Data vs. Decision Fusion in the Category Theory Framework

Mieczyslaw M. Kokar
Department of Electrical and Computer Engineering
Northeastern University
Boston, MA, U.S.A.
kokar@coe.neu.edu

Jerzy A. Tomasik
Universite d'Auvergne
LLAIC1, BP86 63172 AUBIERE
France
tomasik@llaic.u-clermont1.fr

Jerzy Weyman
Department of Mathematics
Northeastern University
Boston, MA 02115, USA
weyman@research.neu.edu

Abstract − *In this paper we first formally define the notions of* data fusion *and* decision fusion. *Then we formulate a theorem that decision fusion is a special case of data fusion. We show the meaning of this theorem on a simple example of edge detection. Edge detection can be done in two ways: by first fusing the original images and then detecting edges in the fused image (data fusion) or by first detecting edges in each image separately and then fusing the results (decision fusion) of edge detection in the decision fusion block. We show, first in general and then on the edge detection example, that decision fusion can be viewed as a special case of data fusion. To the designer of an information fusion system this means that the choice of the decision fusion approach over data fusion in any specific case needs to be supported by some additional consideration, for instance the computational complexity of the fusion algorithm.*

Keywords: Formal methods, category theory, data fusion, decision fusion, classification.

1 Introduction

One of the goals of our research is to develop methods for reasoning about an information fusion system in the design phase (cf. [4]). In other words, we want to be able to formally compare various design solutions before the system is implemented. For instance, we would like to be able to reason about the uncertainty of the system's decision associated with various design solutions. Having such a method would help the designer to choose the solution that is best for a given scenario. A typical example of such a decision is whether to first fuse data and then detect/recognize objects in the fused data or first detect/recognize objects in each signal/image separately and then fuse the decisions. The former solution is usually termed as *data fusion*, while the latter is called *decision fusion* [2, 9].

In this paper we ask a more general question - is it possible to compare the two solutions - data vs. decision fusion in general? More specifically, we ask whether decision fusion is a special case of data fusion. The answer to this question is positive - any decision fusion system can be viewed as a data fusion system. The implication of this statement to the designer of an information fusion system is that the choice of "data fusion" as a design solution does not really limit the designer since the designer is still able to achieve the same functionality of the system (exactly the same behavior) as if the choice were "decision fusion". Note that the inverse is not true. Note however that such a comparison can be done only from a given point of view. In this paper we take the point of view of the function and the behavior of the system. When we add another criterion, for instance the computational complexity of the fusion algorithm, the situation is quite different, since decision fusion is usually less complex than data fusion.

In this paper we use a running example of edge detection to explain our approach; this example is presented in Section 2. Since our goal is to reason about design choices in a formal way, we need to present our formalization of the information fusion problem. This formalization is briefly explained in Section 3. In Section 5 we state the theorem that decision fusion is a

special case of data fusion. And finally in Section 6 we provide our conclusions and suggestions for future work.

2 Example

In order to explain our ideas in this paper we use a simple example of an information fusion scenario. We consider two vision sensors $Sens_1$ and $Sens_2$ observing an object in the world (Figure 1). The first sensor (Sen_1) returns the image denoted as $I_1(x_1, y_1)$ and the second sensor (Sen_2) returns the image $I_2(x_2, y_2)$. The functions I_1 and I_2 consist of two subfunctions. For $Sens_1$, there is a function $g_1(x_1, y_1)$ which returns pixel values, which are then filtered by $h_1(x_1, y_1)$ returning the values of $I_1(x_1, y_1)$. Similarly, $Sens_2$ consists of two functions g_2 and h_2.

The goal of the fusion system is to utilize the information from both sensors in order to detect edges of the observed object. This goal can be achieved in two ways:

1. *Data Fusion:* Two images $I_1(x_1, y_1)$ and $I_2(x_2, y_2)$ are fused into one combined image $I(x, y)$ and then edge detection is performed on this image. The resulting edges (or more precisely, *edge points*) are denoted by $E(x, y)$.

