sensor the performance of the AMFRS is consistently better than the misclassification rate of the recognition system that selects features using the MDWC approach.

These experiments showed that although the MDWC-based recognition system adapts well to the training set of triangle/rectangle range signatures, it does not have enough generalization power to perform equally well on the whole range of the test data. The reason for this is that the MDWC features used to train the classifier are concentrated in two time/frequency areas. The AMFRS, on the other hand, has a better generalization power due to selecting interpretable features which are more spread across the time/frequency domain.

Figure 7 shows the resulting AMFRS (continuous line) and MDWC (dashed line) misclassification rates for different levels of noise when signals from both

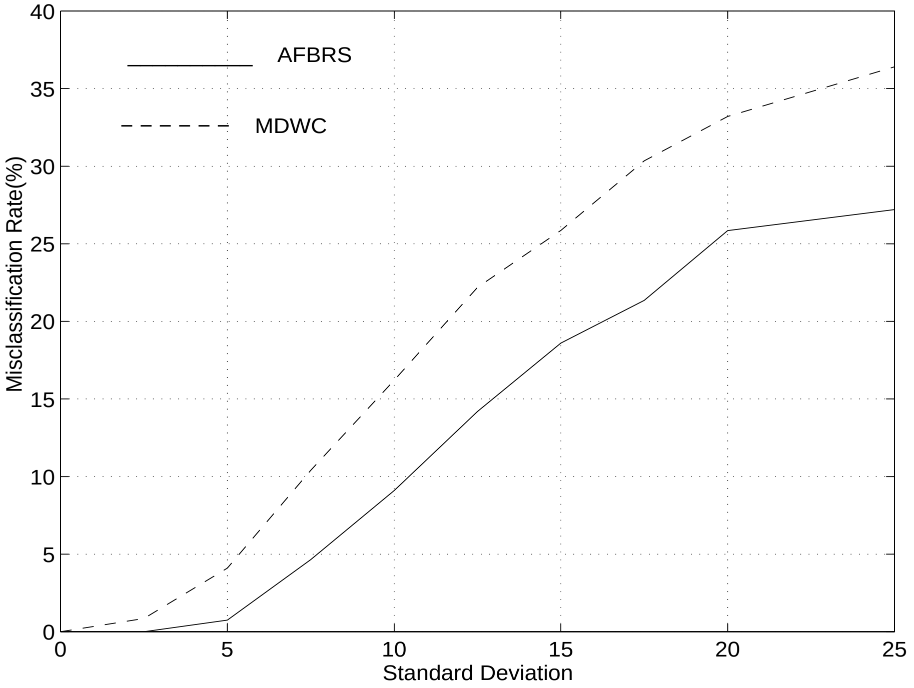


Fig. 6. Misclassification Rates for Single-Sensor (Range) Triangle/Rectangle Recognition Using Model Theory (AFBRS) and MDWCs for Feature Selection

sensors (range and intensity) are used. Additionally, this figure shows the misclassification rate of the AMFRS when only range data are used. As can be seen from this figure, both AMFRS and MDWC perform better than a single sensor system, but the AMFRS performs better than the MDWC-based system, when both sensors are used.

Again, our experiments showed that the MDWC-based recognition system adapts well to the training set of triangle/rectangle signatures, but it does not perform equally well with the whole test data set. Similarly as for the single sensor system, the MDWC features (only the intensity features happened to be selected by the MDWC entropy-based algorithm) that were selected and used to train the classifier are concentrated in one time/frequency area. The AMFRS was able to achieve a higher generalization power of the recognition by selecting interpretable features (both range and intensity) which are more spread across the time/frequency domain.

