Energy-Efficient Information-Driven Coverage for Physical Movement Monitoring in Body Sensor Networks

Hassan Ghasemzadeh, *Student Member, IEEE,* Eric Guenterberg, *Student Member, IEEE,* and Roozbeh Jafari, *Member, IEEE*

***Abstract*—Advances in technology have led to the development of various light-weight sensor devices that can be woven into the physical environment of our daily lives. Such systems enable on-body and mobile health-care monitoring. Our interest particularly lies in the area of movement-monitoring platforms that operate with inertial sensors. In this paper, we introduce the notion of compatibility graphs and describe how they can be utilized for power optimization. We first formulate an action coverage problem that will consider the sensing coverage from a collaborative signal processing perspective. Our solution is capable of eliminating redundant sensor nodes while maintaining the quality of service. The problem we outline can be transformed into an NP-hard problem. Therefore, we propose an ILP formulation to attain a lower bound on the solution and a fast greedy technique. Moreover, we present a system for dynamically activating and deactivating sensor nodes in real time. We then use our graph representation to develop an efficient formulation for maximum lifetime. This formulation provides sufficient information for finding activation duties for each sensor node. Finally, we demonstrate the effectiveness of our techniques on data collected from several subjects.**

***Index Terms*—Physical Movement Monitoring, Body Sensor Networks, Power Optimization, Information Coverage, Signal Processing, Classification.**

I. Introduction

A LARGE number of patients require long-term care and fall victim to a pervasive lifestyle of constant monitoring in pursuit of optimal treatment. Examples of such patients include those recovering from operations, those undergoing rehabilitation, and the elderly. Physicians currently depend on self-reporting to determine patients' activity levels, including amount of time spent walking, sleep schedules, etc. However, recent advances in sensor and computer technology allow patients to wear several small sensors with embedded processors and radios. Collectively, these sensors form a body sensor network (BSN). BSNs have the ability to diagnose critical events such as heart attacks or heart failure and monitor activity by recording the duration, quality and type of movement performed. This data can be more accurate and comprehensive than self-reporting.

An important goal in designing BSNs is to minimize power consumption while preserving an acceptable quality of service. Patients will be expected to charge the sensors or replace the batteries on a regular basis, as they do with cell phones and other electronics. However, the frequent need to charge and the bulk of the battery can frustrate the users, causing them to no longer wear the sensors. Furthermore, batteries are the heaviest component in the system. By decreasing power usage, the size and weight of each sensor node can decrease, thus increasing patient comfort and device wearability. Deactivating unnecessary sensor nodes is a simple and highly effective method of power reduction, but the method of determining which nodes to deactivate depends greatly on the function of the sensor network.

Our pilot application of physical movement monitoring can be used for rehabilitation, sports medicine, geriatric care, and gait analysis. Movement monitoring uses several sensor nodes to distinguish between different types of movements, such as walking, standing up, sitting down, lying down, and kneeling. Typically, the sensor units are homogenous and use accelerometers and gyroscopes to classify human actions. Some movements, such as walking, can be easily determined from almost any location on the body, whereas detecting a leg-raise would specifically require a leg sensor, and differentiating between falling, sitting down, and lying down may require several nodes.

Current methodologies for discerning active nodes tend to be designed for sensor coverage over a large area or incremental diagnosis. They are either overly complicated or inadequate when used to monitor physical movement. This paper introduces a graph model of local knowledge provided by each node and shows how this model can be used to address energy requirements of our distributed system. A new power optimization technique which we call *action coverage* is presented. The objective is to select the smallest number of sensor nodes that can adequately distinguish among all expected activities. This selection can be altered dynamically to disperse power load, route around a failed node, or cover a diverse set of activities. As our experiments will show, by limiting our interest to upper or lower-body movements, we can reduce the number of sensors required for the set of actions to one. To cover all body parts, at least four sensor nodes are required. Moreover, we show the capability of the model to maximize system lifetime while preserving the global knowledge provided by the system.

We introduce compatibility graphs that simplify the visualization of the problem and lead directly to an algorithm for determining the minimum size set for action coverage.

Manuscript received 15 January 2008; revised 28 September 2008.

H. Ghasemzadeh, E. Guenterberg and R. Jafari are with the Department of Electrical Engineering, University of Texas at Dallas, Richardson, TX, 75080 USA (e-mail: h.ghasemzadeh@student.utdallas.edu).

Digital Object Identifier 10.1109/JSAC.2009.090107.

Because the problem is NP-hard, we formulate an ILP that attempts to find a lower bound on solutions. We also provide a quick heuristic algorithm that provides a reasonable approximation. Finally, we present experimental verification of these techniques for both coverage and lifetime problems.

II. Related Work

Energy efficiency is one of the most important performance metrics in BSNs as the system lifetime is directly affected by the power consumed by each sensor node. *Coverage*, the measure of quality of service provided by a sensor node, is one of the fundamental issues in designing wireless sensor networks. While interpretation of the coverage problem varies subject to the application area of the sensor network, the ultimate goal is to develop robust algorithms for power optimization [1]. The most studied coverage problems include geographical-based approaches such as *area coverage* and *point coverage*. In area coverage, the objective is to monitor a certain area in space while maintaining quality of service. It requires an effective technique to determine active and idle sensor nodes over time. In point coverage, however, the objective is to monitor only a set of points in space.

A great deal of work has been done to minimize the number of homogeneous nodes covering a geographical area. The authors of [2] describe a method of forming disjoint sets of sensor nodes such that every set is capable of monitoring the area. The area is divided into several fields, and the field covered by the fewest nodes is referred to as critical. The algorithm then selects the nodes that cover the critical elements. Three approximation algorithms for the *set k-cover* problem are presented in [3]. In these algorithms, the objective is to partition sensor nodes into covers such that the number of covers that monitor the target area is maximized. Another area coverage mechanism is presented in [4], in which each node continuously makes decisions to activate or deactivate itself using information from its neighbors. A sensor becomes inactive if it discovers that its neighbors can effectively monitor its area. The authors of [5] model the problem as disjoint sets in an undirected graph, where sensors correspond to vertices and an edge represents two sensor nodes that are within close proximity. A graph-coloring mechanism finds the minimum number of active nodes. The authors of [6] investigate the coverage problem for target-tracking applications. They show that by transforming the problem into a disk-coverage problem, the number of active sensors required for satisfactory localization would be four times the amount for detection applications. A distributed coverage algorithm for target-detection applications, called *Co-Grid*, is presented in [7]. This technique considers the network as several coordinating fusion groups located on overlapping virtual grids. A probabilistic model is used to maintain area coverage based on information derived from minimum event detection probability and system false alarm rate from active nodes. In [8], the authors model the coverage problem as a decision problem to decide whether each location in the area is adequately monitored or not. According to the sensing range of each sensor node, which can be viewed as a unit disk or a non-unit disk, two different decision problems can be addressed. In another area coverage technique presented in [9], the authors define sensing regions according to detection constraints. Thus, the sensing regions are not necessarily disks around nodes. The overall coverage can be increased by forcing collaborative signal processing among sensor nodes.