2. *Decision Fusion:* The two images $I_1(x_1, y_1)$ and $I_2(x_2, y_2)$ are analyzed separately by edge detection algorithms. This results in edges $E_1(x_1, y_1)$ and $E_2(x_2, y_2)$. Then the detection information (edges) is fused into one $E(x, y)$.

As we can see, in the end both systems derive the same kind of global information about edges represented by $E(x, y)$.

For simplicity we assume that edge detection is based on the magnitude of the gradient, for the image of $Sens_1$, for the image of $Sens_2$ and for the fused image.

3 Formal Definition of Fusion

We formalize the information fusion problem in the formal specification language, Slang [8, 1]. The process of developing Slang specifications is supported by the Specware tool. Specware is based on category theory [7]. A specification consists of *specs*. Each spec can be viewed as a pair (Σ, T), where Σ are *signatures* (languages) and T - *theories* over the signatures. Signatures have the following form: $\Sigma = (\sigma, F)$, where σ are *sorts* and F are functions over the sorts. Theories associated with the signatures are represented by collections of axioms over the signatures. Specs are considered as *objects* in the category *Spec* related through *morphisms* [7]. Specs and morphisms are represented as *diagrams*. We always assume that our theories are *consistent*, i.e., that they have models, formally denoted as $M \models T$.

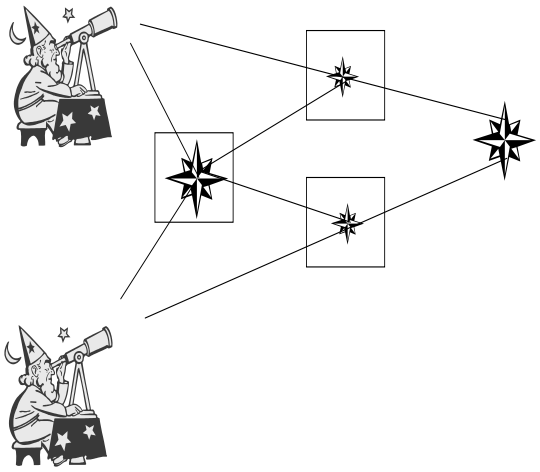

Figure 1: A Fusion Scenario

3.1 Data Fusion

In this paper we focus on two kinds of information fusion – *data fusion* and *decision fusion*. In general, the goal of data fusion is to develop a spec S_f and a fused class of models $\{M_f\}$, as described below. The inputs to this fusion process are some or all of the following specifications:

$$\begin{aligned} S_w &= ((X, E, \Delta : X \to E), T_w) \\ S_1 &= ((X_1, V_1, f_1 : X_1 \to V_1), T_1) \\ S_2 &= ((X_2, V_2, f_2 : X_2 \to V_2), T_2) \\ S_c &= (C = \{C_1, C_2, ...\}) \end{aligned} \tag{1}$$

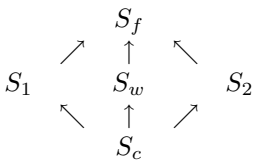

Figure 2: Data Fusion

The goal of data fusion is to find the diagram $\mathcal{D}$ - a diagram of relations among the specs (see Figure 2), where S_f is a specification:

$$S_f = ((X, E, \Delta : X \to E,$$
$$X_1, V_1, X_2, V_2, f_1 : X_1 \to V_1, f_2 : X_2 \to V_2, \quad (2)$$
$$D_f : (X_1 \to V_1) \times (X_2 \to V_2) \to (X \to 2^E)), T_f)$$

satisfying the conditions:

$$M_f \models T_f \quad (3)$$

$$T_f \vdash \forall_{x \in X}\ \Delta(x) \in D_f(f_1, f_2)(x) \quad (4)$$

In the above formulation S_w specifies the world that both sensors observe; X represents the world coordinates, E is the objects in the world. The function Δ assigns these objects to particular locations. We assume that we may have access to particular instances of this function. We use this capability for testing the resulting fusion system. S_w can contain theories T_w that capture known dependencies and constraints that the world is known to obey.

