Analytical Analysis of Data and Decision Fusion in Sensor Networks

Zille Huma Kamal, Mohammad Ali Salahuddin, Ajay Gupta, Mark Terwilliger, Vijay Bhuse, Benjamin Beckmann
Computer Science Department, Western Michigan University Kalamazoo, MI, USA

Abstract - MEMS technology has improved such that the capabilities of large sensor devices can now be encompassed in devices that are the size of a penny. These resource constraint devices are confronted with the challenges of ensuring accuracy of observations while conserving power resources. This paper analyzes the two modes of collaborative signal processing employed in sensor networks, namely, data fusion and decision fusion, to achieve greater accuracy. A breakdown of the computation and communication complexity is presented for each fundamental step of data and decision fusion processes, to help in identifying features for possible optimization. We show that a tradeoff exists between the two approaches depending on the underlying application domain.

Keywords: fusion, collaborative signal processing and classification.

1.0 Introduction

With the improvement in technology, engineers have been successful in reducing the size of sensors and actuators. Processing and communication-enabling units have been integrated with these devices, making these tiny sensor/actuators smart devices, also known as, smart sensors. These smart sensors are inexpensive making them one of the most useful cheap devices. Their use has been envisioned in military, environmental and everyday applications. For example, a military application envisages a network of these smart devices to detect, classify and track an enemy target. A commercial application envisages a network of smart sensors to optimize farming by controlling fertilization and watering.

Power resources available for computation and communication have also been reduced, with the reduction in the size of these devices, affecting the accuracy, utilization and lifetime of these devices. Imagine trying to replace the batteries of the smart sensors, which are used to monitor the enemy lines in a battlefield or maintain the sensor networks in oceans. Due to these limitations of the smart sensors, researchers have emphasized the need for some form of collaborative signal processing to improve the accuracy of measurements by incorporating in-network processing within a sensor network without extensively consuming the power resources.

Collaborative Signal Processing (CSP) is one of the most powerful techniques to improve accuracy and reliability of sensed readings of a network of wireless devices, while increasing the lifetime of the network [1]. If a network of smart sensors can be programmed to collaborate then this network would be one of the most inexpensive monitoring, controlling and tracking systems available to date. Collaborative Signal Processing can be further divided into data fusion and decision fusion. These two fusion methods achieve collaboration through either limited computation or limited communication. Decision fusion, takes a toll on power resources for computation while data fusion consumes power for communication. It is critical to decide which form of CSP is beneficial in conserving the power resources of the network, while achieving accurate results.

This paper has been written with the objective of exploring CSP analytically. To explore CSP we must identify the components of CSP and the feasibility of migrating this technology to wireless sensor network, in terms of computational and communicational complexity. Also, emphasizing the significance of CSP might prompt hardware and software designers to optimize wireless sensor networks by integrating collaborative signal processing on the processor board. The rest of the paper is organized as follows. In section 2 we give brief details of smart sensor hardware commonly used today. This section also details an example target detection and tracking application suitable for CSP. The general CSP algorithm is discussed in section 3. Three different phases employed in CSP namely, training, testing and deployment are detailed in section 4. Section 5 analyzes CSP empirically based on the analytical analysis presented in sections 3 and 4. Finally, we give some conclusions and future work in section 6.

2.0 Smart sensor hardware and a target detection and tracking application

There are various kinds of tiny smart sensors including, UC Berkeley's Mica Motes, MIT's μAMPS (μ-Adaptive Multi-domain Power aware Sensors), WINS at Sensoria, etc. Let us consider the computation and communication capabilities of the Mica Motes (commonly known as Motes). Motes are $2.25 \times 1.25 \times 0.25$ inches in dimension and equipped with an Atmega 128L processor, a 128 KB flash memory, a 4 KB EEPROM, and a 916MHz (options are available for 315MHz and 433MHz) radio all powered by 2× AA batteries. Motes can be easily interfaced via a 51-pin connector to a variety of sensor boards. Motes can be programmed using TinyOS to form self-configuring sensor networks. In the rest of the paper, without loss of generality, we will refer to tiny smart sensors as motes.

