Achieving Real-Time Target Tracking Using Wireless Sensor Networks

Tian He[§], Pascal Vicaire[†], Ting Yan[†], Liqian Luo[‡], Lin Gu[†], Gang Zhou[†], Radu Stoleru[†], Qing Cao[‡], John A. Stankovic[†] and Tarek Abdelzaher[‡]

[†]Department of Computer Science, University of Virginia
[§]Department of Computer Science and Engineering, University of Minnesota
[‡]Department of Computer Science, University of Illinois, Urbana-Champaign

Target tracking systems need to meet certain real-time constraints in response to transient events, such as fast-moving targets. While the real-time performance is a major concern in these applications, it should be compatible with other important system properties such as energy consumption and accuracy. This work presents the real-time design and analysis of VigilNet, a large-scale sensor network system which tracks, detects and classifies targets in a timely and energy efficient manner. Based on a deadline partition method and theoretical derivations to guarantee each sub-deadline, we are able to make guided engineering decisions to meet the end-to-end tracking deadline. The results from 10,000-node simulation and 200 mote field test reveal the effectiveness of our design.

Categories and Subject Descriptors: C.3. [**Special-Purpose and Application-Based Systems**]: Real-Time and Embedded Systems; C.2. [**Computer Communication Networks**]: Distributed Systems

General Terms: Design, Performance, Experimentation

Additional Key Words and Phrases: Sensor Networks, Real-Time, Tracking, Energy Conservation

1. INTRODUCTION

Recent developments in sensor techniques make wireless sensor networks (WSNs) available to many application domains [Dutta et al. 2005; He et al. 2004; Juang et al. 2002; Simon et al. 2004; Xu et al. 2004; Arora and et al. 2005]. Most of these applications, such as battlefield surveillance, disaster and emergency response, deal with various kinds of real-time constraints in response to the physical world. For example, surveillance may require a sensor node to detect and classify a fast moving target within 1 second before it moves out of the sensing range. Compared with the traditional distributed systems, the real-time guarantee for sensor networks is more challenging due to the following reasons. First, sensor networks directly interact with the real world, in which the physical events may exhibit unpredictable *spatiotemporal* properties. These properties are hard to characterize with traditional methods. Second, although the real-time performance is a key concern, it should be performance compatible with many other critical issues such as energy efficiency and system robustness. For example, it is not efficient to activate the sensors all the time only for the benefit of a fast response. This naive approach severely reduces

the system lifetime [He et al. 2004]. Third, the resource constraints restrict the design space we could trade off. For example, the limited computation power in sensor nodes makes the Fast Fourier Transformation not quite suitable for real-time detection. All these issues challenge us with two questions. *How to make the design of a large-scale real-time sensor network system manageable?* And *how to trade off among system metrics while maintaining the real-time guarantee?* Our answer to these questions, presented in this paper, is a case study of the VigilNet system, a real-time outdoor tracking system using a large-scale wireless sensor network.

Our contribution lies in the following aspects: 1) This work addresses a real-world application with a running real-time system, designed and implemented over the last few years. 2) We demonstrate how to guarantee the end-to-end tracking deadline in a complex sensor system. For a given sub-deadline partition, we identify the system configurations that meet the sub-deadlines without compromising other important system properties. 3) The real-time design and tradeoffs are validated by a large-scale field evaluation with 200 XSM motes and an extensive simulation with 10,000 nodes. These evaluations reveal quite a few practical design suggestions that can be applied to other real-time sensor systems.

The remainder of the paper is organized as follows: Section 2 introduces the tracking process in VigilNet. Section 3 identifies the real-time requirements. Section 4 provides a real-time analysis of VigilNet's tracking performance and its tradeoffs. In Section 5 we describe the implementation of the VigilNet system. In Section 6, we evaluate the real-time performance of VigilNet in an outdoor field test. In Section 7, we conduct a large-scale simulation to further validate and analyze the real-time issues in VigilNet. Section 8 discusses the related work. Section 9 concludes the paper.