Referring to the example of Section 2, the coordinates of the world are X, Y. The objects are $E = [0, 1]$ - a subset of real numbers representing the confidence of an edge point being at a particular world location. The function Δ assigns to each location in the world a value from the interval $[0, 1]$.

$$\Delta : X \times Y \to [0, 1] \quad (5)$$

The specifications S_1, S_2 represent specifications of the two sensors. X_1 is the coordinate of the sensor specified by S_1 and X_2 is the coordinate of the sensor specified by S_2. V_1 and V_2 are sorts that denote values returned by the two sensors. The functions f_1, f_2 are the *measurement functions* of the two sensors. T_1 and T_2 specify theories of sensor operation.

In our example, both sensors have two coordinates denoted as X_1, Y_1 and X_2, Y_2, respectively. Their measurement functions are $f_1 = I_1$ for $Sens_1$ and $f_2 = I_2$ for $Sens_2$. The measurement functions return the values from V_1 and V_2, respectively. Since I_1 and I_2 are compositions of two functions, the theories of S_1 and S_2 must have appropriate axioms to this effect.

$$I_1 = h_1 \circ g_1 \quad (6)$$

$$I_2 = h_2 \circ g_2 \quad (7)$$

The specifications of the first sensor ($Sens_1$) is shown below. We do not show the specification for the second sensor since it is similar to the specification of the first sensor.

$$S_1 = ((X_1, Y_1, V_1, V_{11},$$
$$g_1 : X_1 \times Y_1 \to V_1,$$
$$h_1 : V_{11} \to V_1, \quad (8)$$
$$I_1 : X_1 \times Y_1 \to V_1),$$
$$I_1 = h_1 \circ g_1)$$

The sensor specification includes in its theory part the axiom stating that the function I_1 is computed as a composition of the measurement function g_1 and the filtering function h_1 (see Eq. 6).

S_c in Figure 2 is a collection of simple specs, specifications of *coordinate sorts*. The purpose of identifying these specs is to show the relationships between the world coordinates and the sensor coordinates. They unify sorts that represent the same coordinates. Consequently, we have $S_c = \{X_x, X_y\}$.

For our example we assumed that we want to associate X_1 and X_2 with X_x, Y_1 and Y_2 with X_y. The unification of sorts is achieved by specifying morphisms between particular specifications. In this example the morphisms would be

$$morphism : S_c \to S_1 = \{X_x \to X_1, X_y \to Y_1\} \quad (9)$$

$$morphism : S_c \to S_2 = \{X_x \to X_2, X_y \to Y_2\} \quad (10)$$

$$morphism : S_c \to S_w = \{X_x \to X, X_y \to Y\} \quad (11)$$

The specification S_f is obtained in two steps. First, a colimit of S_c, S_1, S_2 and S_w is taken. At this point some of the sorts, as explained above, are identified (or "glued" together). This means that some of the sorts listed in the spec S_f would actually be glued and thus that spec would not have as many sorts as shown. For instance, the six sorts would form two equivalence classes $\{X, X_1, X_2\}$ and $\{Y, Y_1, Y_2\}$. Note that this does not mean that in the final spec we would not distinguish between the variables of these two sorts. We would still have the variables representing the values coming from the two sensors separately. Only after *data association* is done could we use the same variables for the two sensors. In this paper we assume, for simplicity, that the coordinates of the two sensors are perfectly associated and thus will use the symbols X and Y to represent the coordinates of the two sensors in the final specification of the system.

In the second step the resulting specification is extended by adding the function D_f. Its signature is constructed out of the signatures of the two sensors and of the world. This function takes two measurement functions f_1,f_2 as inputs and returns a decision function that assigns subsets of objects to the world coordinates.