We now delineate a target detection and tracking application. Assume that a drone drops the motes that are dust size (see Smart Dust project [2]) over a region. We divide the region to be observed into sectors. We refer to sensing features, such as, temperature, light, seismic, acoustic etc. as modalities. When the readings of certain modalities exceed a predefined threshold value, the motes in that sector are said to have detected an object. All motes that detected an object then classify the object into one of the various predetermined categories, while motes in all other sectors remain idle to conserve energy. For example, when a moving object is detected in a sector, the network of motes try to classify whether the object is an enemy armored vehicle, a civilian vehicle or an animal. If the detected object is of interest it is tracked through the region. To track the object, active motes alert idle motes, when the target is headed towards their sector. The classification phase is the most computationally intensive phase. Therefore, it would be beneficial to study the prospects of introducing CSP in this phase of the target detection and tracking application.

3.0 Collaborative Signal Processing

This section delineates the initial scenario (also known as the activation stage) for CSP and the operational characteristics of the two common modes of collaborative signal processing, namely data fusion and decision fusion. The motes have detected an object, when a given characteristic in the environment changes. For example, the motes could continuously monitor the temperature of a region and when that temperature rises above a certain threshold, an object has been detected. When this happens, an event is said to be triggered, which will cause the network of motes to move from the detection phase into the classification phase. Since this is the most computationally intensive phase of the application we introduce CSP here. We will now discuss the operational characteristics of the two modes of CSP.

In data fusion, motes collect data, that is, signal readings that are pertinent to certain modalities or features of the predetermined categories. The motes then transmit this data to a pre-elected manager mote in the network. This manager mote then aggregates the data received from all participating motes in the network and then classifies the object into one of the predetermined categories.

In decision fusion, the mote collects the data and processes the data locally to classify the object into one of the predetermined categories. These motes then transmit the results of their classification to an elected manager mote in the network. The manager mote aggregates the independent classification results from each of the motes and makes a final global decision as to which category the object should be classified. The manager mote's decision is propagated through the network and depending upon whether the object is of interest, the object is tracked.

4.0 Phases of Collaborative Signal Processing

This section gives background information on various phases of CSP based on previous work [1]. Typically, data and decision fusion in CSP can be decomposed into three phases:

- The training phase – where each mote is exposed to a number of sample signal readings belonging to each of the predetermined categories. Based on this training every mote computes a mean and a covariance for each of the categories.
- The testing phase – where each mote is exposed to a number of different test signal readings. These signal reading are drawn from known modalities of the categories. Motes are then prompted to classify these test signals into one of the predetermined categories, using the mean and covariance for each category computed in the training phase. The frequency of the

correctness of these decisions (and erroneous decisions) is maintained in a table known as the *confusion matrix*. Based on this confusion matrix, statistics such as the prior probability, false alarm rate, etc. can be inferred.

- The deployment phase – where each mote is deployed in the region of interest and is capable of detecting an object. Classification of the object is either made locally or globally, depending on the form of CSP (data or decision fusion), and if the object is of interest it is tracked in the region.

This section lists the notations used throughout the rest of the paper. When an event is triggered, an f x d dimensional matrix, known as an event feature matrix, is generated at the detecting mote. A feature refers to a modality used to distinguish categories. The rows of the event feature matrix represent features that are sensed and the columns represent the dimensions of the temporal processing, such as FFT, conducted on any given feature. Table 1 lists the other parameters associated with CSP.

Table 1: Notation of environment variables used for data and decision fusion analysis.

Notation	Meaning
n_0	Number of motes in system
k	Number of predetermined categories
N	Number of samples during training phase for each category at each mote
$\mu_{I,j}$	Mean at mote i for category j, where $1 \le i \le n_0$ and $1 \le j \le k$
$S_{i,j}$	Covariance at mote i and category j where $1 \le i \le n_0$ and $1 \le j \le k$
f	Number of features sensed
d	Number of points used for temporal/spectral processing of signal.
$M_{i,j}^{\alpha}$	Event feature matrix at mote i for category j during the training phase
$M_{i,j}^{\beta}$	Event feature matrix at mote i for category j during the testing phase
$M_{i,j}^{\chi}$	Event feature matrix at mote i for category j during the deployment phase

We next give a detailed analysis of the computational and communicational complexities involved in each phase of data and decision fusion in collaborative signal processing.