2. OVERVIEW OF VIGILNET TRACKING OPERATIONS

VigilNet is an energy-efficient surveillance and tracking system, designed for spontaneous military operations in remote areas. In these areas, the events of interest happen at a relatively low rate, i.e. the duration of significant events (e.g., intruders) is very short, compared with the overall system lifetime (e.g., 5-minute tracking per day). According to our empirical results [He et al. 2006], nearly 99% of energy is consumed during the idle-waiting period for potential targets. Therefore to conserve energy, the most effective approach is to selectively turn a subset of nodes off, and wake them up on demand in the presence of significant events. This power management technique fundamentally shapes the VigilNet tracking process. It introduces a set of new delays that traditional tracking systems do not experience.

In this section, we give a brief overview of the VigilNet tracking operation, serving as a background for the real-time design and analysis in the following sections. As shown in Figure 1, after a target enters the area, it activates the first sensor node that can confirm the detection, then other nodes nearby are awakened to form a group to deliver the aggregated reports to the base. More specifically, the VigilNet tracking operation has six phases:

A. **Initial Activation:** VigilNet stays in the power management state when there are no targets. The power management protocol puts nodes into either one of two states: *sentry* and *non-sentry*. In brief, a node becomes a sentry node

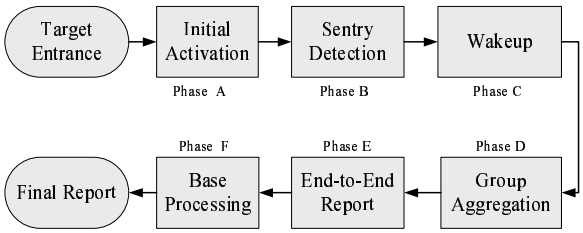

Fig. 1. The Delay Breakdown in Tracking Operation

if it is a part of the routing infrastructure or it needs to provide the sensing coverage. Otherwise, it becomes an inactive non-sentry node. The details of sentry selection can be found in [He et al. 2004]. If the sentry nodes are awake 100% of time (i.e. the deployed area is always covered), any incoming target is covered by at least one sentry node immediately. On the other hand, if the sentry nodes have a certain duty cycle (i.e. they go to sleep and wake up periodically to save energy), there will be an initial activation delay, denoted as $T_{initial}$, before the first sentry node starts to sense the incoming target.

B. **Initial Target Detection:** After the initial activation, it takes a certain delay, defined as $T_{detection}$, for the first sentry node to *confirm* the detection. This delay consists of the hardware response delay, the discrete sampling delay and the delay to accumulate a sufficient number of samples before a detection algorithm recognizes the target.

C. **Wake-up:** Normally, the detection from a single sentry node does not provide sufficient confidence in detection and classification, therefore a group-based tracking is designed in VigilNet. In order to form a group with a reasonable size, non-sentry nodes need to be awakened after the initial target detection by a sentry node in Phase B. We define the wake-up delay T_{wakeup} as the time required for a sentry node to wake up other sleeping non-sentry nodes. This delay is determined by the toggle period of none-sentry nodes.

D. **Group Aggregation:** Once awakened, all nodes that detect the same target join the same logic group in order to establish a unique one-to-one mapping between this logic group and the detected target. Each group is represented by a leader which maintains the identity of the group as the target moves through the area. Group members (who by definition can sense the target) periodically report to the group leader. A leader reports a detection to the base after the number of member reports exceeds a certain threshold, defined as the degree of aggregation (DOA). We use $T_{aggregation}$ to denote the group aggregation delay, which is the time required to collect and process the detection reports from the member nodes.

E. **End-to-End Report:** After group aggregation, the leader node reports the event to the nearest base. Multiple bases are used to partition a network into several sections, in order to bound the end-to-end delivery delay T_{e2e}.

F. **Base Processing (T_{base}):** A base is in charge of processing the reports from the leaders of different logic groups. Since the reports from the same logic group are spatiotemporally correlated, a string of consecutive reports can be

further analyzed and summarized for end users. For example, taking the time stamps and the locations of targets as the inputs, a base uses the least-square estimation to obtain the velocity of each target.