For our example, the morphisms $S_1 \to S_f$, $S_2 \to S_f$ and $S_w \to S_f$ would be specified first (similarly as the morphisms shown above) and then the colimit operation would be specified next. The resulting specification would include the sorts X, Y, E, the operations

$I_1, I_2, g_1, g_2, h_1, h_2$ and all the axioms from S_w, S_1, S_2. The colimit operation would guarantee that sorts are unified appropriately, and the operations are applied to the appropriate sorts. Additionally, it would insure that the axioms from the source specifications are preserved, i.e., they are theorems of the colimit specification. This kind of mechanisms for formally checking the colimit operation are part of the Specware tool [1].

The signature of the fusion function for our example would take the form as shown in Eq. 12 below. Note that the mapping is to the set E rather than to 2^E. This means that we expect a concrete value for each of the objects (in this case, edges) rather than a distribution of confidence as a result of the fusion process. This differs from our general specification where the mapping is to 2^E. The rationale behind this kind of mapping is to show that the decision is not always unique, in some cases it may return a number of possibilities rather than just one specific object.

$$D_f : (X_1 \times Y_1 \to V_1) \times (X_2 \times Y_2 \to V_2) \to (X \times Y \to E) \quad (12)$$

We do not elaborate further on what the form of the function D_f should be. We do not need to go into this level of detail to show the point that decision fusion is a special case of data fusion. This claim will apply to any function D_f.

3.2 Decision Fusion

In our framework decision fusion is expressed by the diagram of Figure 3.

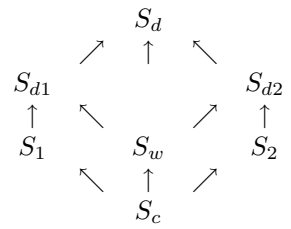

Figure 3: Decision Fusion

S_{d1} and S_{d2} represent the following specs:

$$S_{d1} = ((X_1, V_1, \Delta : X \to E, f_1 : X_1 \to V_1, D_1 : (X_1 \to V_1) \to (X \to 2^E)), T_{d1}) \quad (13)$$

$$S_{d2} = ((X_2, V_2, \Delta : X \to E, f_2 : X_2 \to V_2, D_2 : (X_2 \to V_2) \to (X \to 2^E)), T_{d2}) \quad (14)$$

The functions D_1, D_2 are the decision functions for the sensors $Sens_1$ and $Sens_2$ respectively. In the process of decision fusion these two functions are used instead of raw data. The spec S_d represents the decision fusion block.

$$S_d = ((X_1, V_1, X_2, V_2, \Delta : X \to E, f_1 : X_1 \to V_1, D_1 : (X_1 \to V_1) \to (X \to 2^E), f_2 : X_2 \to V_2, D_2 : (X_2 \to V_2) \to (X \to 2^E), D_d : (X \to 2^E) \times (X \to 2^E) \to (X \to 2^E)), T_d) \quad (15)$$

Note that in this spec D_d takes the assignments that are the results of application of functions D_1 and D_2 and combines these two assignments into one (fused) assignment.

Returning back to our example, we take the decision function D_1 to have the signature

$$D_1 : (X_1 \times Y_1 \to V_1) \to (X \times Y \to E) \quad (16)$$

In other words, the decision function D_1 takes the function I_1 and returns another function (the decision function) which maps the world coordinates to the values of edges. An edge in an image is manifested through a discontinuity (for continuous images) or a significant jump in the intensity value (in a digital image). There are various edge detection techniques (cf. [6, 3]). The simplest method is to take the gradient magnitude. Denoting the (normalized) gradient magnitude by $M(I)(x, y)$ we would have

$$D_1 \equiv M(I_1) \quad (17)$$

This information would be incorporated into the theory T_{d1} shown in the spec S_{d1}. T_{d1} would then incorporate the axioms about the gradient magnitude operator and the thresholds used for detection. Although D_2 could use a different edge detection algorithm, in this paper we assume, for simplicity, that D_2 also uses the same kind of "edgeness" operator.