In the point coverage technique presented in [9], sensor nodes are organized into several set covers that can be successively activated. At each time, only one of the set covers is considered as the active set, which further indicates the sensor nodes that are responsible for monitoring targets and transmitting data. In another point coverage algorithm introduced in [10], the set of sensors is divided into disjoint sets such that every set completely covers all targets. The disjoint sets are modeled as disjoint set covers and the problem is shown to be NP-Complete. Any polynomial-time approximation algorithm has a lower bound of two. The authors of [11] introduce the notion of *information coverage* to further reduce the number of active nodes in the area coverage problem. The technique is characterized based on a parameter estimation approach where the cooperation of sensor nodes is considered. Several other coverage techniques can be found in [12], [13], [14].

Certain distributed tracking systems employ a method of utilizing collaborative signal processing to determine which sensors must be initiated. An information-driven sensor collaboration technique proposed in [15] decides which node is most appropriate to perform the sensing. Such tracking approaches often attempt to estimate the future position of a target, given its past and present positions.

The above techniques use the sensing range of each sensor node to minimize the number of sensors completely covering a geographical space. Such area-based approaches are not necessarily effective for physical movement monitoring and BSNs. In this case, complete coverage of the body is not necessary; it is simply a reliable indication of which of the body's actions are important. Furthermore, the technique of sequentially activating sensors employed in tracking systems does not apply to physical movement monitoring systems. This is because actions such as standing, walking, or kneeling are relatively short and the key identifying features may occur early in the movement. Therefore, it is essential to activate all the required sensors before the action occurs.

While the coverage problem generally plays a great role in reducing power consumption in wireless sensor networks, collaborative signal processing techniques must also be used to constrain the energy consumed due to high-volume communication. This power reduction strategy involves decreasing the communication overhead for classification. The classifier combination itself can be performed according to several schemes [16]. To achieve a significant data association in a distributed system, each sensor node must associate its local measurements with individual targets in a detection or classification application [17]. With this technique, each sensor node will individually perform a preliminary classification and send the result to a central node identified as the "master" node. The master can combine the results for a final classification. A significant technique presented in [18] is boosting, in which each individual classifier is re-sampled and the majority of votes are used to combine the results. AdaBoost [19] is another decision combiner that uses a weighted voting

Fig. 1. **A node with inertial sensors**

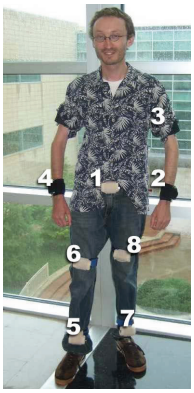

Fig. 2. **Experimental subject wearing eight sensor nodes.**

scheme to make a global decision. It combines a set of hypothesis through weighted majority voting on the classes predicted by each hypotheses. Another method for classifier combination is presented in [20] where bayesian averaging and boosting techniques are combined to find conjunctive feature combinations that achieve high classification accuracy. These collaborative classifiers were designed to be executed on a single system, and therefore do not consider communications overhead. Authors in [21] propose a distributed classification system for wireless sensor networks. In their system, hard decisions made by individual nodes are communicated over noisy links to a coordinator node which optimally combines local results to make a final decision. A vision-based object detection approach presented in [22] constructs a classifier by selecting only a small number of significant features using AdaBoost. It also presents a method for combining successively complex classifiers in a cascade structure to accelerate the detection speed. Some other distributed classification/tracking algorithms can be studied through [23], [24].

Use of inertial sensors in BSNs is motivated by biomedical applications, and has received much attention during recent years. Authors in [25] introduce a framework for human action recognition using motion sensors. They integrate on-body sensors including accelerometers and gyroscopes in a wireless sensor network to classify physical movements. In [26], authors report the results of a study on activity recognition using different types of sensory devices, including built-in wired sensors, RFID tags, and wireless motion sensors. Authors in [27] use a tri-axial accelerometer mounted on the waist to recognize basic daily movements using a hierarchical classification scheme. A pattern recognition technique for evaluating the performance of the human postural control system using inertial and EMG sensors is presented in [28]. Authors in [29] develop algorithms for recognition of daily activities using five accelerometers placed on the human body. They use a decision tree classifier and achieve 84% classification accuracy. Authors in [30] integrate seven different sensor into a single node to classify twelve movements. The accuracy reported by their system is 90%.

Our research takes a novel approach by combining both classification and coverage. We introduce a generic formulation of coverage problem according to local knowledge provided by different sensor nodes. We employ the results of the classification to reduce the number of active nodes at the decision stage. Moreover, during the classification stage, we demonstrate an approach to further reduce the number of nodes needed to communicate. To the best of our knowledge, this issue has not been investigated previously. For this paper, we exclusively focus on reducing the number of nodes either in static or in real-time. Although the effectiveness of this technique has not yet been analyzed in comparison to the alternatives, it provides sufficient information on the capability of our platform for distributed signal processing.

III. System Architecture and Signal Processing

The pilot application for our research is physical movement monitoring. The purpose of our system is to classify transitional movements into pre-defined actions. Given a set of movements, the system must distinguish between every pair of motions. We capture inertial information of physical movements using motion sensors. Our system consists of several sensor units; each has a tri-axial accelerometer, a bi-axial gyroscope, a microcontroller, and a radio, as shown in Fig. 1. Our accelerometers are LIS3LV02DQ with $1024\ LSb/g$ sensitivity and are used in $2g$ mode for the experiments. We use IDG-300 gyroscopes with $2\ mV/^\circ/s$ sensitivity and $0.014\ ^\circ/s/\sqrt{Hz}$ noise performance. The processing unit of each node, or mote, samples sensor readings at 22Hz and transmits the data wirelessly to a base-station using a TDMA protocol. This sampling rate is experimentally chosen to provide sufficient resolution while compensating for bandwidth constraints of our sensor platform. Our motes, Tmote Sky, are commercially available from moteiv® and are each powered by two AA batteries. The sensor board is custom designed, and the base station is a separate mote connected to a laptop by USB. For our experiments, we arranged eight sensor nodes on our subjects as shown in Fig. 2.

The signal processing and classification is a six step process as shown in Fig. 3.

1) *Sensor data collection:* Data is collected from each of the five sensors on each of the eight sensor nodes at 22Hz.
2) *Preprocessing:* Data is filtered with an eight-point moving average. The number of point used to average the signal is chosen to maintain sharp step response while a smooth output signal can be obtained.
3) *Segmentation:* We determine the portion of the signal that represents a complete action. For experimental purposes, this is done manually to avoid introducing errors by automatic segmentation.