3. REAL-TIME REQUIREMENTS IN VIGILNET

To ensure the effectiveness of target tracking, VigilNet must meet a certain real-time constraint. Specifically, VigilNet should detect, classify and analyze the incoming targets within a certain end-to-end deadline (e.g., 5 seconds from Phase A to F). As shown in Section 2, the end-to-end deadline is affected by many system parameters, which form a high-dimensional design space where the number of possible configurations increases exponentially with the number of system parameters. It is intractable to identify a system-wide global optimal solution within this design space. Therefore, we adopt the deadline partition method to divide the end-to-end deadline into multiple sub-deadlines. By confining the real-time decisions within each phase, we make the end-to-end analysis manageable in a lower-dimensional design space. For a given end-to-end deadline, a designer can make an initial partition, according to the workload, system resources available and preliminary estimation of the delay in each phase. As a concrete example, supposing a target enters the field with a speed up to 20 mph, to guarantee that this target can be detected by the first sentry node with a probability higher than 90%, we need to design a detection algorithm with a sub-deadline less than 1 second, assuming the detection range is 10 meters. After the initial partition, one can identify a system configuration to guarantee the sub-deadlines, based on the analysis in this work. As long as the individual sub-deadlines are met, we have a certain guarantee on the end-to-end delay.

4. VIGILNET REAL-TIME TRACKING ANALYSIS

The description in this section follows the natural order of VigilNet's tracking operations presented in Section 2. Such design and analysis is validated later with a real system implementation consisting of 200 XSM nodes as well as a large-scale simulation in Section 6 and 7, respectively.

4.1 Initial Activation Delay and Its Tradeoffs

In a duty-cycle-based power management scheme, the sentry nodes go to sleep and wake up periodically. In this case, the initial activation delay $T_{initial}$ may not be zero, because the sentry nodes near the target's entry point may be asleep when the target enters the field. In this section, we identify a quantitative relationship between the energy savings and the $T_{initial}$, which helps us make decisions to guarantee that the initial activation finishes within a given sub-deadline $D_{initial}$.

In our VigilNet design, all sentry nodes agree on a common sentry toggle period P and a common sentry duty cycle SDC. The starting time of a period is randomized in each node. For each period, a sentry wakes up and stays awake for $P \cdot SDC$, then goes to sleep for $P \cdot (1 - SDC)$. Assuming a target enters the tracking area from point s for L meters till it reaches the point e, as shown in Figure 2(a), we first derive P_r, the probability that a single sentry node detects this target. Obviously, the nodes that may detect the target must be in the rectangle or the

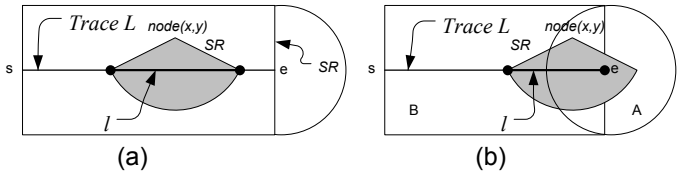

Fig. 2. Detection Probability for fast targets

semi-circle shown in the Figure 2(a). The size of the area is $2SR \cdot L + \pi \cdot SR^2/2$, where SR is the Sensing Range. For a single node located at (x, y) in this area, the probability that the node detects the target $P(x, y)$ is $SDC + l(x, y)/(P \cdot TS)$ if $SDC + l(x, y)/(P \cdot TS) \leq 1$, where $l(x, y)$ is the overlapping length of the node's sensing range and the target's trace, and TS is the Target Speed. If we consider all possible locations in this area, we can get P_r in Equation 1 by integrating and normalizing $P(x, y)$ over the area. We note that when (x, y) is in the circle (area A), as shown in Figure 2(b), $l(x, y) = \sqrt{SR^2 - y^2} + L - x$. When (x, y) is in area B, $l(x, y) = 2\sqrt{SR^2 - y^2}$.