The D_d operator can be defined in many different ways. In the following discussion we will use a very simple form:

$$D_d(M_1, M_2)(x, y) \equiv \bar{M}(x, y) = \frac{1}{2}(M(I_1)(x, y) + M(I_2)(x, y)) \quad (18)$$

4 The *subclass* Relation

In order to be able to compare various fusion systems we introduce the relation of *subclass*, which is a relation between fusion systems.

Definition 1 *Let S_f^1 and S_f^2 be two data fusion systems like in Figure 2, where all nodes except S_f^1 and S_f^2 are the same. We say that S_f^1 is a subclass of S_f^2 if there is a morphism of specifications $\mu : S_f^2 \to \bar{S}_f^1$, where $\bar{S}_f^1$ is a definitional extension of S_f^1, such that the diagrams shown in Figure 4 commute.*

$$\begin{array}{ccccccccc} \bar{S}_f^1 & \leftarrow & S_f^2 & \bar{S}_f^1 & \leftarrow & S_f^2 & \bar{S}_f^1 & \leftarrow & S_f^2 \\ & \nwarrow & \uparrow & & \nwarrow & \uparrow & & \nwarrow & \uparrow \\ & & S_1 & & & S_2 & & & S_w \end{array}$$

Figure 4: Commutativity Requirements for *subclass* Relations

5 Decision Fusion as a Subclass of Data Fusion

The idea that decision fusion is a special case of data fusion is captured by the following theorem (see Figure 5).

Theorem 1 *The class of decision function systems, as defined in Figure 3, is a subclass of data fusion systems, as defined in Figure 2.*

In this paper we provide only an outline of the proof of this theorem. A full proof is presented elsewhere [5]. In the proof we assume that we have a decision fusion diagram as in Figure 3. We need to produce a data fusion diagram as in Figure 2 such that there is a morphism from the diagram of Figure 2 to the diagram of Figure 3. As a first step we define S_f as a definitional extension $\bar{S}_d$ of S_d by defining a new function $\bar{D}_f : (X_1 \to V_1, X_2 \to V_2) \to 2^E$, where

$$\begin{array}{ccccccc} & & & S_d & & & \\ & & \nearrow & \uparrow & \nwarrow & & \\ & S_{d1} & & S_f & & S_{d2} & \\ & \uparrow & \nearrow & \uparrow & \nwarrow & \uparrow & \\ & S_1 & & S_w & & S_2 & \\ & & \nwarrow & \uparrow & \nearrow & & \\ & & & S_c & & & \end{array}$$

Figure 5: Decision Fusion is a Subclass of Data Fusion

$$\bar{D}_f \equiv D_d \circ (D_1 \times D_2). \tag{19}$$

This relation is expressed by the diagram as in Figure 6.

The definitional extension [8] S_d is equipped with an embedding $S_d \to \bar{S}_d$ which is the identity on all sorts,

$$\begin{array}{ccc} (X_1 \to V_1), & & \\ (X_2 \to V_2) & \overset{D_1 \times D_2}{\to} & (X \to 2^E) \times (X \to 2^E) \\ \searrow\ \bar{D} & & \swarrow\ D_d \\ & (X \to 2^E) & \end{array}$$

Figure 6: Derivation of Data Fusion Function from Decision Fusion Function

operations and axioms from S_d. We define the arrows from $S_i \to S_f$ $(i = 1, 2)$ as a composition

$$S_i \to S_f \equiv S_i \to S_{di} \to S_d \to \bar{S}_d \tag{20}$$

Then we define the arrow $S_w \to S_f$ as a composition

$$S_w \to S_f \equiv S_w \to S_{di} \to S_d \to \bar{S}_d \tag{21}$$

We can easily check that the new diagram we constructed is a data fusion diagram as in Figure 2. The identity morphism $S_f = \bar{S}_d \to \bar{S}_d$ makes S_d a subclass of S_f according to Definition 1. Therefore the class of decision fusion systems is a subclass of data fusion systems.

