Energy-Efficient Surveillance System Using Wireless Sensor Networks*

Tian He, Sudha Krishnamurthy, John A. Stankovic, Tarek Abdelzaher,
Liqian Luo, Radu Stoleru, Ting Yan, Lin Gu

Department of Computer Science
University of Virginia, Charlottesville, VA 22903
{tianhe, skrish, stankvoic, zaher, ll4p, rs6bd, ty4k, lg6e}@cs.virginia.edu

Jonathan Hui, Bruce Krogh

Department of Electrical and Computer Engineering
Carnegie-Mellon University, Pittsburgh, PA 15213
jwhui@cs.berkeley.edu, krogh@ece.cmu.edu

ABSTRACT

The focus of surveillance missions is to acquire and verify information about enemy capabilities and positions of hostile targets. Such missions often involve a high element of risk for human personnel and require a high degree of stealthiness. Hence, the ability to deploy unmanned surveillance missions, by using wireless sensor networks, is of great practical importance for the military. Because of the energy constraints of sensor devices, such systems necessitate an energy-aware design to ensure the longevity of surveillance missions. Solutions proposed recently for this type of system show promising results through simulations. However, the simplified assumptions they make about the system in the simulator often do not hold well in practice and energy consumption is narrowly accounted for within a single protocol. In this paper, we describe the design and implementation of a running system for energy-efficient surveillance. The system allows a group of cooperating sensor devices to detect and track the positions of moving vehicles in an energy-efficient and stealthy manner. We can trade off energy-awareness and surveillance performance by adaptively adjusting the sensitivity of the system. We evaluate the performance on a network of 70 MICA2 motes equipped with dual-axis magnetometers. Our results show that our surveillance strategy is adaptable and achieves a significant extension of network lifetime. Finally, we share lessons learned in building such a complete running system.

*This work was supported by the DAPRPA IXO offices under the NEST project (grant number F336615-01-C-1905).

MobiSys'04, June 6–9, 2004, Boston, Massachusetts, USA.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design

General Terms

Design, Performance, Experimentation, Measurement

Keywords

Sensor networks, Energy conservation, Tracking, Wireless

1. MOTIVATION

One of the key advantages of wireless sensor networks (WSN) is their ability to bridge the gap between the physical and logical worlds, by gathering certain useful information from the physical world and communicating that information to more powerful logical devices that can process it. If the ability of the WSN is suitably harnessed, it is envisioned that WSNs can reduce or eliminate the need for human involvement in information gathering in certain civilian and military applications. In the near future, sensor devices will be produced in large quantities at a very low cost and densely deployed to improve robustness and reliability. They can be miniaturized into a cubic millimeter package (e.g., smart dust [16]) in order to be stealthy in a hostile environment. Cost and size considerations imply that the resources available to individual nodes are severely limited. We believe, however ,that limited processor bandwidth and memory are temporary constraints in sensor networks. They will disappear with fast developing fabrication techniques. The energy constraints on the other hand are more fundamental. According to R.A. Powers [20], battery capacity only doubles in 35 years. Energy constraints are unlikely to be solved in the near future with the slow progress in battery capacity and energy scavenging. Moreover, the untended nature of sensor nodes and the hazardous sensing environment preclude manual battery replacement. For these reasons, energy awareness becomes the key research challenge for sensor network protocol design. Several researchers have addressed energy conservation recently. Most of them

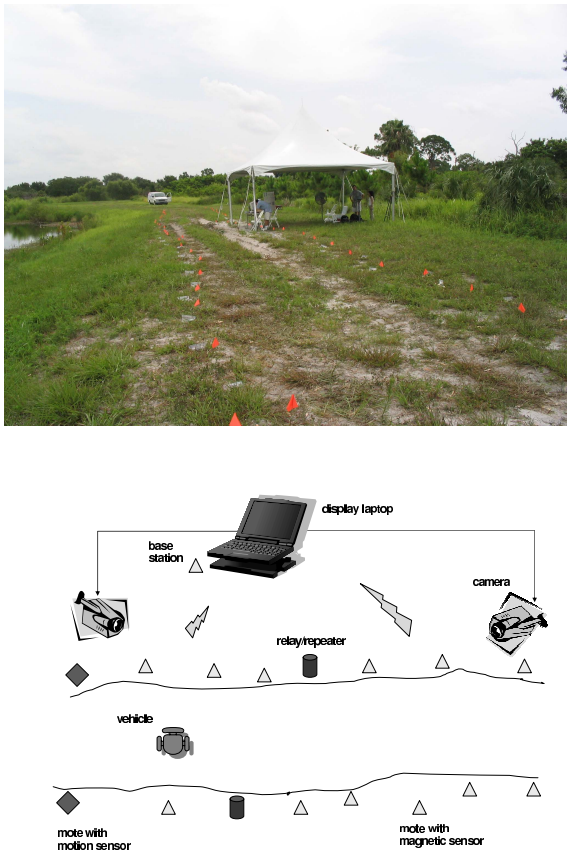

Figure 1: Sensor Network Deployment

focus on particular protocols and investigate whether their energy conservation goal can be achieved. To the best of our knowledge, none of them investigate energy-conservation for a running system as whole. Normally they evaluate their approach through simulations. Simulation approaches tend to make simplified assumptions that often do not hold well in practice and they are subject to incompleteness. For example, in [22][23][21], several sensing coverage schemes are proposed for energy conservation. None of them consider energy consumption in activities other than sensing.

In this paper, we describe our effort that involves system design and implementation on a MICA2 platform with 70 MICA2 motes. The primary goal of the system is to support the ability to track the position of moving targets in an energy-efficient and stealthy manner. Our experimental results show that the probability of false alarms observed reaches zero when aggregation is achieved among more than 3 member motes. The experimental results we obtained also show that with 5% of deployed motes serving as sentries and the non-sentries operating at a 4% duty cycle, our algorithm extends the lifetime of a sensor network by up to 900%.

The main contributions of this paper are: 1) the design and implementation of a running system with energy-awareness as the main design principle across multiple components, 2) Mechanisms for dynamic control, which allow tradeoffs between energy-efficiency and system performance by adjusting the sensitivity of the system, and 3) a physical implementation and field evaluation that reveal the practical issues that are hard to capture in simulation.

The remainder of this paper is organized as follows. Section 2 describes the requirements of a typical ground surveillance application. In Section 3, we describe the system setup and hardware components. In Section 4, we provide an overview of our system design. In Section 5, we elaborate on how the individual components of the system contribute to energy-efficient tracking. In Section 6, we discuss implementation issues concerning our system. We present experimental results in Section 7, and summarize the lessons learned from our experience in Section 8. Finally we conclude in Section 9 and discuss some future work in Section 10.

2. APPLICATION REQUIREMENTS

Our system design is motivated by the requirements of a typical ground surveillance application. The general objective of such an application is to alert the military command and control unit in advance to the occurrence of events of interest in hostile regions. The event of interest for our work is the presence of moving vehicles in the deployed region. The deployed sensor devices must have the ability to detect and track vehicles in the region of interest. Successful detection and tracking requires that the application obtain the current position of a vehicle with acceptable precision and confidence. When the information is obtained, it has to be reported to a remote base station within an acceptable latency. Several application requirements must be satisfied to make this system useful in practice:

- **Longevity:** The mission of a surveillance application typically lasts from a few days to several months. Due to the confidential nature of the mission and the inaccessibility of the hostile territory, it may not be possible to manually replenish the energy of the power-constrained sensor devices during the course of the mission. Hence, the application requires energy-aware schemes that can extend the lifetime of the sensor devices, so that they remain available for the duration of the mission.

- **Adjustable Sensitivity:** The system should have an adjustable sensitivity to accommodate different kinds of environments and security requirements. In critical missions, a high degree of sensitivity is desired to capture all potential targets even at expense of possible false alarms. In other case, we want to decrease the sensitivity of the system, maintaining a low probability of false alarms in order to avoid inappropriate actions and unnecessary power dissipation.

- **Stealthiness:** It is crucial for military surveillance systems to have a very low possibility of being detected and intercepted. Miniaturization makes sensor devices hard to detect physically; however, RF signals can be easily intercepted if sensor devices actively communicate during the surveillance stage. A zero communication exposure is desired in the absence of significant events.

- **Effectiveness:** The precision in the location estimate, and the latency in reporting an event are the metrics that determine the effectiveness of a surveillance system. Accuracy and latency are normally considered important metrics of tracking performance. However, the requirement of these two metrics can actually be

slightly relaxed in many tracking applications. For example, it may be acceptable to obtain location estimation within a couple of feet and receive a detection report within a couple of seconds. We, therefore, focus primarily on the first three metrics mentioned above.