$$\begin{aligned} P_r &= \frac{\int_A (SDC + \frac{\sqrt{SR^2-y^2}+L-x}{P \cdot TS})ds + \int_B (SDC + \frac{2\sqrt{SR^2-y^2}}{P \cdot TS})ds}{(2SR \cdot L + \pi SR^2/2)} \\ &= SDC + \frac{\pi \cdot SR \cdot L}{(2L + \pi \cdot SR/2) \cdot TS \cdot P} \end{aligned} \tag{1}$$

We note that P_r calculated by Equation 1 is valid only when the target speed is faster than or equal to $2SR/(P - P \cdot SDC)$. We define this as a *fast target*. For a target with a speed slower than $2SR/(P - P \cdot SDC)$, which we define as a *slow target*, it may happen that for a node located at (x, y), the corresponding $l(x, y)$ is greater than $(P - P \cdot SDC) \cdot TS$, so that $SDC + l(x, y)/(P \cdot TS) > 1$. For the node at this location, the probability that it detects the target is 1 instead of $SDC + l(x, y)/(P \cdot TS)$. Therefore, we need to revise the result in Equation 1 for slow targets. We define variable a such that $SDC + 2a/(TS \cdot P) = 1$. In Figure 3, it can be easily proven that if a node appears inside area C bounded by the dashed arc and lines, the probability that it detects the target is 1. Note that the distance between the dashed line and the target trace is $\sqrt{SR^2 - a^2}$. The dashed arc is centered at $(L - 2a, 0)$ and its radius is SR. The rest of the area is divided by the circle with radius SR centered at $(L, 0)$ into area A' and area B'. The detection probabilities for nodes in area A' and area B' have the same forms as those for nodes in area A and area B in the fast target case, correspondingly. Then we have

$$\begin{aligned} P_r &= \frac{\int_{A'} (SDC + \frac{\sqrt{SR^2-y^2}+L-x}{P \cdot TS})ds + \int_{B'} (SDC + \frac{2\sqrt{SR^2-y^2}}{P \cdot TS})ds + \int_C 1ds}{(2SR \cdot L + \pi SR^2/2)} \\ &= SDC + \frac{\pi \cdot SR^2 \cdot L + min[(L-a)k(SR,a), 0]}{(2SR \cdot L + \pi \cdot SR^2/2) \cdot TS \cdot P}, \end{aligned} \tag{2}$$

in which $k(SR, a) = 2a\sqrt{SR^2 - a^2} - 2SR^2 \cos^{-1}(a/SR)$.

In the paper, we omit the intermediate derivation, for those interested, more information can be found at [Cao et al. 2005].

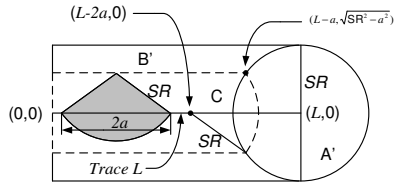

Fig. 3. Detection Probability for slower targets

Now we are ready to provide a statistical real-time guarantee for the initial activation process, i.e., we need to ensure a target is detected before the sub-deadline $D_{initial}$. Equivalently, a target should be detected before it enters for $L = TS \cdot D_{initial}$ meters. Obviously, $P(T_{initial} < D_{initial})$ equals $P(T_{initial} \cdot TS < L)$, where $P(T_{initial} \cdot TS < L)$ is the probability that at least one of nodes in the area (A+B) detects the target. If there are n nodes in the area, the probability that at least one of them detects the target is $1 - (1 - P_r)^n$. Suppose the sentry density is D_s and n conforms to a Poisson distribution with parameter λ $=(2SR \cdot L + \pi \cdot SR^2/2)D_s$, therefore, the probability that the initial activation finishes before sub-deadline $D_{initial}$ is:

$$P(T_{initial} < D_{initial}) = P(T_{initial} \cdot TS < L) = 1 - e^{-P_r \cdot \lambda} \tag{3}$$