4.1 Training Phase

During the training phase, each mote takes N measurements when exposed to each of the k predetermined categories and computes a mean, $\mu_{i,j}$, and covariance, $S_{i,j}$, for each category as in 1 and 2.

$$\mu_{i,j} = \frac{1}{N}\sum_{\alpha=1}^{N} M_{i,j}^{\alpha} \tag{1}$$

$$S_{i,j} = \frac{1}{N-1}\sum_{\alpha=1}^{N} (M_{i,j}^{\alpha} - \mu_{i,j})(M_{i,j}^{\alpha} - \mu_{i,j})^{T} \tag{2}$$

where, $1 \le i \le n_0$, $1 \le j \le k$ and Y^T represents the transpose of matrix Y.

Assuming that the FFT/temporal processing of a signal is done in hardware, computation of the mean for a category at a mote requires $(N + 1)fd$ local operations resulting in an f x d dimensional mean matrix $\mu_{i,j}$. Computation of the covariance at each mote requires $N\,(2f^2d - f^2 + 2fd) + f^2$ local operations resulting in an f x f dimensional covariance matrix, $S_{i,j}$. The computation of the inverse of the covariance matrix, $S_{i,j}^{-1}$ which is needed in the later phases requires on of the order of $2f^3 + O(f^2)$ local operations [3]. Note that the off-diagonal entries of the covariance matrix for each category represents the covariance between different components of the signals generated. The diagonal entries of the matrix represent the variance within each component of the signal [4].

Based on the training phase, we can represent each of the k categories by a mean $\mu_{i,j}$ and covariance $S_{i,j}$ at each mote i, for $1 \le i \le n_0$ and $1 \le j \le k$. Typically each category can be assumed to be represented by a Gaussian distribution, with mean $\mu_{i,j}$ and covariance $S_{i,j}$ at each mote. A Gaussian distribution is an extension to the Normal distribution (which all continuous variables have [5]) for a multi-dimensional system. We can also assume that after the training phase, the inverse and determinant of the covariance matrix, $S_{i,j}^{-1}$ and $|S_{i,j}|$ respectively, are computed and stored for each category at each mote. These calculations are made and stored to avoid redundant, repetitive and expensive computations needed in later stages. Figure 1 illustrates the phases of CSP.