3. SYSTEM DESCRIPTION AND REQUIREMENTS

Figure 1 shows the deployment of our ground surveillance system. We deployed 70 tiny sensor devices, called MICA2 motes [14], along a 280 feet long perimeter in a grassy field that would typically represent a critical choke point or passageway to be monitored. Each of the motes is equipped with a 433 MHz Chipcon radio with 255 selectable transmission power settings. While this radio is sufficient to allow the motes deployed in the field to communicate with each other, it is not capable of long-range (> 1000 ft) communication when put on the ground. Therefore, we assume that in a real system where the command and control units may be deployed several thousands of feet away from the sensor field, devices capable of long-range communication, such as repeaters, will be deployed as gateways to assist the sensors to relay back information from the motes in the field to the base station. In our prototypical deployment, we use a mote as the base station that is attached to a portable device, such as a laptop. The portable device is the destination of the surveillance information and is mainly used for visualization in our prototype system. The camera devices shown in Figure 1 are controlled by the laptop to provide the next level of surveillance information, when triggered by the sensor field.

Each mote is equipped with a sensor board that has magnetic, acoustic, and photo sensors on it. While the different sensors make it possible for a mote to detect different kinds of targets, only the magnetic sensors are relevant to the application described in this paper. We use the HMC1002 dual-axis magnetometers from Honeywell [13]. These magnetic sensors detect the magnetic field generated by the movement of vehicles and magnetic objects. They have an omni-directional field of view and are therefore less sensitive to orientation. They have a resolution of 27 μGauss and their sensing range varies with the size of the magnetic object they are sensing. From our experiments, we found that these sensors can sense a small magnet at a distance of approximately 1 ft and slowly moving passenger vehicles at a distance of approximately 8-10 ft.

4. SYSTEM OVERVIEW

The key contribution of this work is the design and implementation of a wireless sensor network prototype that enables energy-efficient tracking and detection of events. Such a system is useful for surveillance applications, such as the one outlined in Section 2. The system we have designed is organized into a layered architecture comprised of higher-level services and lower-level components, as shown in Figure 2. It is implemented on top of TinyOS [12]. We first provide an overview of the different software components we have designed and then follow that with a detailed discussion of the role played by those components in the context of our tracking and surveillance application.

Time synchronization, localization, and routing comprise the lower-level components and form the basis for implementing the higher-level services, such as aggregation and power management. Time synchronization and localization are important for a surveillance application because the collaborative detection and tracking process relies on the spatio-temporal correlation between the tracking reports sent by multiple motes. The time synchronization module is responsible for synchronizing the local clocks of the motes with the clock of the base station. The localization module is responsible for ensuring that each mote is aware of its location. In our prototype system, we use a simple localization configuration, which statically assigns motes their location at the time they are programmed, assuming we know about where they will be placed. In actual deployment, such as a battlefield in which it is important to track the absolute geographical coordinates of the hostile tanks, the static configuration can be replaced with dynamic localization schemes such as in [10].

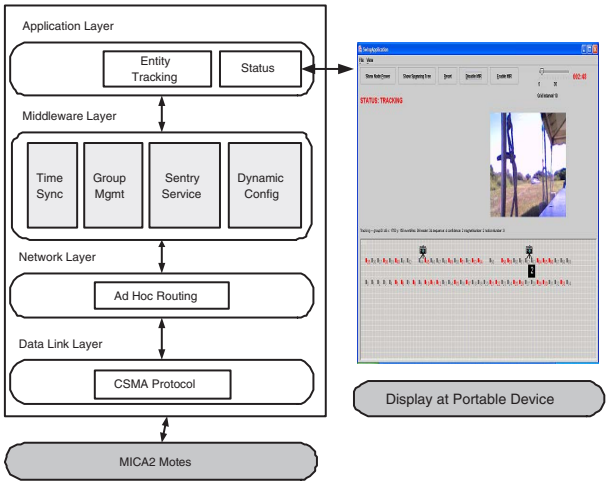

Figure 2: Energy-Efficient Tracking System

The routing component establishes routes through which the motes exchange information with each other and the base station.

Power management and collaborative detection are the two key higher-level services provided by our system. The sentry service component is responsible for power management, while the group management component is responsible for collaborative detection and tracking of events. The sentry service conserves energy of the sensor network by selecting a subset of motes, which we define as *sentries*, to monitor events. The remaining motes are allowed to remain in a low-power state until an event occurs. When an event occurs, the sentries awaken the other motes in the region and the group management component dynamically organizes the motes into groups in order to enable collaborative tracking. Together, these two components are responsible for energy-efficient event tracking.

All the deployed motes are programmed to run the distributed application. Our system supports the ability to reprogram the motes dynamically with new configuration parameters such as sensitivity . This eliminates the need to download the application code on all the motes each time the configuration is modified. We have a display module for portable devices (Figure 2)which is not part of the software that runs on each mote. We use it primarily for visu-

alization and debugging purposes. Optionally, the display software also has the logic to filter out any residual false alarms that have not been filtered out in the network. We now elaborate how the individual components of the system shown in Figure 2 interact with each other in the context of a typical tracking application. In particular, we discuss the design decisions that make the target system energy-efficient and illustrate trade-offs between performance and energy-awareness.

5. TIME-DRIVEN SYSTEM DESIGN

In our system, the MICA2 motes prepare for tracking by going through an initialization process. This process is used to synchronize the motes, set up communication routes, and configure the system with the correct control parameters. The initialization process proceeds in a sequence of phases and the transition between phases is time-driven, as shown in Figure 3. Phases I through IV comprise the initialization process which normally takes about 2 minutes. At the end of phase IV, the motes begin the power management and tracking activity. After performing this activity for a certain duration of time (e.g., one day), they begin a new system cycle. The duration of each phase is a control parameter that can be dynamically configured by the base station. Our multi-phase cyclic process satisfies following design objectives:

- First, it eliminates interference between operations. The constrained bandwidth in MICA2 doesn't allow a high concurrency in communication. If all operations run simultaneously, the traffic will severely interfere with each other.
- Second, we can confine the exposure of sensor activity within a short period time during the initialization phase (phase I to IV). As a result, the system can achieve zero exposure (complete stealthiness) during surveillance when no significant event happens.
- Third, a new system cycle is a natural way to allow the rotation of sentry responsibility among motes in order to achieve uniform energy dissipation across the network.
- Last, the cycling introduces system-wide soft-states. It allows the motes to periodically synchronize their clocks to avoid significant clock drifts over time. In addition, since mote failures and new deployment may occur anytime during a cycle, a new system cycle gives the remaining motes an opportunity to repair routes and discover new neighbors.

We now discuss the activities occurring during each phase of the system cycle in more detail.

5.1 Phase I: Basic Initialization

We observe that three functions in our system need system-wide broadcast: time synchronization, network backbone creation and system-wide reconfiguration. These functions can be isolated into three different modules that perform separately. However, such a design would not be bandwidth and energy efficient due to the multiple flooding phases required. Instead, we use a unique application-specific design to perform these operations simultaneously in one flooding operation to reduce overhead as described in following sections.

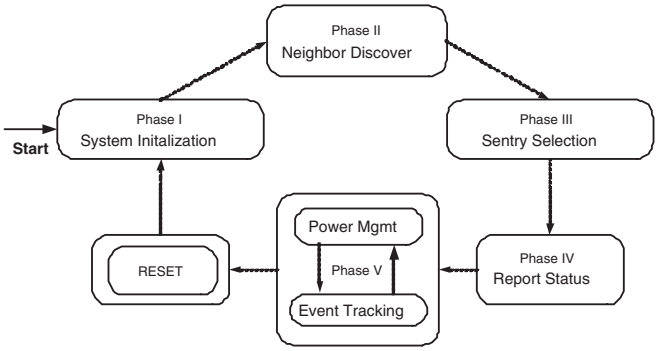

Figure 3: Time Driven System Transition

5.1.1 Time Synchronization

System initialization begins with time synchronization. Several schemes proposed recently are able to achieve a high synchronization precision, however they do not match well with our system requirements. GPS-based schemes typically achieve persistent synchronization with a precision of about 200 ns. However, GPS devices are expensive and bulky. The reference broadcast scheme (RBS) proposed in [5] maintains information relating the phase and frequency of each pair of clocks in the neighborhood of a node. The relation is then used to perform time conversion when comparing the timestamps of two different nodes. While RBS achieves a precision of about 1 μs, the message overhead in maintaining the neighborhood information is high and may not be energy-efficient in large systems.