For our example, the diagram of Figure 6 takes the form as in Figure 7. As we can see from this diagram, the decision fusion system from our example is also a data fusion system. The fusion function D_f for this system is

$$D_f \equiv \bar{M} \circ (M_1 \times M_2) \tag{22}$$

$$\begin{array}{ccc} (X_1 \times Y_1 \to V_1), & & (X \times Y \to E) \\ (X_2 \times Y_2 \to V_2) & \overset{M_1 \times M_2}{\to} & \times (X \times Y \to E) \\ \searrow\ \bar{D}_f & & \swarrow\ \bar{M} \\ & (X \times Y \to E) & \end{array}$$

Figure 7: Example: Derivation of Data Fusion Function from Decision Fusion Function

6 Conclusion

This paper has two goals. The first one is to show a definition of "information fusion" in a formal framework. To achieve this goal we put the problem of information fusion in the category theoretical framework. More specifically, we used the category *Spec* in which category objects (nodes) are specifications of software systems (algorithms) and the category arrows are morphisms between the specifications. An information fusion system (or its specification) is then represented as a diagram consisting of such nodes and arrows. First,

we showed a diagram for a data fusion system and then a diagram for a decision fusion system. We used a simple example of edge detection to explain the main concepts of this representation. More precisely, the goal of the fusion system was to derive edge edges (more precisely *edge points*).

The second goal was to show that within this formalization one can carry out formal reasoning about information fusion systems. Towards this goal, we formulated a theorem saying that decision fusion is a special case of data fusion. We showed the meaning of such a theorem, outlined a proof of the theorem and finally showed an example of a construction that takes a decision fusion system as input and produces a data fusion system. This result does not seem either surprising or difficult to show, for instance by example. In this paper, however, we were able to show that the subclass relation holds using rigorous formal approach. After all, it is not so obvious that this fact is true; only after you see such a proof does this become obvious.

Essentially, as one can see from the paper, the constructed data fusion system has exactly the same behavior as the decision fusion system used in the construction. As such, this construction does not seem to have a high practical value. Note, however, that we compared the two design solutions from only one point of view - the function of the system. There are many other points of view and many other reasons for using the decision fusion approach over data fusion. One of such reasons might be the computational complexity of the fusion system. If the response time of the fusion system is critical, and the computational complexity of the decision fusion solution is lower than that of data fusion, the choice of decision fusion is fully justified.

It is worthwhile to note that not necessarily everybody will agree with the definitions of and distinctions among such concepts as data fusion and decision fusion, as presented in this paper. However, since these concepts were presented in a language with formal semantics, we can truly understand the meaning of such definitions. Consequently, if one uses different definitions for these two concepts, at least one can clearly understand what we meant. This is perhaps the most important aspect of formal methods.

References

[1] Specware: User guide, version 2.0.3. Technical report, Kestrel Institute, 1998.

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

[3] E. R. Dougherty and C. R. Giardina. *Matrix Structured Image Processing*. Prentice-Hall, 1987.

[4] M. M. Kokar, J. A. Tomasik, and J. Weyman. A formal approach to information fusion. In *Proceedings of the Second International Conference on Information Fusion, Vol. 1*, pages 133–140, 1999.

[5] M. M. Kokar, J. A. Tomasik, and J. Weyman. Formalization of the information fusion problem. *In preparation for submission*, 2001.

[6] J. S. Lim. *Two-Dimensional Signal and Image Processing*. Prentice-Hall, 1990.

[7] B. C. Pierce. *Basic Category Theory for Computer Scientists*. MIT Press, 1991.

[8] Y. V. Srinivas. Category theory: Definitions and examples. Technical Report TR-90-14, University of California at Irvine, 1990.

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