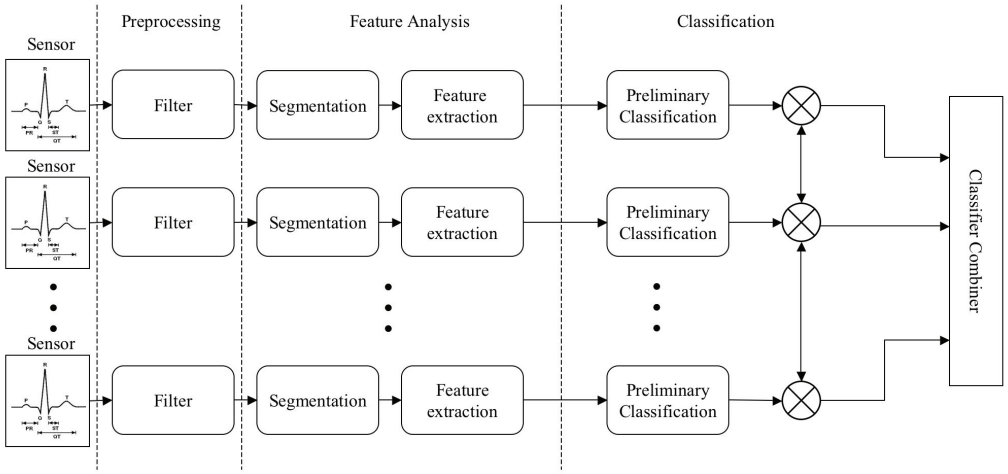

Fig. 3. **Signal processing flow**

TABLE I
FEATURES

No.	Feature	Description
1	Mean	Mean value over the entire segment
2	Start-to-End	Difference between the start and the end points
3	Std	Standard deviation
4	Peak-to-Peak	Difference between the maximum and the minimum values
5	RMS	Root mean square of the segment
6	Median	Signal value separating the higher half of the segment
7	Max	Maximum amplitude value over the entire segment

4) *Feature extraction:* Single-value features are extracted. We transform sensor readings into a set of informative attributes, including the seven features shown in Table I.
5) *Per-node classification:* Each node uses the aforementioned features to determine the most likely action. We use the k-Nearest Neighbor (k-NN) [31] classifier due to its simplicity and scalability.
6) *Final classification:* The final decision can be made using either a data fusion or a decision fusion scheme. In the data fusion, features from all sensor nodes are fed into a central classifier. The classifier then combines the features to form a higher dimensional feature space and classifies movements using the obtained features. In the decision fusion, however, each sensor node makes a local classification and transmits the result to a central classifier where a final decision is made according to the received labels. We use the data fusion scheme for our classification where a central (k-NN) classifier makes a final decision on the current movement occurred in the system considering features from all senor nodes.

We currently process all our data offline in MATLAB. This is convenient for rapid prototyping and algorithm development. However, we have great suspicion that our algorithms for signal processing can be implemented and executed on the motes. In addition, we have yet to develop an approach to automatically segment the data into actions and inactivity. Our simple processing will be performed on the nodes once we finished developing this automated action segmentation [32].

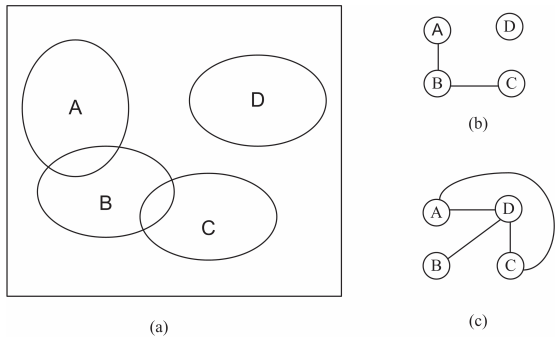

Fig. 4. **Evolving towards a compatibility graph**

IV. PRELIMINARIES

Action coverage refers to how well a system can distinguish between various actions or events. In our system, we have a variety of sensor nodes placed around the body. While detection of all studied movements requires a global view of the system, each individual node in the system has local knowledge of the event taking place. The amount of knowledge presented by each node determines the ability of the node in regards to action recognition. An example is shown in Fig. 4. In Fig. 4a, we show an example of two feature spaces. The ellipses represent classification boundaries. In reality, the shapes are not perfect ellipses. Each node in our system has five data streams (x, y, z acceleration, and x, y angular velocity) and seven features per data stream, forming 35 dimensions per node.

Regions where the ellipses overlap represent potential misclassifications. Any point in the intersection of A and B or B and C cannot be confidently assigned to either class. In Fig. 4b, overlapping vs. well separated classes is translated into a *conflict graph*. The vertices represent classes, and the edges represent ambiguities between the classes. Finally, Fig. 4c, the so-called *compatibility graph*, is generated by complementing the conflict graph of Fig. 4b. If a compatibility graph is not complete, then there exist some movements that the node cannot correctly classify. A complete graph is equivalent to the capability of distinguishing between every pair of classes [33].

One of the most popular class separability measures in the field of pattern recognition is the Bhattacharyya distance [34] [31]. This measure is related to the well-known Chernoff bound and therefore has an explicit expression for a generalized Gaussian distribution. The Transformed Divergence is another common empirical measure of class separability, which is computationally simpler than the Bhattacharyya distance. However, the Bhattacharyya distance is more theoretically sound because it relates directly to the upper bound of the probabilities of classification errors [35]. Both the Transformed Divergence and Bhattacharyya distance measures are real values between 0 and 2, where 0 indicates complete overlap between the signatures of two classes, and 2 indicates a complete separation between the two classes. Both measures are monotonically related to classification accuracies. The larger the separability value, the better the final classification result.

In our experiments, we exploit the Bhattacharyya distance as a measure of separability between pairs of classes. Since we are dealing with such a distribution, we make the Bhattacharyya distance our probabilistic distance. The distance between two distributions i and j is represented by $\beta(i,j)$ in equation 1 where μ_i and Σ_i denote the mean vector and the covariance matrix associated with distribution i, respectively.

$$\beta(i,j) = 2(1 - e^{-\alpha(i,j)})$$

$$\alpha(i,j) = \tfrac{1}{8}(\mu_i - \mu_j)^{'}((\tfrac{\Sigma_i+\Sigma_j}{2})^{-1}(\mu_i - \mu_j) + \tfrac{1}{2}\ln(\frac{\frac{|\Sigma_i|+|\Sigma_j|}{2}}{\sqrt{|\Sigma_i||\Sigma_j|}}) \quad (1)$$

The Bhattacharyya distance is assumed to be directly related to classification accuracy. Also assuming that the Bayes error is approximately equal to the upper bound that is characterized by Bhattacharyya distance, the distance is the lower bound of classification accuracy [36].

V. Action Coverage Formulation

Given a set of sensor nodes $S = \{s_1, s_2, ..., s_n\}$ placed in a body sensor network to detect a set of movements $M = \{1, 2, ..., m\}$, this section will provide a formal definition of the action coverage problem.

Definition 1. Two movements j_1 and j_2 are said to be *compatible* if they have complete separability based on Bhattacharyya metric indicated by equation (1).

Definition 2. A compatibility graph is an undirected graph $G_i = (V, E_i)$ constructed for a sensor node s_i, where V is a set of vertices identical to the set of movements M, and E_i is a set of undirected edges such that edge $(u, v) \in E_i$ if movements u and v are compatible at node s_i.

As can be seen from the above definitions and the example illustrated in Fig. 4, we can build compatibility graphs for each individual node. Each sensor node has limited capability in discriminating between pairs of movements, which is shown in the corresponding compatibility graph. The term *compatible* defined for a pair of actions is effectively representative of movements' distinguishability.