We argue that fine-grained clock synchronization [5] achieved by costly periodic beacon exchanges may not be suitable for the energy-constrained surveillance system. Moreover continuous adjustment through beaconing in these solutions defeats our purpose of stealthiness. In our system, we value energy-efficiency and stealthiness above high synchronization precision. Therefore, we use a lightweight scheme that synchronizes the motes only during initialization phase, using a synchronization beacon broadcast by the base station at the beginning of each initialization cycle. Since the underlying MAC layer provided by TinyOS does not guarantee reliable delivery, the base station retransmits the synchronization beacon multiple times. Receivers take the timestamp from the beacon plus a transmission delay as their own local time. The synchronization beacons are propagated across the network through limited flooding with timestamp values reassigned at intermediate motes immediately prior to transmission. To satisfy the stealthiness requirement, we confine time synchronization within the initialization phase. The timer drift accumulated overtime is rectified by a new system cycle.

5.1.2 Diffusion Tree Creation

While the primary purpose of the synchronization message is to coordinate the clocks of the motes, it also serves as an exploratory message for motes to set up reverse routes to the base station, like the technique used by directed diffusion [15]. The route that is set up during the propagation of the time synchronization message is essentially a diffu-

sion tree rooted at the base station. The decision to use a diffusion tree is made based on several observations. 1) Sent along with the time synchronization operation, it is nearly free of cost in communication and code memory. 2) It allows any leaf motes to go to sleep without disrupting communication of other motes.

We encounter two practical issues when implementing the diffusion tree algorithm on the MICA2 platform.

- **Mote Failures:** The failure of a MICA2 mote can disable a subtree below it. Initially, we attempted to add failure detection to the MAC layer to quickly identify link failures and chose alterative routes. Soon, we discovered that link layer reliability in such a bandwidth constrained platform is too heavyweight and the effective data rate is reduced by nearly 50%. With such an observation, we introduce soft-state into the diffusion tree. The diffusion tree is refreshed per system cycle to prune failed links and discover new routes. After this modification, no bandwidth penalty is experienced during data communication.

- **Asymmetric Links:** If motes choose their parents without considering the distance separating them, it results in asymmetric links which leads to different reception rates along different directions between the same pair of motes. This asymmetry can be solved by link layer handshaking; however we discovered that it is very expensive. The practical strategy we adopt is that we use a lower transmission power when sending out synchronization messages to ensure that motes choose parents that are within a smaller radius. When transmitting application data, we use the maximum transmission power to ensure symmetric communication along the diffusion three in directions to and from the base station.

5.1.3 Dynamic Reconfiguration

The capability of dynamic reconfiguration facilitates retasking of sensor networks in future changes of mission requirements. Currently, this capability makes our work in system tuning and debugging much easier. When we deployed 70 motes on the field for the first time, it took us an hour to collect the motes and reprogram them manually, before the reconfiguration capability was added into the system. Now we can reconfigure the network within 1 minute. Our system supports reconfiguration with the help of the time synchronization message. The base station piggybacks the values of the control parameters in the synchronization message and motes adopt the new values when they accept the synchronization message. Such a strategy is energy-efficient, because it comes along with time synchronization beacons, obviating the need to send separate messages to reset parameters on the motes. Examples of control parameters that can be dynamically reconfigured include the duration of each phase shown in Figure 3, the duration for which a mote remains asleep and awake when power management is enabled, the sampling rate and the degree of in-network aggregation. This reconfiguration capability enables us to dynamically trade off between the energy-awareness and tracking performance as we show later in this paper.

5.2 Phase II: Neighbor Discovery

After the basic initialization phase, the motes make a transition to a neighbor discovery phase. Motes notify their neighbors by locally broadcasting `HELLO` messages. In the `HELLO` message, a sender sends its identifier, its status indicating whether it is a sentry or not, and the number of sentries that are currently covering it. The sender also identifies the sentry mote it reports to, if it is covered by at least one sentry. This local information is used to build a neighborhood table at each mote, and forms the basis for sentry selection in Phase III.

5.3 Phase III: Sentry Selection

In our sentry selection scheme, the decision to become a sentry is made locally by each mote, using the information gathered from its neighbors (The neighbor discovery goes through Phase II and III).

A mote decides to become a sentry if any one of the following conditions holds. 1) it is one of the internal nodes of the diffusion tree, or 2) it discovers that none of its neighbors either is a sentry or is covered by a sentry. When a mote decides to become a sentry, it advertises its intent. Three practical issues need to be solved to make this scheme work in a running system:

- **Race Conditions:** Contention occurs when multiple motes in the same neighborhood decide to become sentries at the same time. In order to reduce the collision probability, each mote uses a random backoff delay to transmit a `SENTRY DECLARE` message. If a mote receives a `SENTRY DECLARE` message from one of its neighbors during the backoff period, it updates its neighborhood table and cancels any pending outgoing `SENTRY DECLARE` messages. It then re-evaluates its decision to become a sentry based on the updated neighborhood information. If the mote finds that it is still necessary for it to become a sentry, it repeats the sentry declaration process described above.

- **Energy Balancing and Efficiency:** We set the backoff delay of a mote inversely proportional to its residual energy. Thus, a mote with higher residual energy has a greater likelihood of being selected as a sentry, thereby balancing the energy dissipation uniformly across the network. The backoff delay of a mote is also inversely proportional to the number of neighbors that are not covered by a sentry. Thus, motes in regions where there is insufficient sensing coverage are favored for being selected as sentries. The key feature of this sentry selection algorithm is that it provides an adaptive, self-configuring technique for choosing the sentries purely based on local information. However, the lack of global knowledge may result in a non-optimal number of sentries.

- **Sensing Coverage:** Surveillance addresses the sensing coverage problem of every physical point in the terrain, instead of communication coverage as in LEACH [11] and SPAN [3]. Since the sensing range of our Honeywell magnetometer [13] is much smaller than the Chipcon radio range, we need to use a smaller transmission power setting to send out `SENTRY DECLARE` messages in order to ensure sensing coverage. The power setting is chosen in such a way that there is

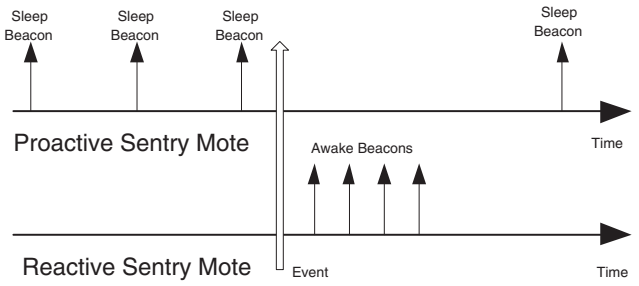

Figure 4: Two Power Management Schemes

at least one sentry within each sensing range. Unlike [22, 21], this unique design enables us to provide sensing coverage without the requirement of localization. More details can be found in the evaluation Section 7.1.

5.4 Phase IV: Status Report

After the routing backbone is finalized, all the motes use the backbone to report their status to the base station in Phase IV. The base station forwards those reports to the display module, which can then be used to visualize network topology, residual energy distribution and sentry distribution and detect any failed motes. Since the sole purpose of Phase IV is for visualization and debugging, it is optional.

5.5 Phase V-A: Power Management

The selection of sentries sets the stage for the power management phase. In this phase, the non-sentry motes alternate between sleep and wakeup states. A mote in the sleep state conserves power by disabling all processing, including those that are related to communication and sensing. We have proposed and implemented two different schemes to control the sleep-wakeup cycle. Now we discuss the pros and cons of these two schemes to clarify some practical issues

In the first implementation, which we call **proactive** control (Figure 4), the sentry mote sends out sleep beacons periodically. A non-sentry mote stays awake until it receives a beacon from its sentry mote, signaling the non-sentry mote to sleep for a certain duration of time. Upon receiving the sleep beacon, the non-sentry mote makes a transition to the sleep state and remains in that state for the specified amount of time. It wakes up when the timer expires and repeats the process by waiting for the next sleep beacon. Since neighboring non-sentry motes are likely to receive the same sleep beacon, their sleep-wakeup cycle proceeds in a lock-step fashion. The regular synchronization of the non-sentry motes with their respective sentries is beneficial in two ways. First, it allows multiple motes to receive the same beacon, and obviates the need to send out individual sleep beacons to put each non-sentry mote to sleep. This reduces the message overhead. Second, since motes in a neighborhood are all awake at the same time, the correlated sleep-wakeup cycle helps improve the tracking efficiency.