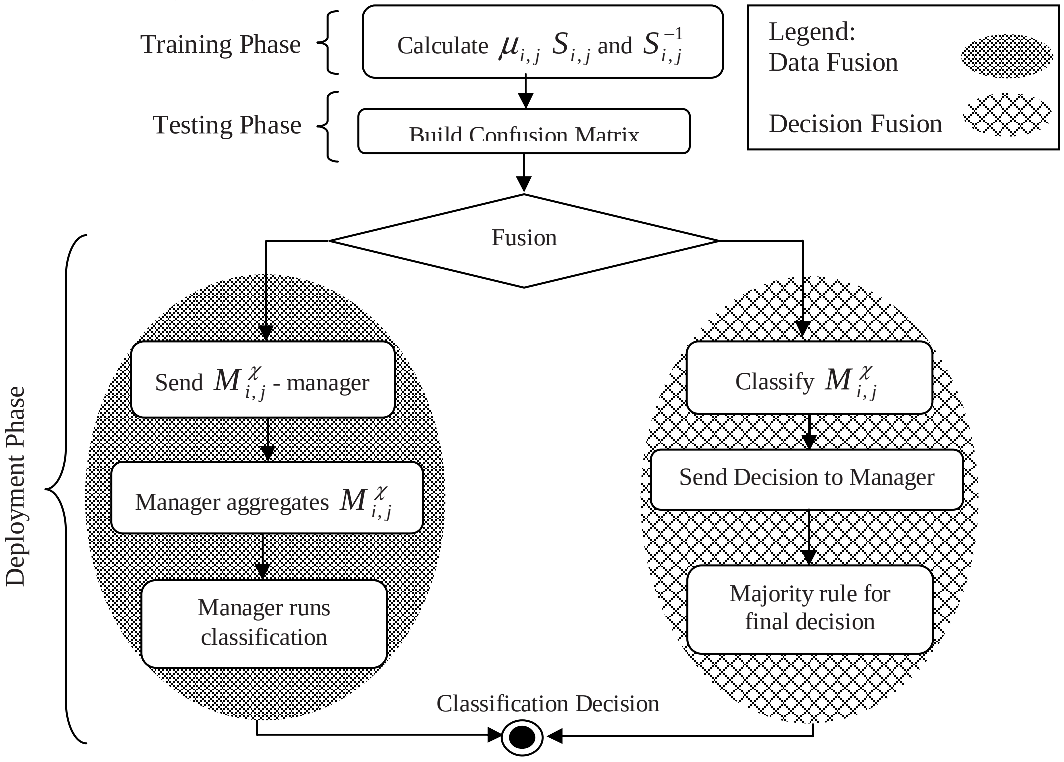

Figure 1: Phases of CSP

4.2 Testing Phase

In the testing phase, we observe the frequency of correct and incorrect classification by the motes. During the testing phase, we classify the $M_{i,j}^{\beta}$ event feature matrix into one of the *k* categories and maintain a confusion matrix, which records the result of classification decisions at each mote. Table 2, indicates a sample confusion matrix, where entry *x*,*y* of the confusion matrix indicates the number of measurements classified into category *y* but actually belonging to category *x*.

Table 2 illustrates an example of a confusion matrix.

Classified True	1	2	…	*k*
1	ϕ'_{11}	ϕ'_{12}		ϕ'_{1k}
2				
…				
J	ϕ'_{j1}	ϕ'_{j2}		ϕ'_{jk}
…				
K			…	ϕ'_{kk}

As an example, if entry *x,y* of the confusion matrix is ϕ'_{xy} then the probability of detection for category *j*, such that *1* ≤ *j* ≤ *k* can be computed as $P(C_j) = \dfrac{\phi'_{jj}}{\sum_{\mathrm{m=1}}^{\mathrm{k}} \phi'_{\mathrm{jm}}}$ [1].

4.2.1 Classification

One approach to classification is to use the Maximum A Posterior (MAP) classifier, which maps the observed data into one of the *k* categories such that the probability of misclassification (false positive) is minimized [1]. We know that the posterior probability is formulated as

$$P(C_j \mid M^{\beta}_{i,j}) = \frac{P(C_j)P(M^{\beta}_{i,j} \mid C_j)}{P(M^{\beta}_{i,j})} \tag{3}$$

where, $1 \le j \le k$, $1 \le i \le n_0$, $P(C_j \mid M^{\beta}_{i,j})$ is the posterior probability, $P(M^{\beta}_{i,j} \mid C_j)$ is the likelihood, $P(M^{\beta}_{i,j})$ is the probability of occurrence and P (C_j) is the prior probability.

The probability $P(M^{\beta}_{i,j})$ can be assumed to be a uniform distribution such that all C_j values in the region of interest are equally likely. We also assume the likelihood density to be Gaussian and can be represented as the probability density function of a d - dimensional Gaussian random matrix defined as

$$P(M^{\beta}_{i,j} \mid C_j) = \frac{1}{(2\pi)^d \mid S_{i,j} \mid^{1/2}} e^{\left[-\frac{1}{2}(M^{\beta}_{i,j} - \mu_{i,j})^T S^{-1}_{i,j} (M^{\beta}_{i,j} - \mu_{i;j})\right]} \tag{4}$$

where, T denotes the transpose of the matrix and d is the dimension of a event feature matrix.