A. Problem Definition

The action coverage problem is used to find a minimal set of nodes that still encompasses full coverage within their capacity. The idea behind action coverage is that only a subset of sensor nodes is sufficient to provide accurate detection of every target action. We refer to such a subset as a *complete set* and define it as follows.

Definition 3. A simple graph $G = (V, E)$ is a *complete graph* if for every pair of distinct vertices u and v, there is an edge $(u, v) \in E$.

Definition 4. A subset S' of sensor nodes ($S' \subseteq S$) is a *complete set*, if the compatibility graphs derived from S' altogether form a complete graph. That is, the graph G' computed by $G' = \cup_{i:s_i \in S'} G_i$ is a complete graph.

Definition 5. Given a finite set of sensor nodes $S = \{s_1, s_2, ..., s_n\}$ and a set of movements $M = \{1, 2, ..., m\}$, *Minimum-Cost Action Coverage (MCAC)* is the problem of finding a subset $S' \subseteq S$ in which every pair $j_1, j_2 \in M$ are compatible and $|S'|$ is minimized.

B. Problem Complexity

In this section we address the complexity of the problem outlined. We show that this problem is NP-hard, and therefore no polynomial-time algorithm exists that solves it unless P=NP. The formulation of the MCAC problem closely resembles the well-studied *Minimum Set Cover (MSC)* problem.

Definition 6. Let U be a set of finite elements and $S = \{s_1, s_2, ..., s_n\}$ be a collection of subsets of U such that their union forms U. *Minimum Set Cover* is the problem of finding a subset $S' \subseteq S$ that covers all elements in U and for which $|S'|$ is minimized.

Theorem 7. The MCAC problem is NP-hard.

Proof. We prove that the MCAC problem is NP-hard by exact reduction from MSC. Consider an MCAC instance (M,S) consisting of a finite set of sensor nodes $S = \{s_1, s_2, \ldots, s_n\}$ and a set of movements $M = \{1, 2, \ldots, m\}$. We replace every $s_i \in S$ with the corresponding compatibility graph indicated by a set of edges E_i. Let $\widetilde{S}$ be the set of all E_i's; $\widetilde{S} = \{E_1, E_2, \ldots, E_n\}$. We form a complete graph with the

set of edges from all compatibility graphs: $\widetilde{U} = \bigcup_{i=1}^{n} E_i$. We then consider $(\widetilde{U}, \widetilde{S})$ an instance of the MSC problem. Since MSC as an optimization problem is NP-hard, the MCAC problem is also NP-hard.

We have proved that the Minimum Cost Action Coverage problem is NP-hard. Consequently, our goal is to compute the minimum number of nodes that achieve full action coverage. This can be accomplished using either an ILP or greedy approach. ILP is used to obtain the lower bound of the solution, while the greedy approach provides a fast heuristic. The quality of the solution generated by the greedy algorithm is compared to the lower bound generated by the ILP in the experimental results section.

C. ILP Approach

In this section, we present an integer linear programming formulation for the action coverage problem. Since each node is represented by a graph, we state this problem as follows.

Problem 8. Given compatibility graphs $G_1 = (V, E_1)$, $G_2 = (V, E_2)$, ...,$G_n = (V, E_n)$, and a complete set of all edges $E = \bigcup_{i=1}^{n} E_i$, select a subset of graphs $G'_1, G'_2, \ldots, G'_m$ taken from $G_1, G_2, \ldots, G_n$, such that $\bigcup_{i=1}^{m} E'_i = E$ and the number of selected graphs (m) is minimized.

The corresponding ILP formulation is presented as follows.

$$x_i = \begin{cases} 1, & \text{if graph } G_i \text{ is selected} \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

$$Min \sum_{i=1}^{n} x_i \quad (3)$$

subject to:

$$\sum_{i:e_j \in G_i} x_i \geq 1 \qquad \forall e_j \in E \quad (4)$$

$$x_i \in \{0, 1\} \quad (5)$$

The variables x_i ($i = 1, 2, ..., n$) indicate whether graph G_i is selected to form a complete graph. The inequality constraint (4) ensures that for each edge e_j in the complete graph, at least one of the compatibility graphs that contains that edge is selected. The objective function (3) attempts to minimize the number of graphs selected to form a complete graph. This is equivalent to minimizing the number of active nodes, which suitably leads to energy reduction in the system.

In [37], Lund et al. show that the set cover problem cannot be approximated in polynomial time to within a factor of $O(\log n)$ unless NP has quasi-polynomial time algorithms. In the following, we present a greedy solution for the action coverage problem which obtains this approximation.

D. Greedy Approach

The greedy approach selects the compatibility graphs as follows: at each stage, it picks a compatibility graph G_i that covers the most uncovered edges; next it picks the next graph that covers the most remaining edges; this continues until all edges are covered. At the end of the algorithm, graph G will be a complete graph. A detailed description of this approach is shown in Algorithm 1.

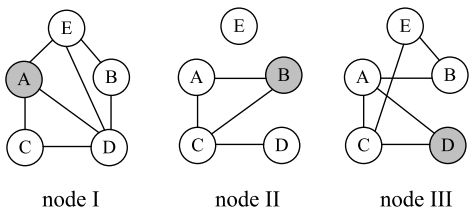

Fig. 5. **Compatibility graphs for dynamic design decisions**

Algorithm 1 Greedy Solution for Action Coverage

Require: Set of compatibility graphs $G_1 = (V, E_1)$, $G_2 = (V, E_2)$,..., $G_n = (V, E_n)$
Ensure: Target complete graph $G = (V, E)$
  $CG = G_1 \cup G_2 \cup ... \cup G_n$
  $G = \emptyset$
  while $G \neq CG$ **do**
    for all uncovered graphs G_i **do**
      $\alpha_i = |G_i \cap (CG - G)|$
    end for
    Find uncovered graph G_i s.t. G_i=argmax$_i\{\alpha_i\}$
    $G = G \cup G_i$
    Add G_i to the list of covered graphs
  end while

VI. Dynamic Design Decision

Earlier, we presented static action coverage for a movement monitoring system. That is, we found the minimum number of active nodes that cover all actions. In this section, however, we study the potential of our approach in regards to the dynamic deactivation of nodes. Once the action has occurred, each node classifies it individually. The final classification involves some notion of collaboration between the nodes in real-time. Our dynamic sensor selection tends to find even smaller set of nodes based on current classification results obtained by individual nodes. Previous studies of embedded sensor nodes have shown that data communication is very expensive in terms of energy consumption, whereas data processing is relatively inexpensive [38]. Therfore, further reducing the number of nodes involved at this stage reduces the communication overhead, and thus the power usage.

We explain details of this technique through an example where the system consists of three sensor nodes with compatibility graphs shown in Fig. 5. This system monitors subjects for the five movements A, B, C, D and E. Sensor nodes I, II, and III classify the movement as A, B, and D, respectively. These classified movements are shown as shaded vertices. The compatibility graph for *node I* indicates that the target movement could be A or B as this node is not able to distinguish between A and B. The graph for *node II* indicates that the movement could be B, D, or E; and for *node III*, target movement could be one of D, B, or E. By intersecting these possibilities, we see that the global classification results movement B. However, we do not need both *node II* and *node III* to determine this; one or the other is sufficient. We could potentially reduce power by eliminating one of the nodes before initiating communication.