The second implementation to control the sleep-wakeup cycle is called the **reactive** control (Figure 4). In this scheme, the sentries are not required to send out explicit beacons to put the non-sentry motes to sleep. Instead, the transition between sleep and wakeup states is timer-driven. Each non-sentry mote remains awake for *awakeDuration* amount of time and then sleeps for *sleepDuration* amount of time. A non-sentry mote breaks out of its cycle and remains awake for a longer duration only when receiving an awake beacon from a sentry mote.

The reactive scheme is more stealthy compared to the proactive scheme, because no unnecessary beacons are sent unless an event occurs. Hence, the reactive approach is more appropriate for a surveillance application. However, one practical issue needs to be solved in the reactive scheme; since the non-sentries do not periodically synchronize their clocks with the clocks of their sentries, the clocks of the non-sentry motes may drift in course of time. Consequently, neighboring non-sentry motes may no longer have a sleep-wakeup cycle that is strictly in lock-step. As a result, a sentry no longer knows for certain which of its neighbors are awake. It has to retransmit the awake beacon multiple times in order to awaken non-sentries when an event occurs (Figure 4). We compare the message overhead between the proactive and reactive schemes in Section 7.3.1.

5.6 Phase V-B: Event Tracking and Reporting

After the sentry backbone has been created and power management is enabled, the motes are ready for tracking. Tracking and power management are toggle-states in phase V. When an event happens, motes wakeup and start tracking, when event disappears, motes toggle back to power management states.

A simple way to track events is by allowing each mote that has sensed an event to report its location and other relevant information about the event to the base station. The base station can then filter out the false alarms and infer the location of the event from the genuine reports. The advantage of this approach is that it allows all of the complex processing of the sensor readings to be deferred to the more powerful base station. However, the main drawback is that, if the motes are densely deployed, multiple motes may sense the event at the same time and send their individual reports to the base station. This results in higher traffic and wasteful expenditure of energy which can be reduced by aggregating multiple reports about the same event and sending a digest, instead of the individual reports to the base station. Previous in-network aggregation techniques fuse the data at the source through cluster headers [11] and/or along the route back to the sink [1][9][15][17]. In addition, Zhao [6] propose a optimal sensor selection approach to aggregate the fidelity of detections while eliminating redundant communication.

The system we have designed also performs in-network aggregation by organizing the motes into groups. However, distinguished from previous schemes, the groups in our work are more dynamic in the sense that they are formed in response to an external event and migrate when an event moves. A group represents an event uniquely and exists only as long as the event is in the scope of the sensor field. The design of our group management and tracking component is described in [2]. We review its key features here for completeness. It should be noted that the work reported in this paper is the first real implementation of the aforementioned design.

Each mote is programmed to detect an event by its sensory signature. This signature is a condition on the output of a filter that processes the raw sensory measurements (and removes noise). When the indicated condition is detected by a set of nearby motes, the group management compo-

nent reacts by creating a group. All motes that detect the same event join the same group. The main contribution of the group management component, described in [2], is to establish a unique one-one mapping between a group and a physical event as well as to maintain the membership of the group as the event moves through the environment. It is assumed that different events are far enough apart that membership of motes to the corresponding groups can be decided without ambiguity based on spatial adjacency to one of the events.

Each group is represented by a *leader* to the external world. Group members (who by definition can sense the tracked event) periodically report to the group leader. The leader records each report keeping only the most recent one from each member. Reports that are older than a certain threshold are purged. We define the confidence level of event detection as the number of distinct motes that have reported the event in the last t_r units of time. When the confidence level of detecting an event is at least as high as the threshold required by the application, called the *degree of aggregation* (DOA), the leader sends a digest of the reports to the base station. The confidence threshold can be tuned to manipulate the sensitivity of the system. A low threshold increases sensitivity at the expense of possible false alarms. A high threshold could result in missing some smaller targets. The effect of manipulating the degree of aggregation is explored experimentally in Section 7.2.2.

This concludes the description of our system design. In the next section, we discuss the implementation issues on the TinyOS / MICA2 platform.

6. IMPLEMENTATION

The architecture described in Section 4 was built on top of TinyOS [12]. TinyOS is an event driven computation model, written in NesC [7] specifically for the motes platform. TinyOS provides a set of essential components such as hardware drivers, scheduler and basic communication protocols. These components provide low level support for application modules, which are also written in NesC. NesC is a C-like language that enables the programmers to define the function of components and the relations (dependencies) among them. Components from TinyOS and user applications are processed by the NesC compiler into a running executable, which runs (in our case) on the MICA2 mote platform. MICA2 is the third generation mote built for wireless sensor networks [4]. Besides normal computation and communication capabilities, MICA2 motes have (i) selectable transmission power settings (255 levels) which enable us to dynamically adjust the communication range, (ii) a snooze function with up to six sleep modes provided by the ATmega128 Microcontroller, and (iii) a wireless reprogramming capability that eliminates the need for manual code downloads. The first two functions are utilized extensively by our protocols. The last facilitates deployment. In particular, we use a lower communication power setting during neighbor discovery for diffusion tree creation. This ensures that when the diffusion tree is created and communication power is subsequently increased, all found edges along the tree are quite reliable. In contrast, running diffusion tree creation at the normal power setting could result in unreliable or asymmetric edges between some nodes. This choice would ultimately reduce performance. The snoozing function is used to put motes into the power saving mode.

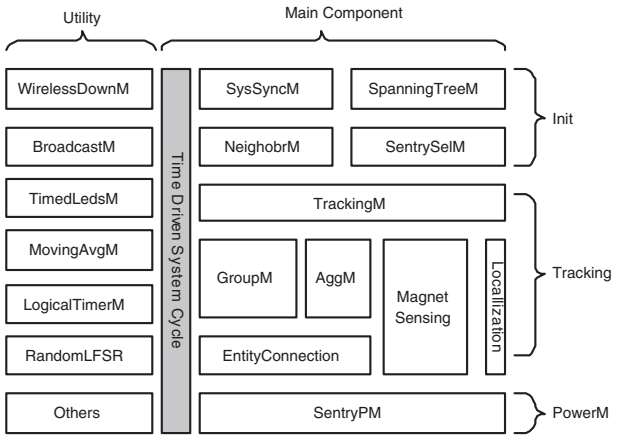

Figure 5: System Architecture

The implementation of our system on the MICA2 motes was driven by several requirements that arise from platform limitations. Namely:

- **Energy Efficiency:** MICA2 operates on a pair of batteries that approximately supply 2200 mAh at 3V. It consumes 20mA if running a magnetic sensing application continuously which leads to a lifetime of 5 days.
- **Bandwidth Efficiency:** The Chipcon radio on MICA2 provides an effective data rate of 12.4kbps, which equals a maximum packet rate of 43 pkts/sec. Our experiments show that a mote barely reaches 20 pkts/sec when it is exposed to channel contention.
- **Simplicity:** Our system requires many essential functions shown in Figure 5 to make target tracking efficient, while the whole system must fit in 4K data memory and 128K code memory. This necessitates a simple, yet effective, design for the MICA2 platform.
- **Flexibility:** Our prototype system spans 280 feet and comprises 70 motes. Once deployed, motes can not be easily collected. Dynamic configuration is desirable for fast performance tuning and debugging.

6.1 Software Architecture

The architecture of our system, written in NesC, is shown in Figure 5. The whole system occupies 39,496 bytes of code memory and 3,725 bytes of data memory. We divide system components into four major groups; initialization, tracking, power management, and general utilities. Initialization components are responsible for basic infrastructure establishment. Tracking components support the event tracking functions. The SentryPM module performs power management which puts motes to sleep as described earlier, when no significant events are detected. We also use some utilities to facilitate downloading, debugging, tuning and statistical logging. We provide a *backbone* module which is in charge of time-driven transitions between phases. We also use this module to pass state information among other modules to reduce the dependency among components.

In implementing the above architecture, several system challenges were met, primarily due to lack of common operating system support which TinyOS doesn't have. Some of the most important issues were the following:

Concurrency Control: TinyOS provides minimal support for concurrency control. The latest NesC compiler detects potential data races and give warnings at compile-time, however, it still requires the programmer to deal with it. Data race can be avoided by atomic sections or tasks. An atomic section is implemented through disabling and enabling interrupts. This requires the critical section to be very short. Otherwise, the system will become unresponsive. For example, if the soft timer cannot get updated by clock interrupts, time drift will happen. A better approach is to put all operations that access shared data into a task context. This guarantees sequential access to the data. However, the current task model doesn't allow parameter passing. The solution to this limitation is to put parameters into shared variables accessible by all tasks and use atomic sections to protect the read and write operation on these variables.