9 Conclusions

In this paper, we showed an example of an approach to designing fusion systems using formal methods. This approach allows the designer of the system to incorporate symbolic knowledge about the targets into the system design, provided that such knowledge (target theories and their interpretations) is

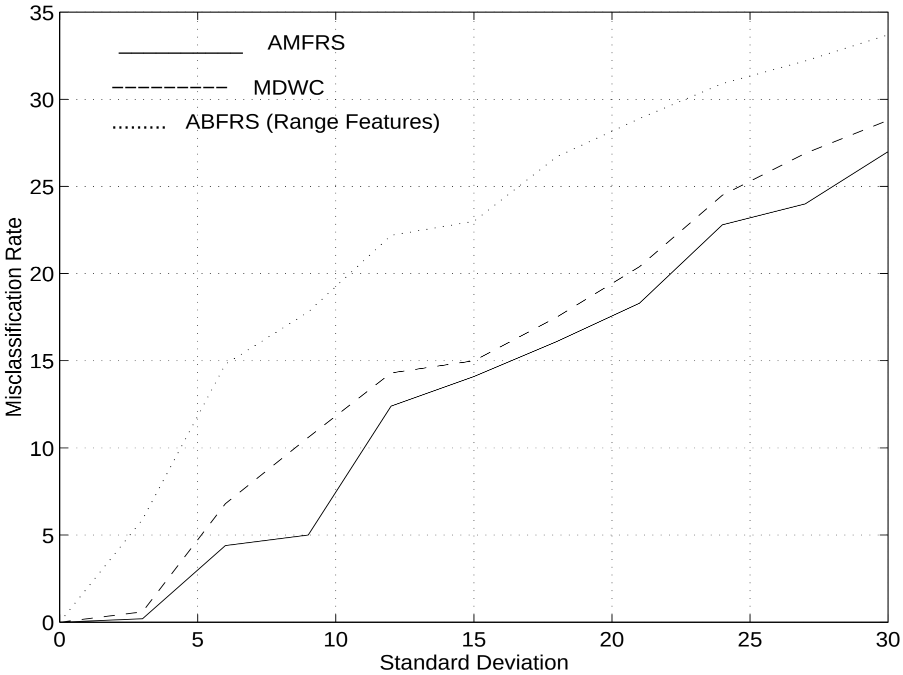


Fig. 7. Misclassification Rates for Multi-Sensor Triangle/Rectangle Recognition Using Model Theory (AMFRS) and MDWCs for Feature Selection

available. The design steps were explained on a simple automatic target recognition example. The generalization of the example to more complex systems is straight forward.

In many practical cases, there exists some symbolic knowledge about targets. This knowledge is often (implicitly) incorporated into ATR systems by the designers. Such an implicit representation of symbolic knowledge makes it very difficult to maintain the ATR system. Since in our approach, the knowledge is explicitly represented and kept as one module of the system, it is easier to maintain and extend. Also, since a formal representation of knowledge is used, this knowledge can be easily analyzed using generic formal methods tools. We showed how to use such symbolic knowledge in the process of designing a fusion system. In particular, a number of operators were introduced for the derivation of a fusion system. Some of these operators guarantee that the result of their application is a consistent system while for some others, the consistency property needs to be proved using formal method tools (theorem provers).

Another problem with incorporating symbolic knowledge into ATR systems is the lack of methods for interpreting symbolic features in sensory signals. In this paper we presented an approach to interpreting symbolic features as wavelet coefficients. Wavelets are a powerful tool for representing signals. The connection of this tool, through formal methods, to symbolic knowledge is a very important step towards bridging the gap between the powerful signal processing algorithms and efficient classifiers.

In this paper we applied our approach to design a system (AMFRS) for the triangle/rectangle recognition problem. We also showed through a series of experiments that the AMFRS has better recognition accuracy than the MDWC-based multi-sensor recognition system, that the single-sensor system based on our methodology of feature selection has better recognition accuracy than the MDWC-based single-sensor recognition system, and that the multi-sensor AMFRS has better recognition accuracy than our single sensor system. Although the MDWC-based system was able to adapt well locally to subsets of generated target signals, it did not perform as well on a larger variety of input data. The MDWC system selects features based only on data and thus may select wrong features when the data are concentrated within some specific range. The AMFRS, on the other hand, bases its feature selection on the universal knowledge that is given by the target theories. Note that this knowledge was input into the system by the system designer. The comparison is then between a data driven design and a knowledge driven design. We showed that a knowledge driven approach can lead to better results than a data driven approach. However, this can happen only if the knowledge (in our case target theories) is available during the design process.