Hence, we propose the following approach: First, select a master node. This is done by selecting the node whose target movement vertex has the highest out degree. In this case, in the compatibility graph for *node I*, movement A has an out-degree of three, for *node II*, movement B has an out-degree of two; and for *node III*, movement D also has an out-degree of two. Therefore, *node I* should be the master. Next, add the master node to the solution space. Then, apply the action coverage problem from the master node's point of view and find the minimum number of nodes that will achieve full coverage of the target movement. In this case, only the edge (A, B) is missing from the master node, which can be covered by either of the remaining nodes. Finally, obtain the set of possible classifications from each of the remaining nodes (including the master), and intersect them to achieve final classification. Assume the action coverage allows *nodes I* and *II* to be the active nodes. The results issued are $\{A, B\}$ and $\{B, E, D\}$, leaving B as the final target movement.

VII. System Lifetime Formulation

The action coverage problem outlined above can significantly reduce the number of sensor nodes needed to attain a full coverage. A solution for MCAC gives the smallest complete set capable of providing global knowledge of the whole system. This is done by taking into account the information derived from the most knowledgeable nodes in the network. By restricting action coverage to the smallest complete set, system lifetime tends to be dependent on how long the minimal set would work. That is, the problem appears when a sensor node in the complete set runs out of energy. When this happens, the system may no longer be operational. It turns out that a global optimization is required to obtain maximum lifetime. In other words, system lifetime can be increased by forcing the nodes within the complete sets to collaborate. Given a list of all possible complete sets, the objective is to investigate how long each complete set should be active to maximize system lifetime. In the following formulations, we notice that each complete set can be viewed as a set of sensor nodes capable of providing full coverage.

Problem 9. Let **S** be a finite set of complete sets S_k such that $|\mathbf{S}| = l$. Let t_k be the total time complete set S_k is activated. Thus, the system lifetime, T, is computed by summing t_k. Also, let E_i be the initial energy per unit time for node s_i and a_{ik} be a constant indicating whether complete set S_k includes sensor node s_i or not. The maximum lifetime problem is the problem of finding the time values t_k such that T is maximized.

This problem can be formulated as follows.

$$t_i = \text{total time complete set } S_k \text{ is activated} \tag{6}$$

$$a_{ik} = \begin{cases} 1, & \text{if complete set } S_k \text{ includes node } s_i \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

$$Max \;\; T \tag{8}$$

subject to:

$$\sum_{k=1}^{l} t_k = T \tag{9}$$

$$\sum_{k=1}^{l} a_{ik} t_k \leq E_i \qquad \forall i \in \{1, 2, \ldots, n\} \tag{10}$$

$$t_k \geq 0 \qquad \forall k \in \{1, 2, \ldots, l\} \tag{11}$$

The objective function (8) implies that the system lifetime is the term to be maximized. The equation (9) ensures that all movements are covered during system lifetime. At each time unit, there is exactly one complete set consisting of active nodes that cover the desired actions. The inequality (10) limits the total activation time for each sensor node (s_i) to its initial energy per unit time (E_i).

The above formulation is a typical LP formulation where t_k are real numbers and the objective is to maximize the lifetime T. The optimal solution consisting of values of t_k, and consequently T, can be computed in polynomial time. The total time that sensor s_i is active can be computed by equation (12).

$$T_i = \sum_{k=1}^{l} a_{ik} t_k \tag{12}$$

Since the values t_k represent the total time the nodes within each complete set are activated, they cannot specify the activation and deactivation times for each sensor node in a direct way. That is, we are required to find a scheduler that determines the time frame each sensor node should be active to impart full coverage over the lifetime. Fortunately, the above formulation gives direct insights into node scheduling because any scheduling considered for complete sets is still schedule-preserving for sensor nodes. The reason is that each complete set can be viewed as a set of sensor nodes, and different complete sets are mutually exclusive; i.e., they cannot occur at the same time. Thus, an activation/deactivation of each complete set directly influences the nodes within that set.

An easy-to-implement algorithm that can be utilized for node scheduling is to activate each complete set S_k for its total lifetime t_k. Given complete sets S_k and corresponding activation times t_k, this algorithm works in this way: Choose a complete set S_b arbitrarily. Let t_b be the activation time for S_b. Activate all sensor nodes within S_b for time t_b then remove set S_b from further consideration. Repeat the algorithm until there is no more complete set to process.

This optimization approach assumes equal probability of occurrence for a priori movements of interest. The solution to this LP formulation gives the optimal activation time associated with each sensor node. The technique is still applicable for sensor activation in real time, where no information on the frequency of occurrence of each movement is available. In this case, the system can be forced to activate complete sets with respect to the remaining energy within each set.

TABLE II
MOTE LOCATIONS

No.	Description
1	Waist
2	Left-forearm
3	Left-arm
4	Right-forearm
5	Right-ankle
6	Right-thigh
7	Left-ankle
8	Left-thigh

VIII. EXPERIMENTAL ANALYSIS

We prepared an experiment with eight sensor nodes placed on a subject as shown in Fig. 2, using Telos motes by moteiv® with our custom-designed sensor board. The location of each sensor node is also listed in Table II. For each of the five data streams (x, y, z acceleration and x, y angular velocity), we extracted the seven features previously listed. In this particular experiment, we had three male test subjects between the ages of twenty-five and thirty-five. Each subject performed the twenty-five movements listed in Table III, for ten trials each. The following experimental analyses use the data collected from this experiment.

A. Compatibility Graphs

For each sensor node the Bhattacharyya distance is calculated between all movement pairs, and compatibility graphs are generated. A compatibility graph generated from data collected from the waist node is shown in Fig. 6. For this figure, a subset of movements is shown for simplicity. Each vertex corresponds to a movement as labeled in Table III. The edges represent pairs of compatible movements. For example, there are edges between vertex 10 and all other vertices except 14 and 22. This means the action "turn clockwise 90 degrees" can be distinguished with a high level of confidence from all actions except "look back clockwise" and "grasp an object with two hands". The rest of the links shown in the figure can be interpreted similarly.

In Table IV, we show statistics extracted from eight compatibility graphs considering 25 movements. The second column ($|E|$) represents the total number of edges within each graph. The third column (AD) shows the average outdegree of the vertices. This measure can imply the level of separability from the rest of movements. The next two columns (MnD and MxD) present the minimum and maximum outdegrees among all movements. Finally, the last two columns show the movements that have minimum and maximum outdegrees respectively. For instance, in the compatibility graph constructed for the fourth node, movement 8 (kneeling) has the smallest degree (12), but movements 2, 12 and 13 are completely separated from the rest, as they have a degree of 12.