Equation 3 identifies a feasible region for us to decide the system parameters such as sentry duty cycle (SDC) and sensing range (SR) to ensure the real-time property in Phase A. In addition, we can obtain the expected value of $T_{initial}$ from the formula $E(T_{initial}) = \int_0^\infty (1 - P(T_{initial} < t))dt = \int_0^\infty (1 - P(SD < TS \cdot t))dt$. According to Equations 1 and 3, we have the expected delay for a target whose speed is greater than or equal to $2SR/(P - P \cdot SDC)$:

$$E(T_{initial}) = \frac{e^{-SDC \cdot \pi \cdot SR^2 \cdot D_S/2}}{(2SR \cdot SDC \cdot TS + \pi SR^2/P)D_S} \tag{4}$$

Similarly, for a target whose speed is lower than $2SR/(P - P \cdot SDC)$, we have

$$E(T_{initial}) = \frac{e^{-SDC \cdot \pi \cdot SR^2 \cdot D_S/2}[1 - \frac{k(SR,a)e^{-(2SR \cdot SDC \cdot TS \cdot P)(1-SDC)D_s/2}}{2SR \cdot SDC \cdot TS \cdot P + \pi SR^2 + k(SR,a)}]}{(2SR \cdot SDC \cdot TS + \pi SR^2/P)D_S} \tag{5}$$

One caveat in the analysis needs some attention. Above we derive the expected detection delay for a duty cycle based system with *random deployment*. However, sentry nodes are located more evenly than totally randomly case [He et al. 2004]. Fortunately, we can prove that the random deployment case provides a theoretical upper bound for the sentry-based deployment case. It can be easily proved that if for all t, $P(T_1 < t) > P(T_2 < t)$, we must have $E(T_1) < E(T_2)$. For $0 < P_r < 1$, $1 - (1 - P_r)^n$ is a strictly concave function of n. Therefore, $E(1 - (1 - P_r)^n) \leq 1 - (1 - P_r)^{E(n)}$, and the left side of the equation equals the right side if and only if n is a constant. Given the same $E(n)$, the more scattered the distribution of n is, the smaller the value of $E(1-(1-P_r)^n)$ is. Since the sentry nodes are selected more uniformly than the random case, $P(T_{initial} < D_{initial})$ for the sentry based system is greater than a totally randomly distributed system, and therefore the expected

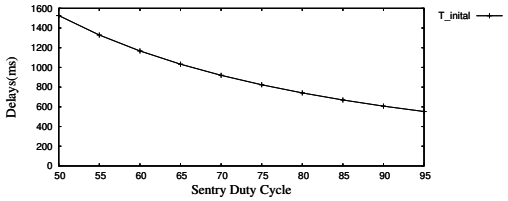

Fig. 4. Initial Delay vs. SDC

delay is smaller. The expected delay for the random case can be used as an upper bound for the expected detection delay for a more evenly distributed system. Later, we will see from the simulation that the analytical result overestimates the $T_{initial}$ by 15%.

We can further take the detection delay $T_{detection}$ into account, since a successful detection in Phase B activates a full tracking process. In this case, we establish an equivalent model for $T_{initial}$. Specifically, in Equation 4 or Equation 5, we substitute SDC with the effective sentry duty cycle $SDC_{eff} = SDC - T_{detection}/P$ and substitute SR with the effective sensing range $SR_{eff} = \sqrt{SR^2 - (T_{detection} \cdot TS/2)^2}$. Figure 4 gives a more concrete view of the tradeoff between SDC and expected $T_{initial}$. We take parameters from the real VigilNet implementation: $D_S = 0.01 node/m^2$, $P = 10s$, $SR = 10m$, $TS = 10m/s$ and $T_{detection} = 1000ms$. This result is consistent with what we obtained from the real experiments and simulations.