Packet Scheduling: For now, the TinyOS communication module doesn't provide a buffering mechanism. It is often the case that multiple components send out packets concurrently. All but one operation fails due to the mutual exclusion mechanism described above, used in the lower layer. The current solution we used is to provide application layer buffering. We reinitiate the transmission with linear backoff when contention happens.

Aggregation: The TinyOS communication module has a relatively high overhead. The packet header is 7 bytes (MAC header+ CRC) and the preamble overhead is 20 bytes in MICA2. For a default payload size of 29 bytes, the overhead to send a single packet is 48%! This limitation motivates us to use aggregation techniques. We use piggybacking whenever possible to increase the effective data rate. For instance, we piggyback system-wide parameters in time synchronization messages and piggyback sentry declaration information in neighbor beaconing. A more advanced aggregation technique such as in [9] is desired to efficiently use bandwidth.

Hardware Limitations: In general, the MICA2 platform is effective in supporting our system. However, in some cases, we have to modify our design to accommodate the limitations on hardware. First, the MICA2 mote has no circuit support for remote passsive wakeup. The current snooze implementation relies on a timer interrupt. This increases the chance of false negatives when the sleep duration of nonsentries is relatively long. Second, while the operating frequency of the Chipcon radio is selectable, external hardware attached to the chip can only support one frequency. This prevents us from designing a better collision avoidance algorithm to improve radio performance. Third, the current timer is supported through software which freezes when snooze is enabled. Though we can compensate for the lost time after a mote is awakened, more precise hardware timer support is desired.

Due to space limitations, here we only give a snapshot of the issues we encountered during the implementation. In general, we feel that platform-specific system designs are necessary to improve the performance.

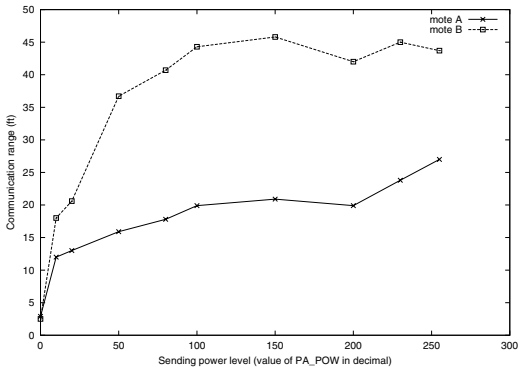

Figure 6: Impact of Sending Power on RF Range

7. PERFORMANCE EVALUATION

We now present experimental results that evaluate the performance of the physical system described in the previous section. We obtained most of the experimental results through an actual deployment of MICA2 motes in a grassy field, using the setup described in Section 3. However, for some experiments which require a long duration of time, we can not afford to deploy the system unattended due to security issues. Instead we conduct this type of experiments with a smaller number motes in controlled environments. In addition, simulations are also used to reveal the tradeoff between different design decisions.

We classify the experiments into three broad categories. The first set of experiments evaluate the MICA2 radio in different environments. The second set of experiments evaluate the performance of the tracking component. Finally, we evaluate the sentry service and the power management features of our system.

7.1 Evaluation of Capability of MICA2 Radio

The communication range of a MICA2 mote depends on several factors, such as the length of the antenna, the transmission power, the elevation above the ground, and the non-line-of-sight effects from objects in the surroundings (e.g., grass, trees, buildings, people, cars). Although the absolute values may vary in different environments, we can still draw some general observations about the MICA2 platform:

- We measure a set of MICA2 communication ranges under different sending power settings with two senders and one receiver. Results shown in Figure 6, indicate that 1) the communication range nonlinearly increases as the sending power increases. It increases more slowly when the power setting is large. 2) Asymmetry in communication range is more than what we expect, and it might primarily come from the differences in hardware calibration.

- We measure MICA2 communication ranges under different antenna lengths and different elevations above the ground. As expected, Table 1 indicates that longer antennas can significantly increase communication range in MICA2. Table 2 shows that the high elevation reduces floor attenuation, and hence increases RF range.

Table 1: Impact of Antenna Lengths on RF Range

Antenna	Power level = 50	Power level= 255
17.3 cm	37 ft	43 ft
34.6 cm	59 ft	> 84 ft

Table 2: Impact of Elevations on RF Range

Elevation	0 ft	0.5 ft	1 ft
Mote A	27 ft	30 ft	> 84 ft
Mote B	43 ft	> 84 ft	> 84 ft

7.2 Evaluation of In-Network Aggregation

In our experimental setup, we deployed 70 MICA2 motes along two sides of a road at a distance of 7-8 ft from each other. They were deployed densely in order to improve the data aggregation among motes.

Our goal is to track a car being driven along the stretch of road and study the impact of system parameters on the tracking performance. One key parameter is the degree of aggregation (DOA). This parameter decides the sensitivity of the surveillance system and is used to trade off between energy-awareness and surveillance performance. It is defined in our system as the minimum number of reports about an event that a leader of a group waits to receive from its group members, before reporting the event's location to the base station. In our implementation, the value of the DOA is dynamically configurable from the base station. We were interested in studying the impact of the degree of aggregation on the following metrics:

- the number of tracking reports (Figure 7),
- the number of false alarms generated (Figure 8), and
- the latency in reporting an event (Figure 9).

7.2.1 Impact of Aggregation on Transmission Overhead

In our tracking experiments we drove a car at a speed varying between 5-10 mph. We varied the degree of aggregation from 1 to 6 and repeated the tracking experiment for each value of DOA ten times. Figure 7 shows how the number of the tracking reports received by the base station varies with the DOA. From the figure, we see that when the value of DOA increases from 1 to 2, the number of tracking reports reduces by almost 50%. As the value of DOA increases even further, we observe that there is a steady drop in the number of tracking reports generated. These results verify the fact that the in-network aggregation, resulting from organizing the sensor motes into groups, significantly reduces the message overhead during tracking, and hence leads to much less energy consumption in data transmission.

7.2.2 Impact of Aggregation on False Alarms

Our next experimental result shows how the degree of in-network aggregation affects the false alarms generated when tracking an event. False alarms are normally caused by events such as burst distortions of readings due to power state transitions and incorrect readings from faulty sensors. Since a simulation-based approach normally assumes that sensors behave according to their specifications, such phenomena are usually not investigated in simulation. We classify false alarms into *false positives* and *false negatives*. A

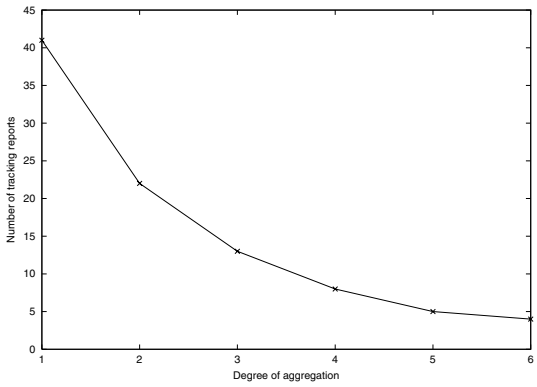

Figure 7: Impact of DOA on the Message Overhead

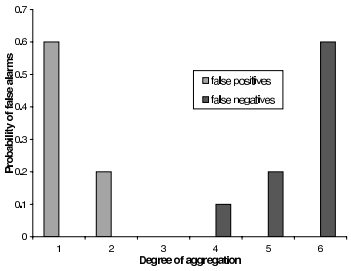

Figure 8: Impact of DOA on False Alarms

false positive occurs when a group of motes report the presence of the moving car in their neighborhood, when in reality, the car is not in their vicinity. A false negative occurs if the base station does not receive any reports of the car, although in reality, there is a car moving though the sensor field. In other words, if the car never appears on the display as it moves from one end of the sensor field to the other, we treat it as a false negative. It is important to emphasize that we do not consider a delayed report as a false negative.

We determined the probability of false alarms for each value of DOA by counting the number of false positives and false negatives we observed on the display during a set of 10 tracking rounds. Figure 8 shows how the probability of false positives and the probability of false negatives are each affected by the degree of aggregation. From Figure 8 we see that as the value of DOA increases from 1 to 6, the probability of false positives drops from 0.6 to 0, while the probability of false negatives increases from 0 to 0.6. These results can be explained as follows.