B. Static Design Coverage

We compare the ILP and greedy approaches using our data. Using both algorithms, we determine the number of nodes needed to distinguish between all twenty-five movements. We split the movements into four mutually exclusive subsets, shown under the "Category" label in Table III. The split is performed intuitively and is based on the level of involvement of body segments in each movement. This categorization provides meaningful information for designing a system that is restricted to monitoring particular movements. The idea comes from the fact that in many medical applications, only a subset of actions are considered valid movements with respect to the temporal and spatial conditions. In the temporal case, the set of actions that are addressed changes from time to time while in the spatial case, movements of interest are updated when the subject moves to a new geographical area. Physicians need to quantify the level of daily activities with respect to certain movements for their patients. Furthermore, the movements a person might perform can significantly change when cooking in the kitchen compared to going to the gym. We present results for a few categories, but the approach can be used for any subset of interest. Table V compares the performance of the two methods on the full set of movements and on each subset. The solutions are indicated using a binary pattern. The nodes that are selected to be active nodes are expressed by a "1" in the resulting pattern. For example, to provide distinguishability information for all twenty-five movements, the ILP technique chose nodes 3, 4, 6 and 7 while the greedy algorithm chose nodes 1, 2, 4, 6 and 7. As expected, the ILP generated a slightly smaller set of nodes compared to the greedy approach. As the results show, the system is capable of providing full coverage of the movements using only four sensor nodes placed on four different segments of the body. To test the effectiveness of action coverage, we also compare the classification accuracy before and after node reduction. The results are shown in Table VII where the second and third columns represent the accuracy using original data and the data collected from active nodes respectively. The very small differences between two cases (e.g. 1.5% for k=1) demonstrate the capability of action coverage to reduce the number of active nodes while maintaining an acceptable quality of service.

Table VI presents the amount of power saving along with the running time of the ILP algorithm. The power saving shows the total percentage of the power preserved in the system compared to the case when no power reduction technique

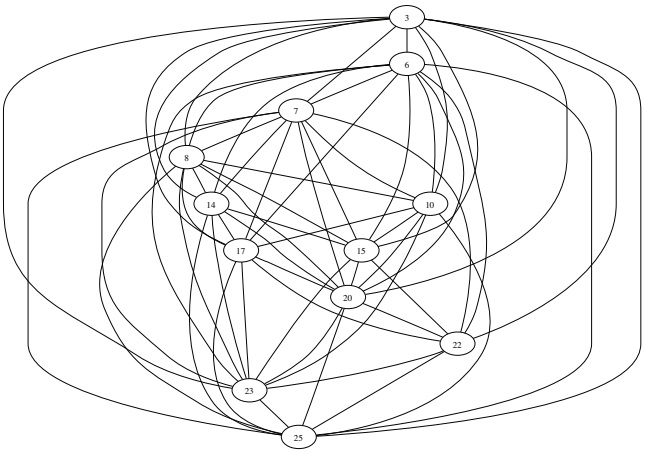

Fig. 6. **Compatibility graph based on data from the waist node for** 12 **movements**

TABLE III
MOVEMENTS FOR EXPERIMENTAL ANALYSIS

No.	Description	Category
1	Stand to sit	Full
2	Sit to stand	Full
3	Stand to sit to stand	Full
4	Sit to lie	Full
5	Lie to sit	Full
6	Sit to lie to sit	Full
7	Bend and Grasp	Upper
8	Kneeling, right leg first	Lower
9	Kneeling, left leg first	Lower
10	Turn clockwise 90 degrees	Turning
11	Turn counter clockwise 90 degrees	Turning
12	Turn clockwise 360 degrees	Turning
13	Turn counter clockwise 360 degrees	Turning
14	Look back clockwise	Upper
15	Move forward (1 step)	Full
16	Move backward (1 step)	Full
17	Move to the left (1 step)	Full
18	Move to the right (1 step)	Full
19	Reach up with one hand	Upper
20	Reach up with two hands	Upper
21	Grasp an object with right hand, turn 90 degrees and release	Turning
22	Grasp an object with two hands, turn 90 degrees and release	Turning
23	Jumping	Full
24	Going upstairs (2 stairs)	Lower
25	Going downstairs (2 stairs)	Lower

TABLE IV
COMPATIBILITY GRAPH STATISTICS

Node #	$\|E\|$[1]	AD[2]	MnD[3]	MxD[4]	MnD_V[5]	MxD_V[6]
1	263	21	17	24	{9,24}	{4,5,6,7,12,13,23}
2	250	20	14	24	{9,21}	{12,13,23}
3	263	21.04	16	24	{24}	{4,5,6,12,13,20,22,23}
4	255	20.4	12	24	{8}	{2,12,13}
5	259	20.72	15	24	{17}	{4,5,6,12,13}
6	252	20.16	15	24	{18}	{1,2,3,12,13}
7	260	20.8	15	24	{18}	{4,5,6,12,13}
8	253	20.24	13	24	{17,18}	{1,2,3,12,13}

1. Number of edges
2. Average degree of the vertices
3. Minimum degree of the vertices
4. Maximum degree of the vertices
5. Minimum degree vertices
6. Maximum degree vertices

TABLE V
SOLUTIONS TO ACTION COVERAGE PROBLEM

Movements	ILP Solution	Greedy Solution
Mote #	1 2 3 4 5 6 7 8	1 2 3 4 5 6 7 8
All	0 0 1 1 0 1 1 0	1 1 0 1 0 1 1 0
Upper Body	0 1 0 0 0 0 0 0	0 1 0 0 0 0 0 0
Lower Body	0 0 0 0 1 0 0 0	0 0 0 0 1 0 0 0
Turning	0 1 0 1 0 0 0 0	0 1 0 1 0 0 0 0
Full Body	0 0 0 0 0 1 1 0	1 1 0 0 0 1 0 0

TABLE VI
ANALYSIS OF ILP SOLUTIONS

Movements	#Nodes	Power Saving(%)	Execution Time(sec.)
All	4	%50	6.761290
Upper Body	1	%87	0.082245
Lower Body	1	%87	0.060753
Turning	2	%75	0.065421
Full Body	2	%75	0.074850

is applied. We solve the linear programming optimization problem in MATLAB on a Dell Laptop with a 1.6GHz Core 2 Duo processor. The execution time of the ILP for 25 movements is about 6.8 seconds.

C. Dynamic Design Coverage

Throughout the classification, three of the trials for each subject and movement were used for training, and the remaining trials were used for validation. Since our experiments are carried out in a controlled environment, this split between training and test data provides good classification results as we will demonstrate later in this paper. Per-node feature extraction is performed to obtain seven features for each action across five sensor readings. We use a data fusion scheme to integrate data from different nodes at the feature level. Therefore, the final classifier works on the same training and test sets whereas the feature space has been extended by the sensor nodes. Only the four nodes selected by ILP for all movements are used for the dynamic analysis (see Table V). Compatibility graphs are generated from the training set. Classification is performed using a k-NN classifier, where k varies from 1 to 8. This

TABLE VII
CLASSIFICATION ANALYSIS

K (No reduction)	Accuracy[9] (Static)	Accuracy[10] (Dynamic)	Accuracy[11] (Dynamic)	#Nodes[12]
1	%97.5	%96.0	%92.2	1.84
2	%96.6	%95.0	%89.9	1.85
3	%96.6	%94.3	%88.9	1.85
4	%94.8	%95.2	%89.1	1.85
5	%95.4	%92.9	%88.2	1.86
6	%91.2	%89.5	%81.1	1.86
7	%90.3	%90.3	%80.8	1.85
8	%89.5	%88.0	%78.9	1.86

9. Classification accuracy considering all sensor nodes
10. Classification accuracy after static node reduction
11. Classification accuracy after dynamic node reduction
12. Average number of nodes in dynamic mode

dynamic technique further reduces the number of active nodes to an average of 1.84 nodes (for $k = 1$) per classification.

