Abstract—Advances in technology have led to the development of various light-weight sensory 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 propose a power optimization technique that will consider the sensing coverage problem from a collaborative signal processing perspective. We introduce compatibility graphs and describe how they can be utilized for power optimization. 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. Along side this, we introduce a system for dynamically activating and deactivating sensor nodes in real-time. Finally, we elucidate the effectiveness of our techniques on data collected from several subjects.

I. INTRODUCTION

A large number of patients require long-term care, and fall victim to a pervasive lifestyle of constant monitoring in a pursuit to attain 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 levels of activities; 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. Altogether, these sensors form a body sensor network (BSN). BSNs have the ability to diagnose critical events such as heart attacks or failure, and monitor activity by recording the duration, quality and type of movement performed. This data can be more accurate and sufficient 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 improving 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 capitalize 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, kneeling, etc. Typically, the sensor units are identical in which they 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 new optimization technique which we call action coverage. The objective is to select the least amount 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, and 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 five sensor nodes are required.

We introduce compatibility graphs which simplify the visualization of the problem and lead directly to an algorithm to determine the minimum size set for action coverage. Because the problem is NP-hard, we formulate an ILP which attempts to find a lower bound on solutions.
We also provide a quick heuristic algorithm which represents a reasonable approximation. Finally, we present experimental verification of these techniques.

II. RELATED WORK

A great deal of work has been done to minimize the number of homogeneous nodes covering a geographical area. Authors in [1] 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 fields and the field covered by a minimal number of nodes is called critical. The algorithm then selects the nodes that cover the critical elements. Another coverage mechanism is presented in [2] 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. Authors in [3] 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. Several other techniques can be found in [4, 5, 6, 7].

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 [8] 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 utilize 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 actions the body is performing are important. Furthermore, the technique of sequentially activating sensors employed in tracking systems may 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.

Another power reduction strategy involves decreasing the communication overhead for classification. 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 [9] is boosting, in which each individual classifier is re-sampled and the majority of votes are used to combine the results. AdaBoost [10] is another decision combiner that uses a weighted voting scheme to make a global decision. It combines a set of hypotheses through weighted majority voting of the classes predicted by each hypotheses. These collaborative classifiers were designed to be executed on a single system, and therefore do not address communications overhead. Authors in [11] propose a distributed classification system for wireless sensor networks. In their system, hard decisions made towards individual nodes are communicated over noisy links to a coordinator node which optimally combines local results to make a final decision.

Our research takes a novel approach by combining both classification and coverage. We employ the results of the classification to reduce the number of active nodes. Moreover, during the classification stage, we demonstrate an approach to further reduce the number of nodes needed to communicate.

At the present stage of our research, we exclusively focus on reducing the number of nodes, and thus have not investigated integrating standard collaborative classification techniques into our system. In one of our tests, we utilize a collaborative algorithm directly suggested by our compatibility graphs. In comparison to the alternative, the effectiveness of this technique has not yet been analyzed.

III. SYSTEM ARCHITECTURE

The pilot application for our research is physical movement monitoring. Our system consists of several sensor units; where each has a tri-axial accelerometer, a bi-axial gyroscope, a microcontroller, and a radio, as shown in Fig. 1. The processing unit of each node, or mote, samples sensor readings at 22 Hz and transmits the data in a wireless manner to a base station using a TDMA protocol. Our motes, Tmote Sky, are commercially available from motev  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:** The data is collected from each of the five sensors on each of the eight sensor nodes at 22 Hz.
**Fig. 2.** Experimental subject wearing eight sensor node. Each node has a tri-axial accelerometer and a bi-axial gyroscope.

**Fig. 3.** Signal processing flow

2. **Preprocessing:** The data is filtered with an eight-point moving average.

3. **Segmentation:** We determine the portion of the signal that represents a complete action. For experimental purposes, this is done by manually making the actions.

4. **Feature Extraction:** Single value features are extracted. Features include:
   - Mean
   - Start-to-end amplitude
   - Standard deviation
   - Peak-to-Peak amplitude
   - RMS power

5. **Per-Node classification:** Each node uses the aforementioned features to determine the most likely action. We use $k$-Nearest Neighbor ($k$-NN) 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. We utilize the former method by feeding features from all sensor nodes into a central classifier.

We currently process all our data offline in MATLAB. This is convenient for rapid prototyping and algorithm development. 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 develop this automated action segmentation.

**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 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 five features per data stream, formulating 25 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, so called compatibility graph, is generated by complementing the conflict graph of Fig. 4b. If a compatibility graph is not complete, then there there exists some movements that the node cannot correctly classify. A complete graph is equivalent to the capability of distinguishing between every pair of classes.