We believe that in many practical situations the knowledge exists, although it is not explicit and is not represented as formal target theories. In order to make the formal approach easier to apply, the designers should be supported with appropriate tools that can extract the symbolic knowledge, represent it as formal theories, and check its consistency. Therefore, future research should address such issues as the use of machine learning techniques to extract symbolic knowledge from signature databases, the use of formal software engineering tools, like theorem provers, to check consistency of symbolic knowledge, the use of formal tools to support the construction of fused theories and models according to the procedure described in this paper.

Acknowledgments

This research was partially sponsored by the Air Force Office of Scientific Research under Grant F49620-98-1-0043 and by the Defense Advanced Research Projects Agency under Grant F49620-93-1-0490. The authors wish to thank Jerzy Tomasik for reviewing this paper and for his overall help with this research. We also want to thank the reviewers for their invaluable comments and suggestions.

References

[1] Formal methods specification and verification guidebook for software and computer systems. Technical Report NASA-GB-002-95, National Aeronautics and Space Administration, 1995.

[2] M. A. Abidi and R. C. Gonzales. *Data Fusion in Robotics and Machine Intelligence.* Academic Press, 1992.

[3] J. K. Aggarwal. *Multisensor Fusion for Computer Vision.* Springer-Verlag, 1993.

[4] S. S. Blackman. Theoretical approaches to data association and fusion. In C.W. Weaver, editor, *Sensor Fusion*, volume 931, pages 50–55. SPIE, Apr.1988.

[5] C. C. Chang and H. J. Keisler. *Model Theory.* North Holland, Amsterdam, New York, Oxford, Tokyo, 1992.

[6] J. J. Clark and A. L. Yuille. *Data Fusion for Sensory Information Processing Systems.* Kluwer Academic Publisher, Boston, 1990.

[7] R. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38, no.2:713–718, 1992.

[8] A. D. Kulkarni. *Artificial Neural Networks for Image Understanding.* Van Nostrand Reinhold, New York, 1994.

[9] B. V. Dasarathy. *Decision Fusion.* IEEE Computer Society Press, 1994.

[10] H. F. Durrant-Whyte. *Integration, Coordination and Control of Multi-Sensor Robot Systems.* Kluwer, 1988.

[11] G. D. Hager. *Task-Directed Sensor Fusion and Planning: A Computational Approach.* Kluwer, 1990.

[12] D. L. Hall. *Mathematical Techniques in Multisensor Data Fusion.* Artech House, Boston - London, 1992.

[13] L. Hong. Multiresolutional filtering using wavelet transform. *IEEE Transactions on Aerospace and Electronic Systems*, 29(4):1244–1251, 1993.

[14] N. Immerman. Languages which capture complexity classes. In *Proc. 15th Ann. ACM Symp. on the Theory of Computing*, pages 347–354, 1983.

[15] L. A. Klein. *Sensor and Data Fusion Concepts and Applications.* SPIE, Bellingham, WA, 1993.

[16] M. M. Kokar and J. A. Tomasik. Towards a formal theory of sensor/data fusion. Technical Report COE-ECE-MMK-1/94 (available at http://www.coe.neu.edu/ kokar), Northeastern University, ECE, Boston, MA, 1994.

[17] Z. Korona and M. M. Kokar. Lung sound recognition using model-theory based feature selection. *Applied Signal Processing*, 5:152–169, 1998.

[18] R. C. Luo and M. G. Kay. Multisensor integration and fusion in intelligent systems. *IEEE Transactions on Systems, Man and Cybernetics*, 19-5:901–931, 1989.

[19] J. Rushby. Formal methods and the certification of critical systems. Technical Report CSL-93-7, SRI International, 1993.

[20] N. Saito. *Local Feature Extraction and Its Applications Using a Library of Bases*. PhD thesis, Yale University, 1994.

[21] S. C. A. Thomopoulos. Sensor integration and data fusion. *Journal of Robotic Systems*, 7(3):337–372, 1989.

[22] P. K. Varshney. *Distributed Detection and Data Fusion*. Springer-Verlag, 1996.

[23] E. Waltz and J. Llinas. *Multisensor Data Fusion*. Artech House, Norwood, MA, 1990.

[24] J. M. Wing. A specifier's introduction to formal methods. *IEEE Computer*, 9:8–24, 1990.