D. Classifier Accuracy

Classification accuracy exhibits how confident twenty-five movements can be recognized. Therefore, we define accuracy as follows:

$$A = \frac{TP + TN}{N} \tag{13}$$

Where TP is the number of true positive samples, TN represents the number of true negative samples and N is the total number of test points.

We feed all the features from all eight motes into a k-NN classifier where $k = 1$, giving an accuracy reading of 97.5%. We repeat the test using data from only the four nodes selected by the ILP as shown in Table V and reach an accuracy of 96.0%. Finally, we used a classifier based on the dynamic design coverage solution and reach an accuracy of 92.2%. The complete results are shown in Table VII.

E. Lifetime

In this section, we will present the results of system lifetime optimization using real data. The initial energy per unit time for each sensor node is calculated based on the overall active power consumption. To calculate this value, we measured the transmission power consumption of a Telos mote equipped with a tri-axial LIS3LV02DQ accelerometer and a bi-axial IDG-300 gyroscope. The average power consumed in a five-minute duration was $88mW$. We assume that the system is equipped with two $1.5V, 2000mAh$ AA batteries. We also assume that all sensor nodes have the same amount of initial energy. We then solve the power lifetime problem for different categories of movements as illustrated in Table VIII. For each set of actions, the total system lifetime (T) and the sensor total activation time (T_i) are calculated. As shown in Table VIII, for some set of actions, the system lifetime is limited by the lifetime of a single node. For instance, if the system is used for detecting turning activities, it cannot work beyond 68 hours, which is the lifetime for node 4. However, if the system is used to identify lower-body actions, it would be operational for 204 hours. In Table IX, we compare the lifetime of the proposed

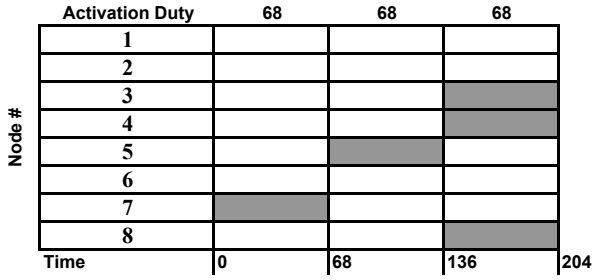

Fig. 7. **Node Scheduling for Lower Body Movements**

model in section VII to the case where the system operates using the minimal set given by the action coverage problem. The results show that when considering several solution sets as derived in section VII, the system lifetime can be increased by a factor of three.

To determine a time table for sensor activations/deactivations, a random scheduler is employed as previously discussed. The results demonstrated in Fig.7 for the lower body movements show the activation duty associated with each mote. A shaded segment represents an activation duration for the corresponding node. Sensor node 7 which provides a full coverage for lower-body movements becomes active first and works for 68 hours. Sensor node 5 which was idle for the first 68 hours of the system operation, wakes up and works for 68 hours until it dies at hour 136. Next complete set then activates nodes 3, 4, and 8. These nodes work for 68 hours until they die at hour 204.

IX. CONCLUSION AND FUTURE WORK

In this article, we proposed a novel power optimization technique that examined the sensor coverage problem from a classification perspective. We used compatibility graphs to combine local information attained by individual nodes and ensure full sensing coverage. We formulated the action coverage problem, which attempts to minimize the number of active nodes in the system while maintaining acceptable accuracy for action recognition. Furthermore, we demonstrated how the collaboration of sensor nodes can lead to maximal lifetime. While each individual node has limited knowledge about the system, global knowledge of the system can be achieved by the data fusion technique described in this paper. The experimental results demonstrated the effectiveness of our approach.

Using a hidden Markov model, we may be able to further reduce the number of potential movements. For instance, someone lying on a bed cannot fall, walk, sit down, or jump. We plan to investigate how this model can assist in further node reduction and power optimization.

REFERENCES

[1] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. B. Srivastava, "Coverage problems in wireless ad-hoc sensor networks," *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 1380–1387 vol.3, 2001.

[2] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Communications, 2001. ICC 2001. IEEE International Conference on.* Helsinki, Finland: IEEE Computer Society, 2001, pp. 472–476.

TABLE VIII
NODE ACTIVATION TIME AND TOTAL SYSTEM LIFETIME(HOURS)

Movements	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T(Lifetime)
All	22	42	33	68	28	31	38	37	**68**
Upper Body	0	35	111	136	57	64	115	72	**136**
Lower Body	0	0	68	68	68	0	68	68	**204**
Turning	0	34	34	68	0	0	0	0	**68**
Full Body	40	39	31	25	21	68	68	68	**136**

TABLE IX
LIFETIME COMPARISON WITH MINIMAL SET ASSIGNMENT

Movements	Lifetime ratio
All	1
Upper Body	2
Lower Body	3
Turning	1
Full Body	2

[3] Z. Abrams, A. Goel, and S. Plotkin, "Set k-cover algorithms for energy efficient monitoring in wireless sensor networks," in *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*. New York, NY, USA: ACM, 2004, pp. 424–432.

[4] D. Tian and N. D. Georganas, "A coverage-preserving node scheduling scheme for large wireless sensor networks," in *WSNA '02: Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. New York, NY, USA: ACM Press, 2002, pp. 32–41.

[5] M. Cardei, D. MacCallum, X. Cheng, M. Min, X. Jia, D. Li, and D. Z. Du, "Wireless sensor networks with energy efficient organization," *J. Interconnection Networks*, vol. 3, pp. 213–229, 2002.

[6] W. Wang, V. Srinivasan, B. Wang, and K. C. Chua, "Coverage for target localization in wireless sensor networks," *Information Processing in Sensor Networks, 2006. IPSN 2006. The Fifth International Conference on*, pp. 118–125, 19-21 April 2006.

[7] G. Xing, C. Lu, R. Pless, and J. A. O'Sullivan, "Co-grid: an efficient coverage maintenance protocol for distributed sensor networks," in *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*. New York, NY, USA: ACM, 2004, pp. 414–423.

[8] C. Huang and Y.-C. Tseng, "The coverage problem in a wireless sensor network," *Mob. Netw. Appl.*, vol. 10, no. 4, pp. 519–528, 2005.

[9] W. Wang, V. Srinivasan, K.-C. Chua, and B. Wang, "Energy-efficient coverage for target detection in wireless sensor networks," in *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2007, pp. 313–322.

[10] M. Cardei and D.-Z. Du, "Improving wireless sensor network lifetime through power aware organization," *Wirel. Netw.*, vol. 11, no. 3, pp. 333–340, 2005.