If $\boldsymbol{C} = P\left(M^{\beta}_{i,j} \mid C_j\right)$, equation 4 can be simplified to $\left(-\frac{1}{2}\left|\left(M^{\beta}_{i,j} - \mu_{i,j}\right)^T S^{-1}_{i,j}\left(M^{\beta}_{i,j} - \mu_{i,j}\right)\right|\right) = C'$

where, C' and $\boldsymbol{C}$ are constants related in an obvious way.

The simplified equation 4, has the properties of a distance function between the matrix $M^{\beta}_{i,j}$ and the mean $\mu_{i,j}$ and is called the Mahalanobis Distance. This implies that the posterior probability is proportional to the likelihood probability, $P(C_j \middle| M^{\beta}_{i,j}) \propto P(M^{\beta}_{i,j} \middle| C_j)$, and we have shown that the likelihood probability can be reduced to a quadratic distance function. So, MAP classifier can now classify an event to a category for which it has minimum mahalanobis distance [6]. This is reflected in equation 5.

$$(\ln(P(C_j \mid M^{\beta}_{i,j})) \propto \left[-\frac{1}{2}(M^{\beta}_{i,j} - \mu_{i,j})^T S^{-1}_{i,j} (M^{\beta}_{i,j} - \mu_{i,j})\right] = MAP^{\beta}_{i,j} \tag{5}$$

where, $MAP^{\beta}_{i,j}$ is the result of the quadratic distance function, which is a d x d dimensional matrix, and ln represents natural logarithm.

Using the Frobenius norm [7] we can normalize the result of the distance function so that we can classify the measurement at mote i into category j such that $\left\|MAP^{\beta}_{i,m}\right\|$ is maximum over $\left\|MAP^{\beta}_{i,j}\right\|$ for $1 \le j \le k$. Assuming the transpose of a matrix and the logarithms are performed in hardware. The computation of the a posteriori probability requires $2df^2 + 2d^2f - d^2 - df + 2f^2 + 5$ local operations.

4.3 Deployment Phase

After the testing phase, the motes are deployed in the region to be monitored. Each mote records the measurement $M^{\chi}_{i,j}$ and then a data or decision fusion starts.

4.3.1 Decision Fusion

There are two ways for motes to collaborate their information, via decision fusion or data fusion, as introduced earlier. In decision fusion, each mote runs the MAP classifier, using the observed $M^{\chi}_{i,j}$, and transmits its decision to a pre-elected manager mote. This manager mote then aggregates the decisions received from the other $(n_0 - 1)$ motes and classifies the object into a category, using a majority rule or other statistical techniques. In the simplest form, we can assume that the aggregation takes place at the manager mote after decisions have been received by it. In smart forms of aggregation, aggregation takes place along the path decision results traverse to the manager mote but discussion and analysis of these types of aggregations is beyond the scope of this paper.

Running the MAP classifier for a category requires $2df^2 + 2d^2f - d^2 - df + 2f^2 + 5$ local operations, therefore total local computation at a mote is $(2df^2 + 2d^2f - d^2 - df + 2f^2 + 5)k$ for all categories. Assuming d is much greater than f, local computational complexity at each mote is $O(d^2fk)$. If the majority rule was used for pooling the decisions at the manager mote, then the detected object would be assigned the category with the greatest number of ‘votes’. Total computation cost of the majority rule is $O(n_0)$ at the manager mote. The total computation cost of decision fusion on the network is $O(d^2fkn_0)$. If we assume that the cost of communicating an integer over the wireless radio medium is constant, then the communicational complexity would be the cost of communicating $(n_0\text{-}1)$ messages from all motes to the manager mote. Consequently, a broadcast message is sent from the manager mote to all other motes informing them of the decision. Therefore, the total cost of communication on the network is $O(n_0)$.

4.3.2 Data Fusion

Prior to deployment of motes, the designated manager mote computes and stores an aggregated mean and covariance matrix for each category over all n_0 motes. In data fusion, $n_0 - 1$ motes will transmit $M_{i,j}^{\chi}$ to the manager mote, which can aggregate this information by using simple statistical averaging. The manager mote is then responsible for running the MAP classifier on the aggregated event feature matrix using the aggregated mean and covariance matrices for each category to classify the detected object into one of the predetermined k categories.