When the DOA = 1, the leader of a group reports the event to the base station, as soon as at least one member of the group detects the event. In an ideal scenario in which the sensing is perfect, even a single sensor reading should generate a high level of confidence. However, in practice, the sensor boards are sometimes inaccurate. This could result in an event being reported when it is not actually present. Hence, a single sensor reading may not be very reliable. One way to improve the reliability of event detection is to increase the redundancy, by either waiting for multiple reports from the same sensor mote (temporal redundancy), or by waiting for reports from multiple neighboring sensor motes (spatial

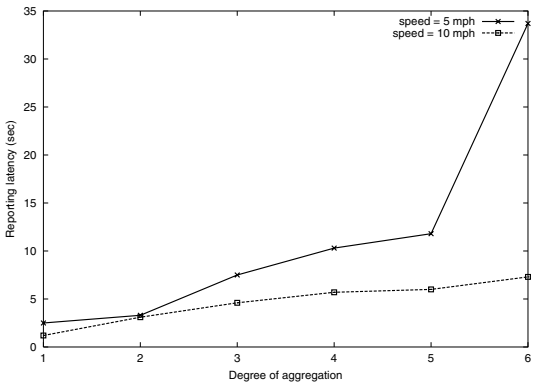

Figure 9: Impact of DOA on Reporting Latency

redundancy). We chose to experiment with the latter option because we assumed that the faults in the sensor boards are independently distributed. Therefore, the probability that multiple neighboring sensor motes are simultaneously in error is lower than the probability that a single sensor mote is in error. From Figure 8, we see that our assumption is validated. The figure shows that if the leader waits until at least 3 different sensor motes have detected the event, before reporting the event to the base station, the number of false positives drops to 0. However, if the sensing range and the density of deployment is not sufficiently high, it is harder to achieve a higher degree of aggregation. This results either in more false negatives, as shown in Figure 8, or in higher reporting latency as shown in the next section.

7.2.3 Impact of Aggregation on Tracking Latency

Figure 9 shows how the reporting latency increases with the degree of aggregation for a car moving at 5 mph through a sensor field where the motes are deployed 7-8 ft apart. We define the reporting latency as the time elapsed from the instant at which the car enters the sensor field until the instant at which the base station receives the first *genuine* report about the location of the car. In addition to the density, the increase in the latency and false negatives depends on the sleep cycle of the sensor motes and the speed of the moving vehicle. To our surprise, we found that we were able to reduce the latency and false negatives for higher degree of aggregation (DOA ≥ 4), by increasing the speed of the vehicle from about 5 mph to about 10 mph (Figure 9). However, increasing the speed beyond that value resulted in more false negatives. The reason is that when motes are some distance apart, a higher speed allows the vehicle to be in the sensing range of more motes during a period of time t_r. Hence, the vehicle can be detected even at a higher degree of aggregation. However, the sensors have a non-negligible reaction time, which further increases if the motes are sleeping. Hence, if the speed is increased beyond a certain threshold, the vehicle may move past the sensing range of the motes before they have a chance to react. That could result in more false negatives.

We must emphasize that the performance numbers we have presented above exhibit some degree of variance across different experimental runs and in different environments. Therefore, instead of using the above experimental results to deduce absolute performance numbers, we use them to draw some general conclusions about choosing the degree of in-network aggregation. First, a higher DOA certainly helps reduce the message overhead and the number of false positives. However, if the density with which the motes are deployed is not sufficiently high, a higher degree of aggregation may adversely affect the tracking performance. This effect is more pronounced in the case of slow-moving events. Even if the motes are densely packed and the events are fast-moving, it is harder to achieve a high degree of aggregation, if the motes sleep for a long duration and their sleep-wakeup cycles are not in lock-step. Thus, we see that the degree of aggregation represents a tradeoff between different parameters. The recommendation we follow based on our results is to choose a value of DOA that is large enough to maintain the probability of false negatives within a certain threshold. Our experiments show that a value of 2 or 3 for the degree of in-network aggregation is reasonable for MICA2 platform. If this value is not large enough to maintain the false positives within the desired threshold, then we recommend using a second tier of false alarm processing at the base station.

The above discussion motivates us to develop an analytical model in the future that captures the tradeoff between the key parameters, such as the degree of aggregation, density of node deployment, sleep duration, and the maximum probability of false alarms that a user can tolerate. Such a model can then be used to choose the appropriate degree of aggregation, when the values of the other parameters are known. Such a model is also valuable in estimating the probability of false alarms that a user can expect for a specific design and configuration.

7.3 Evaluation of Sentry Service

In this section, we analyze the key features of the sentry service component. We first analyze the stealthiness of the power management scheme, and then assess the extension in lifetime achieved for different sentry distributions and for different periods of the sleep-wakeup cycle of the non-sentries.

7.3.1 Stealthiness of Power Management Component

In Section 5.5, we compared and contrasted the proactive and reactive schemes for controlling the sleep-wakeup cycle of the non-sentry motes when power management is enabled. The proactive scheme provides better responsiveness when an event occurs, at the cost of transmitting more messages in the absence of an event. In contrast, the reactive scheme provides better stealthiness during the idle periods, at the cost of retransmitting multiple messages in order to awaken the non-sentries when an event occurs. A sentry chooses the interval between successive retransmissions in such a way that the beacon transmission coincides with the wakeup period of the neighboring non-sentry motes. We use the following equation to control the number of retransmissions of the awake beacon (n_r).

$$n_r = \frac{sleepDuration + awakeDuration}{awakeDuration + 1} \quad (1)$$

A larger value of *awakeDuration* results in fewer retransmissions of the awake beacon when a sentry detects an event. However, if the motes are awake longer, more energy is consumed and therefore, the lifetime of the sensor network reduces.

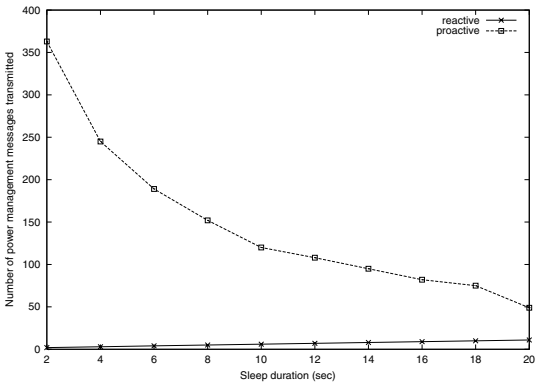

Figure 10: Power Management Message Overhead

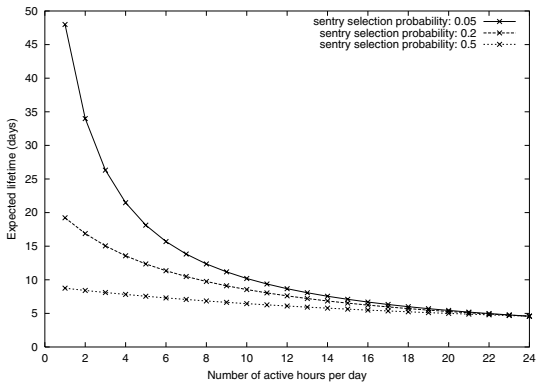

Figure 11: Expected Lifetime of a Sensor Network Using Sentry-based Power Management

Higher message overhead also translates to higher energy consumption. In order to compare the message overhead between the reactive and proactive schemes, we implemented both the schemes and conducted experiments using the Nido simulator[19], a simulator that actually runs our system and TinyOS code. We simulated a simple scenario in which a tank moved across a sensor field in which 10 motes capable of magnetic sensing were deployed. The duration of each simulation run was 600 seconds. The *awakeDuration* of the motes was fixed at 2 seconds for each run. Figure 10 compares the number of messages sent out by the proactive and reactive schemes during the tracking phase when power management is enabled.

Figure 10 shows that the number of power management messages in the reactive scheme increases from 2 to 11 as the sleep duration increases from 2 seconds to 20 seconds. This is justified by Equation 1, which indicates that a longer sleep duration requires more retransmissions of the awake beacon, in order to ensure that one of the beacons is received by the non-sentry motes. In contrast, the message overhead in the case of the proactive scheme reduces as the sleep duration increases. This is because the periodicity with which a sentry sends out the sleep beacon is equal to *sleepDuration* + *awakeDuration*. As the sleep duration increases, the sleep beacons are sent out less frequently, thereby reducing the message overhead.