[11] B. Wang, W. Wang, V. Srinivasan, and K. C. Chua, "Information coverage for wireless sensor networks," *IEEE Commun.Lett.*, vol. 9, no. 11, pp. 967–969, Nov. 2005.

[12] M. Cardei and J. Wu, "Energy-efficient coverage problems in wireless ad-hoc sensor networks," *Computer Communications*, vol. 29, no. 4, pp. 413–420, February 2006.

[13] R. Ghrist and A. Muhammad, "Coverage and hole-detection in sensor networks via homology," *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pp. 254–260, 15 April 2005.

[14] J. Carle and D. Simplot-Ryl, "Energy-efficient area monitoring for sensor networks," *Computer*, vol. 37, no. 2, pp. 40–46, Feb 2004.

[15] F. Zhao, J. Shin, and J. Reich, "Information-driven dynamic sensor collaboration," *IEEE Signal Processing Mag.*, vol. 19, no. 2, pp. 61–72, March 2002.

[16] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. Wiley-Interscience, July 2004.

[17] C. Chong and S. Kumar, "Sensor networks: evolution, opportunities, and challenges," *Proc. IEEE*, vol. 91, no. 8, pp. 1247–1256, Aug. 2003.

[18] R. E. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, pp. 197–227, 1990. [Online]. Available: citeseer.ist.psu.edu/schapire90strength.html

[19] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, 1997.

[20] Y. Ivanov and R. Hamid, "Weighted ensemble boosting for robust activity recognition in video," *MG&V*, vol. 15, no. 3, pp. 415–427, 2006.

[21] J. H. K. Vinod, Ramachandran, and A. M. Sayeed, "Distributed multi-target classification in wireless sensor networks," *IEEE J. Select. Areas Commun.*, vol. 23, pp. 703– 713, April 2005.

[22] P. Viola and M. Jones, "Robust real-time object detection," *International Journal of Computer Vision - to appear*, 2002. [Online]. Available: citeseer.ist.psu.edu/viola01robust.html

[23] R. Brooks, P. Ramanathan, and A. M. Sayeed, "Distributed target classification and tracking in sensor networks," *Proc. IEEE*, vol. 91, no. 8, pp. 1163–1171, Aug. 2003.

[24] D. Li, K. Wong, Y. H. Hu, and A. Sayeed, "Detection, classification, and tracking of targets," *IEEE Signal Processing Mag.*, vol. 19, no. 2, pp. 17–29, Mar 2002.

[25] R. Jafari, W. Li, R. Bajcsy, S. Glaser, and S. S., "Physical activity monitoring for assisted living at home," in *BSN07: Proc. 4th International Workshop on Wearable and Implantable Body Sensor Networks*, Aachen University, Germany, 2007, pp. 213–219.

[26] B. Logan, J. Healey, M. Philipose, E. Tapia, and S. Intille, "A long-term evaluation of sensing modalities for activity recognition," in *UbiComp 2007: Ubiquitous Computing*, 2007, pp. 483–500.

[27] M. Mathie, B. Celler, N. Lovell, and A. Coster, "Classification of basic daily movements using a triaxial accelerometer," *Medical and Biological Engineering and Computing*, vol. 42, no. 5, pp. 679–687, 2004.

[28] R. Ramachandran, L. Ramanna, H. Ghasemzadeh, G. Pradhan, R. Jafari, and B. Prabhakaran, "Body sensor networks to evaluate standing balance: Interpreting muscular activities based on inertial sensors," in *The 2nd International Workshop on Systems and Networking Support for Healthcare and Assisted Living Environments (HealthNet)*. Breckenridge, CO: ACM, June 2008.

[29] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," *Pervasive 2004*, pp. 1–17, April 2004. [Online]. Available: http://dx.doi.org/10.1007/b96922

[30] J. Lester, T. Choudhury, and G. Borriello, *A Practical Approach to Recognizing Physical Activities*. Springer-Verlag Berlin Heidelberg, 2006. [Online]. Available: http://dx.doi.org/10.1007/11748625_1

[31] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience Publication, 2000.

[32] E. Guenterberg, H. Ghasemzadeh, R. Jafari, and R. Bajcsy, "A segmentation technique based on standard deviation in body sensor networks," in *Dallas-EMBS: Proc. IEEE Dallas Engineering in Medicine and Biology Workshop*. IEEE, 2007.

[33] H. Ghasemzadeh, E. Guenterberg, K. Gilani, and R. Jafari, "Action coverage formulation for power optimization in body sensor networks," *Design Automation Conference, 2008. ASPDAC 2008. Asia and South Pacific*, pp. 446–451, March 2008.

[34] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bull. Calcutta Math. Soc.*, vol. 35, pp. 99–109, 1943.

[35] T. Kailath, "The divergence and bhattacharyya distance measures in signal selection," *IEEE Trans. Commun. Technol.*, vol. 15, no. 1, pp. 52–60, February 1967.

[36] B. Kim and D. Landgrebe, "Prediction of optimal number of features," in *IGARSS '90: 10th Annual International Geoscience and Remote Sensing Symposium, Remote Sensing Science for the Nineties*, 1990, pp. 2393–2396.

[37] C. Lund and M. Yannakakis, "On the hardness of approximating minimization problems," *J. ACM*, vol. 41, no. 5, pp. 960–981, 1994.

[38] V. Raghunathan, C. Schurgers, S. Park, and M. Srivastava, "Energy-aware wireless microsensor networks," *IEEE Signal Processing Mag.*, vol. 19, no. 2, pp. 40–50, Mar 2002.

Hassan Ghasemzadeh received the B.Sc. degree in Computer Engineering from Sharif University of Technology, Tehran, Iran, and the M.Sc. degree in Computer Engineering from University of Tehran, Tehran, Iran in 1998 and 2001 respectively. After gaining several years of industry experience in computer networks, he joined Azad University, Damavand, Iran where he served as a faculty member and director of undergraduate studies. He then joined the University of Texas at Dallas in 2007 where he is currently pursuing his Ph.D. in Computer Engineering. His research interests lie in different aspects of embedded systems. He is currently working on distributed signal processing, reconfigurable computing and algorithm design for medical embedded systems. He is a student member of the IEEE.

Eric Guenterberg received his B.S. in Electrical Engineering at the University of California, Los Angeles in 2007. He is currently pursuing his PhD in Electrical Engineering at the University of Texas at Dallas. His research interests are primarily distributed pattern recognition and signal processing with an emphasis on medical/biological applications. He is a research assistant at the ESSP Lab.

Roozbeh Jafari received his B.Sc. in Electrical Engineering from Sharif University of Technology in 2000. He received an M.S. in Electrical Engineering from SUNY at Buffalo, and an M.S. and a Ph.D. in Computer Science from UCLA in 2002, 2004 and 2006 respectively. He spent 2006-2007 in EECS department at UC Berkeley as a post-doctoral researcher. Dr. Jafari is currently an assistant professor in Electrical Engineering at the University of Texas at Dallas. His research is primarily in the area of networked embedded system design and reconfigurable computing with emphasis on medical/biological applications, their signal processing and algorithm design. He is the director of ESSP (Embedded Systems and Signal Processing) Laboratory. He is a member of the IEEE.