We know that running the MAP classifier at the manager mote requires $2df^2 + 2d^2f - d^2 - df + 2f^2 + 5$ local operations for each category. The cost of aggregating the n_0 matrices requires n_0fd local operations by simple averaging techniques. In practice, d is much larger than f. Therefore, total computational complexity of data fusion at the manager mote is $(2df^2 + 2d^2f - d^2 - df + 2f^2 + 5)k + n_0fd$, which can be represented as $O(d^2fk) + n_0fd$. The communication complexity is n_ofd/w, where w is the size of a message. (The network parameters such as packet queuing delays, packet loss, congestion, etc are not incorporated in our analysis, as they are not a dominating factor. Furthermore, we are interested in asymptotic analysis depending upon the major varying parameters of CSP.) The cost of data fusion at each mote is just the cost of sampling which is $O(fd)$. Therefore the total computational cost is $O(n_ofd + d^2fk)$ and the communicational cost is n_ofd/w on the network.

5.0 Analysis

The preceding sections have delineated the concepts behind collaborative signal processing and its modes of fusion. In this section we attempt to simulate a sensor network environment and identify the factors influencing CSP. For analysis purposes we make the following assumptions:

1. The cost of decision fusion at the manager mote is some constant times the cost of the number of motes participating and sending decisions to the manager mote.
2. The cost of data fusion at each mote is some constant times the number of features that they have to sense.
3. If the number of motes is large, multi-hop routing algorithms are needed for motes to communicate with the manager mote. However, for simplicity, we ignore these additional costs.

Based on our analysis we simulate the computational cost of data and decision fusion on motes in the network and on the manager mote, figure 2 depicts our results.

It can be observed from the graph that in data fusion the manager mote bears the burden of computation whereas, in decision fusion all motes participating in the classification phase carry out computations in similar magnitudes to that of the manager mote in data fusion. It is also obvious that the cost of decision fusion at the manager mote is proportional to the number of motes participating in the classification phase. The total cost of computation on the network tends to increase faster in decision fusion than in data fusion. We can infer from figure 5 that when the number of motes is small (say in the order of tens), total cost of data and decision fusions are comparable. For larger number of motes, the total cost of decision fusion quickly far exceeds the total cost of data fusion. It is natural to expect that data fusion facilitates better classification as the manager mote can use the measurements of all the motes to make a better decision by using the spatial information, whereas, in decision fusion each mote makes a decision oblivious of the measurements at other neighboring motes. The excessive cost of communication in data fusion makes the process unlikely to be used in practice for large dense sensor network deployments. Most envisioned applications of sensor networks utilize dense deployments of motes. For a sparse sensor network, the same argument of cost versus accurate classification results would hold. Hence, some forms of integrated techniques, wherein localized data fusion within clusters are followed by inter-cluster decision fusion, appear to show promise for large-scale deployments of sensor networks. We have obtained preliminary results in this area and currently work to improve and formalize CSP.