The results in Figure 10 also show that the message overhead due to power management is significantly lower in the reactive scheme compared to its proactive counterpart. This suggests that the reactive scheme is more stealthy compared to the proactive scheme. While this is true for the 2 second awake period we have chosen, it may not be true for smaller values of *awakeDuration*. In our experiment, we chose a relatively high value of 2 seconds for *awakeDuration*, in order to compensate for the high rate of drift in the software timers in the current TinyOS implementation. If the timer drift is smaller in future implementations of TinyOS, we would choose a smaller awake duration for the motes, so that the overall energy consumption of the network can be reduced. However, a smaller value of *awakeDuration* would increase the message overhead for the reactive scheme. We have currently adopted the reactive scheme for our surveillance application, because it provides better stealthiness for the duration of the sleep-wakeup cycle we have chosen. However, an investigation into a hybrid scheme that combines the advantages of both the proactive and reactive schemes would be worthwhile to pursue as future work. In addition, the hardware solution mentioned in [8] might also be an alternative strategy for aggressive energy conservation.

7.3.2 *Power Savings*

One of the main goals of the sentry service module is the extension of the lifetime of the sensor network. The sentry service extends the lifetime by conserving the energy consumption of the motes when the network is idle. Non-sentry motes alternate between sleep and wakeup states, and in Section 7.3.1, we justified our choice of a timer-driven, reactive approach to control the sleep-wakeup cycle. When a mote is in the sleep state, its radio is turned off, all of its I/O ports are configured appropriately to minimize the current consumption, the ADC module is turned off to disable any sampling, and the controller is placed in a power-save state. When the sleep timer expires, the controller is awakened by a timer interrupt, and all of the modules resume activity. The extent to which our power management approach increases the lifetime of a mote depends on the fraction of time the mote spends in the sleep state. We now use the current consumed in the sleep and wakeup states using the above power management scheme to predict how the expected lifetime of a sensor network varies with the fraction of sentries selected.

A MICA2 mote is powered by a pair of AA batteries, supplying a combined voltage of 3V. Assuming that a pair of batteries will supply 2200 mAh at 3V [18], we can estimate the lifetime of a mote, if we know the current consumed in the sleep and wakeup states and the duty cycle of the mote. The duty cycle of a mote is the number of hours per day it remains awake polling for events. Based on our measurements, we found that a MICA2 mote equipped with a magnetic sensor board and running our sentry-based power management software consumes 20 mA in the wakeup state. The wakeup current includes the current consumed by the magnetometer to sample at a rate of 10 samples per second. On the other hand, we measured the sleep current of the mote to vary between 50 μA to 130 μA, which results in a 99% reduction in the current consumption. We use a sleep current of 130 μA for the discussion in this section.

From the above data, we can determine the lifetime of a sensor network that uses our sentry-based power management scheme. The lifetime of a sensor network depends on the fraction of sentries selected and the fraction of time the non-sentry motes remain awake. Let $P(s)$ denote the prob-

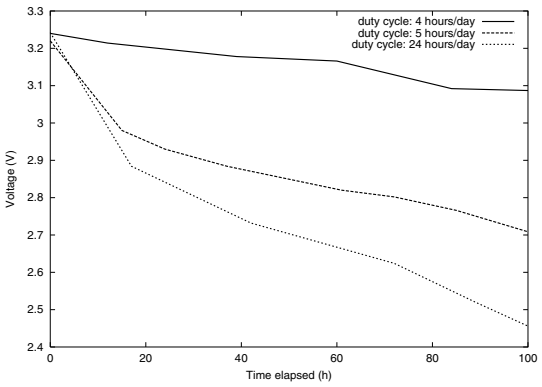

Figure 12: Impact of Sleep Duration on Power Consumption

ability that a mote is selected as a sentry, and $P(a)$ denote the probability that a non-sentry mote is awake. The total current (C) consumed by a mote in the baseline case, when there are no events in the network, is given by Equation 2. The lifetime of the motes, L, is the ratio of the battery capacity to the total current consumed. Assuming a battery capacity of 2200 mAh, the lifetime of the motes in hours is simply $2200/C$.

$$C = P(s) * 20 + (1 - P(s)) * (P(a) * 20 + (1 - P(a)) * 0.13) \quad (2)$$

Figure 11 uses the above equation to predict the expected lifetime of the motes for different percentages of their duty cycle. The actual values of $P(s)$ and $P(a)$ are measured from the our prototype system. A mote that is always asleep is expected to survive for 2 years, whereas a mote that is always awake (i.e. always remains a sentry), can survive only up to 5 days. The exponential curves show that the lifetime greatly improves when the duty cycle is low. For example, when the probability that a mote is selected as a sentry is 0.5, and its duty cycle is reduced from 24 hours per day to one hour per day, its lifetime extends by nearly 100%. The graphs also show that the lifetime improves significantly as the number of sentries is reduced. For example, when the probability that a mote is selected as a sentry is reduced to 0.05, and its duty cycle is reduced to 4%, its lifetime extends by nearly 900%. The probability of selecting a mote as a sentry involves a tradeoff between the sensing coverage that can be achieved and the required network lifetime. A higher probability results in more sentries and provides better sensing coverage. However, it also reduces the lifetime of the network, as Figure 11 shows. In order to reduce the number of sentries without adversely affecting the sensing coverage, we can either choose magnetometers with a higher sensing range or increase the density with which the motes are deployed. For example, in our experiments we found that when the motes were placed at a distance of 8 ft from each other, the probability that a mote was selected as a sentry was nearly about 40%. However, in a more dense deployment in which the motes were placed within a few inches from each other, the probability of selecting a mote as a sentry dropped to about 20%. The reason is that a dense deployment results in a larger number of neighbors for each mote. Therefore, a single sentry is able to cover more neighbors, and that gives fewer motes a chance to elect themselves as a sentry.

In addition to predicting the lifetime of the network using a simple model, we also conducted experiments to compare the rate at which energy is dissipated for differe nt duty cycles in an actual deployment. In each of our experiments we deployed 6 motes, all equipped with magnetic sensor boards, inside an office building. Sentry rotation occurred once every 4 hours. Since there is no direct way to measure the energy consumed by the motes, we used the voltage drop across the batteries supplying power to the motes as an indirect way to measure the energy dissipation. We measured the voltage for each mote at regular intervals over a period of 100 hours and found that the voltage drop was reasonably uniform across the motes. Figure 12 shows the voltage drop during the observation period for one of the 6 motes for different values of duty cycles. From the figure, we see that the battery voltage for a mote does not drop uniformly with time. One of the reasons for the non-uniform energy dissipation is the periodic rotation of the sentry responsibility. The voltage drop of a mote is higher during an interval in which it is serving as a sentry than when it is serving as a non-sentry because the periodic sampling operation performed by a sentry consumes significant energy. The results also confirm that a higher duty cycle results in a higher energy dissipation. We see that when the mote is always awake, it loses most of its capacity within 100 hours (about 4 days). This reasonably matches with the results in Figure 11, which predicted that a mote operating 100% of the time will last only 5 days.

The experimental results we obtained are promising in that they show that the sentry-based power management algorithm is adaptive and that it is successful in extending the lifetime of the sensor network. While our current sentry selection algorithm does not choose the minimal number of sentries, by knowing the lifetime of the mission in advance, we can choose the density of deployment and the duty cycle in such a way that the lifetime requirement can be met.

8. LESSONS LEARNED

The work described in this paper is our experience in building a complete system for using wireless sensor networks for a practical application, and evaluating it through an actual deployment of motes. This practical experience has been valuable, because it has taught us that some of the simplified assumptions made about the hardware platform and operating system in much current research do not hold well in practice. The lessons we learned have greatly impacted some of the design choices we had to make in building our system.

1. **Application-specific Reliability :** We found that the packet loss in the MICA2 platform can be as large as 20%. A well-known approach to counter message loss is to retransmit the message multiple times, in order to improve the probability of delivery. Such retransmissions can be initiated either in the lower layers of the protocol stack or at the application layer. Since retransmitting a message consumes significant energy, it is important that the messages are retransmitted selectively, based on application-specific knowledge. For instance, applications that transmit ephemeral sensor readings, such as the instantaneous temperature, may not require reliability. Lower layers, such as the MAC layer, often lack domain-specific knowledge. So im-

plementing reliability guarantees in the lower layers makes it harder to provide application-specific reliability. Hence, for a system that strives to achieve energy efficiency, providing reliability guarantees at the application layer is a better option.