One of the most popular class separability measures in the field of pattern recognition is the Bhattacharyya distance [12]. 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 [13]. 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 is, the better the final classification result.

In our experiments, we exploit the Bhattacharyya distance as a measure of separability between pairs of classes. This measure has an explicit expression for a generalized Gaussian distribution. Since we are dealing with such distribution, we make the Bhattacharyya distance as our probabilistic distance. The distance between two distributions \(i\) and \(j\) is represented by \(\beta(i,j)\) in Equation 1 where \(\mu_i\) and \(\Sigma_i\) denote mean vector and covariance matrix associated with distribution \(i\) respectively:

\[
\begin{align*}
\alpha(i,j) &= 2(1 - e^{-\alpha(i,j)}) \\
\beta(i,j) &= \frac{1}{2}(\mu_i - \mu_j)^{(\frac{1}{2}(\sum_i + \sum_j)^{-1}(\mu_i - \mu_j) + \frac{1}{2}\ln(\frac{\Sigma_i + \Sigma_j}{\Sigma_i})} \\
\end{align*}
\]

**Definition:** Two classes \(i\) and \(j\) are said to be **compatible** if they have complete separability.

The Bhattacharyya distance is assumed to be directly related to the 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 [14].

**V. Problem Formulation**

**A. Problem Definition**

The action coverage problem is used to find a nominal set of nodes that still encompass full coverage within their capacity. This is equivalent to the set cover problem, which is NP-hard. Consequently, our goal is to compute the minimum number of nodes that achieves full action coverage. This can be accomplished using either an ILP or greedy approach. The 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.

**B. ILP Approach**

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

**Problem:** Given compatibility graphs \(G_1 = (V,E_1), G_2 = (V,E_2), \ldots, G_n = (V,E_n)\), 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 \(m\) is minimized.

The corresponding ILP formulation is presented as follows:

**Algorithm 1 Greedy Solution for Action Coverage**

**Require:** Set of compatibility graphs \(G_1 = (V,E_1), G_2 = (V,E_2), \ldots, G_n = (V,E_n)\)

**Ensure:** Target complete graph \(G = (V,E)\)

\(CG = G_1 \cup G_2 \cup \ldots \cup G_n\),

\(G = \emptyset\)

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. \(G_i = \text{argmax}\{\alpha_i\}\)
  \(G = G \cup G_i\)
  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, the minimum number...
of active nodes that cover all actions. In this section, however, we study the potential of our approach in regards to dynamic deactivation of nodes. Once the action has occurred, each node classifies it individually. Final classification involves some notion of collaboration between the nodes. Moreover, further reducing the number of nodes involved at this stage reduces the communication overhead, and thus the power usage. Consider a system consisting of three nodes with compatibility graphs shown in Fig. 5. This system monitors subjects for five movements A, B, C, D and E. Nodes I, II, and III classify the movement as A, B, and D, respectively. The compatibility graph for Node I indicates that the movement could be A or B. 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 should be B. However, only Node II or Node III are sufficient to determine this. 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. Thus, yielding Node I as 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 at the master node: in this case, 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. Experimental Analysis

We prepared an experiment with eight sensor nodes placed on a subject as shown in Fig. 2 using the motiv® motes with our custom designed sensor board. For each of the five data streams (x, y, z acceleration and x, y angular velocity), we extracted the five 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 I ten trials each. The following experimental analyses use the data collected from this experiment.

A. Static Design Decision

We compared the ILP and greedy approaches using our data. For each sensor node the Bhattacharyya distance was calculated between all movement pairs, and compatibility graphs were generated. A compatibility graph generated from our data is shown in Fig. 6 (For this figure, a subset of movements is shown for simplicity). Using the ILP and greedy algorithms, we determined the number of nodes needed to distinguish between all twenty-five movements. Thereafter, we split the movements into four mutually exclusive subsets, shown under the “Category” label in Table I. Table II compares the performance of the two methods on the full set of movements and on each subset. As expected, the ILP generated a slightly smaller-sized set of nodes compared to the greedy approach.

B. Dynamic Design Decision

Throughout the classification, we used three of the trials from each subject and movement for training, and we

![Compatibility graphs for dynamic design decisions](image-url)
In this article, we proposed a novel power optimization technique that examined the sensor coverage problem from a classification perspective. We utilized compatibility graphs to combine several classifiers and ensure full sensing coverage. The experimental results demonstrated the effectiveness of our approach. The node reduction achieved by using a subset of movements showed the advantage of eliminating some movements prior to classification.

Using a hidden Markov model, and knowledge of the movements involved could decrease the number of potential movements. For instance, someone lying on a bed cannot fall, walk, sit down, or jump. We plan to investigate how the Markov chains can be utilized for further power consumption reduction.

### References