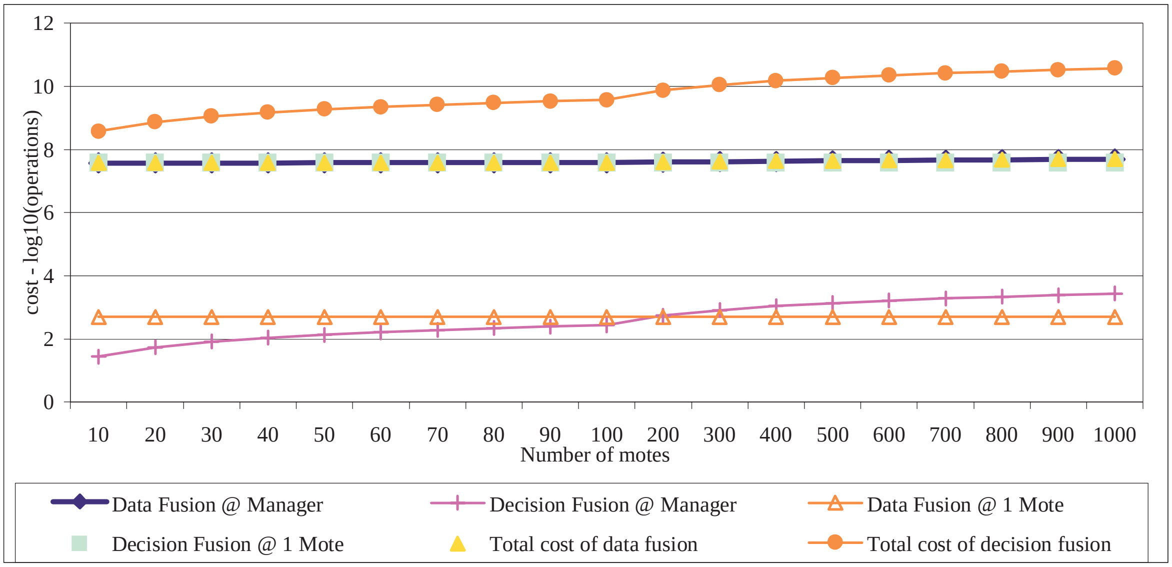

Figure 2: Illustrates how cost of the two fusion processes vary with respect to the number of motes participating in the fusion process both at each individual mote and at the manager mote.

6.0 Conclusion

This paper contributes to the field of wireless sensor networks by recapturing the importance of collaborative information processing in resource constraint environments. We introduced the influential factors in such processes so that future software and hardware designers can incorporate necessary means (possibly hardware solutions) to accommodate collaborative information processing in sensor networks. We have shown that the computational costs of data and decision fusion vary greatly when the number of motes participating in the network change. However, communication analysis indicates that decision fusion outperforms data fusion.

Presently, it is the developer’s burden to design and implement applications, which are efficient in energy preservation, resources and produce accurate results. However, being aware of the influential factors of collaborative signal processing, we can gain insight to better design our hardware and/or software to facilitate this energy and resource conserving fusion process, while increasing accuracy of results computed within the sensor network.

Our future directions involve the development of an adaptive collaborative signal processing environment that will adapt between the two fusion processes depending on the number of motes and the degree of accuracy required. We also hope that our contributions are taken into account for future hardware and software designs so that collaborative signal processing can be efficiently embedded in sensor networks.

7.0 Acknowledgements

Research is supported in part by the National Science Foundation, under grants ACI-0000442, ACI-0203776, and MRI- 0215356, by the Department of Education grant R215K020362, and a Congressional Award, administered by the US Department of Education, Fund for the Improvement of Education. The authors would also like to acknowledge Western Michigan University for its support and contributions to the Wireless Sensor Network Research Laboratory, Computational Science Center and Information Technology and Image Analysis (ITIA) Center. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding agencies or institutions.

8.0 References

[1] D. Li, K.D Wong, Y.H Hu and A.M Sayeed, "*Detection, Classification and Tracking in Distributed Sensor Networks*," IEEE: Signal Processing Magazine, March 2002.

[2] Warneke, M. Last, B. Leibowitz, K.S.J. Pister, "Smart Dust: Communicating with a Cubic-Millimeter Computer", IEEE: Computer Magazine, Jan. 2001, pp 44-51.

[3] C. Meesookho, S. Narayanan and C.S Raghavendra, "*Collaborative Classification Application In Sensor Networks*," 2nd IEEE Sensor Array and Multichannel Signal Processing Workshop, Rosslyn, VA, Aug 2002.

[4] Wikipedia, "*Covariance Matrix*," www.wikipedia.org, July 2003.

[5] Schwarz C. J. "*Stat-301: Statistics for the life sciences - 98-3 term*," Department of Statistics and Mathematics, Simon Fraser University

[6] R. O. Duda, "*Pattern Recognition For HCI*," Department of Electrical Engineering, San Jose State University, 1997.

[7] T.K. Moon, W.C Stirling, "*Mathematical Methods and Algorithms for Signal Processing*," Prentice Hall, NJ, NY, 2000.