2. **False Alarm Reduction:** We found that our sensors generated false alarms at a non-negligible rate. This introduces unnecessary energy consumption and inappropriate actions. False alarms we experienced can be categorized into two major types: Transient and persistent false alarms. A simple exponential weighted moving average (EWMA) on the mote is sufficient to deal with transient false alarms such as the burst distortion of sensing readings. However, if the false alarms are persistent due to errors in the sensor device, more advance techniques are desired. In our system, we successfully eliminated individual persistent false alarms by utilizing in-network aggregation with a relatively high DOA value. In the worst case, when multiple persistent false alarms are generated simultaneously, we are able to filter out such false alarms by analyzing spatial-temporal correlations among the consecutive reports at the base station.

3. **Race Conditions Reduction:** Race conditions are another example of a phenomenon that is often ignored in simulation-based approaches, but must be addressed when building the running system. For example, contention occurs not only when different motes try to transmit simultaneously, but also when different software components on the same mote initiate transmissions simultaneously through split-phase operations. Due to the limited support from TinyOS, the latter can lead to race conditions. Race conditions can be avoided, if the OS can support synchronized processing, based on semaphores, in order to coordinate the shared resources among the contending modules. While TinyOS supports concurrency control through atomic sections and tasks, it is more flexible and efficient to use application level synchronization such as packet scheduling mentioned in Section 6.1 to coordinate the operations.

4. **Asymmetry Reduction:** Another issue we had to address was to account for the effect of asymmetric channels which is largely ignored in simulation approaches. Communication in low power devices, such as the motes, is asymmetric [24] due to differences in hardware, signal attenuation, and residual battery capacity. In practice, we were able to reduce the effect of asymmetric channels by restricting a mote to communicate with only those neighbors that are well within its communication range. This can be achieved by reducing transmission power during the network establishment as we mentioned in Section 5.1.2. Moreover, it also can be achieved by bounding relay distance , if the localization is available.

5. **Software Calibration:** In a simulation-based approach, it is common for sensor devices of the same type to generate the same readings under identical conditions. However, in practice, the same type of sensors are capable of generating quite different sensor readings under identical conditions. Such a phenomenon may occur because of differences in the way the devices are manufactured, and it is often hard to accurately capture those differences in a simulator. We found that the impact of such heterogeneity is significant in the MICA2 platform, such as shown in Figure 6. The variance in the sensor readings can be accounted for at the very outset through software calibration of the sensors.

6. **Other Lessons:** The drift in the software timers in TinyOS presents another practical issue, especially when motes transit into sleep state. In order to compensate for the drift in the soft timers, we need to increase the duration for which a mote remains awake, and design appropriate strategies to control the sleep-wakeup cycle, as described in Section 7.3.1. Another practical challenge we faced was the lack of appropriate tools for debugging a network of motes. We utilize the dynamic configuration method mentioned in 5.1.3 and overhearing tools to facilitate our work. However, more sophisticated debugging and configuration tools will greatly ease the burden on the programmer in the future. We acknowledge that our design choices sometimes are restricted by limited hardware and operation system support. It is desirable to have new features such as interruptible snoozing, sub-controllers for I/O, a more reliable RF module and process management, so that we can improve our design and implementation in the future.

9. CONCLUSIONS

Research in wireless sensor networks has been very active. Most of the published work studies an individual protocol and performs evaluations via simulations. In contrast, in this work we implement an entire integrated suite of protocols and application modules and evaluate the performance on a system composed of 70 MICA2 motes in a realistic outdoor setting. Empirical results identify the capability of the MICA2 radio, the value of in-network aggregation with respect to transmission overhead, false alarm processing and application layer tracking latency, and the value of power management. Design decisions and how those decisions were influenced by the empirical data were described. Key lessons learned were also itemized. From our experience in building and analyzing this system it is clear that key realistic hardware, software and environmental issues must not be ignored in developing usable solutions. This includes realism of sensor performance, asymmetries in communication, false alarms, and race conditions.

10. FUTURE WORK

System design and engineering is one of the keys to bring sensor network paradigm into reality. The system described in this paper is still an ongoing prototype. Many outstanding design issues are yet to be resolved. We are currently investigating 1) target classification under constraint resources through collaborative data fusion, 2) the possibility to design a more aggressive power management strategy with passive wake-up capabilities [8], 3) approaches to build extremely robust routing infrastructure, which can survive under hostile environments, 4) a practical localization scheme and 5) a scalable architecture up to thousands of nodes while maintaining operational performance requirement.

11. REFERENCES

[1] S. Bhattacharya, H. Kim, S. Prabh, and T. Abdelzaher. Energy-Conserving Data Placement and Asynchronous Multicast in Wireless Sensor Networks. In *The First International Conference on Mobile Systems, Applications, and Services (MobiSys)*, May 2003.

[2] B. M. Blum, P. Nagaraddi, A. Wood, T. F. Abdelzaher, S. Son, and J. A. Stankovic. An Entity Maintenance and Connection Service for Sensor Networks. In *The First Intl. Conference on Mobile Systems, Applications, and Services (MobiSys)*, May 2003.

[3] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *6th ACM MOBICOM Conference*, 2001.

[4] CrossBow. *Mica2 data sheet*. Available at `http://www.xbow.com/Products/Product_pdf_files/MICA%20data%20sheet.pdf`.

[5] J. Elson and K. Romer. Wireless Sensor Networks: A New Regime for Time Synchronization. In *Proc. of the Workshop on Hot Topics in Networks (HotNets)*, October 2002.

[6] Z. Feng, S. Jaewon, and R. James. Information-Driven Dynamic Sensor Collaboration for Target Tracking. *IEEE Signal Processing Magazine*, 19(2), Mar 2002.

[7] D. Gay, P. Levis, R. von Behren, M. Welsh, E. Brewer, and D. Culler. The nesC Language: A Holistic Approach to Networked Embedded Systems. In *Proceedings of Programming Language Design and Implementation (PLDI) 2003*, 2000.

[8] L. Gu and J. A. Stankovic. Radio-Triggered Wake-Up Capability for Sensor Networks. In *Proceedings of RTAS*, 2004.

[9] T. He, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher. AIDA: Adaptive Application Independent Data Aggregation in Wireless Sensor Networks. *ACM Transactions on Embedded Computing System, Special issue on Dynamically Adaptable Embedded Systems*, 2004.

[10] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-Free Localization Schemes in Large-Scale Sensor Networks. In *Proc. of the Intl. Conference on Mobile Computing and Networking (MOBICOM)*, September 2003.

[11] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proc. of the Intl. Conference on System Sciences*, January 2000.

[12] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister. System Architecture Directions for Networked Sensors. In *Proc. of Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pages 93–104, 2000.

[13] Honeywell. *1- and 2-Axis Magnetic Sensors*. Available at `www.ssec.honeywell.com/magnetic/datasheets/hmc1001-2_1021-2.pdf`.

[14] M. Horton, D. E. Culler, K. Pister, J. Hill, R. Szewczyk, and A. Woo. MICA: The Commercialization of Microsensor Motes. *Sensors Online*, April 2002. www.sensorsmag.com/articles/0402/40.

[15] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *the Sixth Annual International Conference on Mobile Computing and Networks*, 2000.

[16] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next Century Challenges: Mobile Networking for Smart Dust. In *Proc. of Intl. Conference on Mobile Computing and Networking (MOBICOM)*, August 1999.

[17] S. Madden, M. Franklin, J. Hellerstein, and W. Hong. TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks. In *Operating Systems Design and Implementation*, December 2002.

[18] A. Mainwaring, J. Polastre, R. Szewczyk, D. E. Culler, and J. Anderson. Wireless Sensor Networks for Habitat Monitoring. In *Proc. of the ACM Workshop on Sensor Networks and Application (WSNA)*, September 2002.

[19] TOSSIM: A Simulator for TinyOS Networks. Available at `webs.cs.berkeley.edu/tos/tinyos-1.x/doc/nido.pdf`.

[20] R. Powers. Batteries for Low Power Electronics. *In Proceedings of the IEEE*, pages 687–693, April 1995.

[21] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill. Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks. In *First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[22] T. Yan, T. He, and J. Stankovic. Differentiated Surveillance Service for Sensor Networks. In *First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[23] F. Ye, G. Zhong, S. Lu, and L. Zhang. PEAS: A Robust Energy Conserving Protocol for Long-lived Sensor Networks. In *Proc. of International Conference on Distributed Computing Systems (ICDCS)*, May 2003.

[24] G. Zhou, T. He, and J. A. Stankovic. Impact of Radio Irregularity on Wireless Sensor Networks. In *The Second International Conference on Mobile Systems, Applications, and Services (MobiSys)*, June 2004.