4.2 Sentry Detection Delay and Its Tradeoffs

After the initial delay in Phase A, a target approaches the vicinity of a sensor which begins to observe a different signal pattern than that without a target. With the current sensing algorithms, the signal pattern can be amplitude, frequency, or a combination of the two. We call the signal pattern corresponding to a target *a target signature*. The recognition of a target signature indicates a sensor-level detection, and produces data for higher-level detection and classification algorithms.

As defined before, $T_{detection}$ is the time for a detection algorithm to recognize a target signature. This delay must be smaller than a certain sub-deadline $D_{detection}$. Multiple reasons contribute to this delay. First, the sensor hardware has a response delay for the physical signals that the target generates. Second, the sensing circuitry requires special operations with a further delay. For example, the magnetometer in MICA2 node [CrossBow 2003] takes about $35ms$ to stabilize after the potentiometer adjustment. Third, the sampling is discrete and periodic, not continuous, which leads to sampling delay. Fourth, the target signature itself may be time related (e.g., a certain frequency), which can not be recognized from just one sample. Finally, the VigilNet system is designed for outdoor deployment. It must adapt to environmental noise and dynamics, such as the change of temperature, the motion of small plants on windy days, and the sound of animals. Hence, noise filtering is a key step in the recognition of the target signal pattern. Such filtering usually needs to accumulate and analyze measurements over a period of time, and imposes a delay in detection.

Now we describe how to decide the sub-deadline $D_{detection}$. Obviously, a detection algorithm must finish before a target moves out of the sensing range of a node.

Suppose that the nominal sensing area is a circle with a fix sensing range SR, the amount of time a target stays in a node's sensing range can be derived from the speed of the target, TS, and the minimum distance from the target's trajectory line to the sensor node. Since the target trajectory intersects with the sensing circle randomly, we assume this minimum distance is uniformly distributed within $[0, R)$, therefore the probability that a target stays in one sensing circle for at least $D_{detection}$ seconds can be calculated as

$$\begin{array}{ll} P(t > D_{detection}) = \sqrt{1 - \frac{(TS \cdot D_{detection})^2}{4SR^2}} & D_{detection} < \frac{2SR}{TS} \\ P(t > D_{detection}) = 0 & D_{detection} \geq \frac{2SR}{TS} \end{array} \quad (6)$$

According to Equation 6, the sub-deadline $D_{detection}$ can be decided by choosing a desired $P(t > D_{detection})$ value.

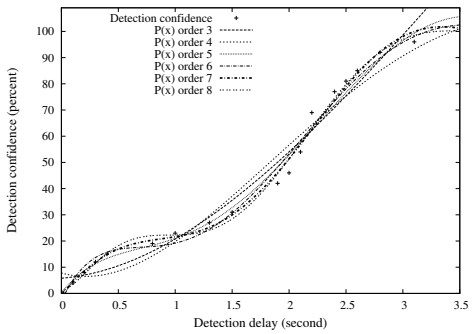

Fig. 5. Detection Confidence vs. Detection Delay

In addition, we desire to know how a detection algorithm performs under a given sub-deadline $D_{detection}$. We define the *Detection Confidence* (DC), as the confidence on the target detection, i.e., 100% DC indicates this sensor has no doubt about the existence of the target. Normally, the longer $D_{detection}$ is, the more information about target signature a sensing algorithm can obtain, and therefore, it can achieve a higher detection confidence DC. Such a relationship depends on the type of sensors. In order to quantitatively analyze the relation between DC and $D_{detection}$ as a case study, we performed experiments on XSM motes with the magnetic sensing algorithm detecting a moving vehicle in an outdoor environment. We approximate the sensing range as 7 meters around the sensor node, according to empirical data. Figure 5 plots the relation between the detection confidence and the detection delay, based on the experiments. As we can see from the figure, DC does not have a linear relation to $D_{detection}$. Based on experimental measurements, we use a polynomial to characterize DC versus $D_{detection}$. Figure 5 shows a series of polynomials of different orders that fit the points representing the relation between the detection confidence and the detection delay. The plotting indicates that the polynomials of an order higher than 5 are fairly close to each other and fit the points well. Hence,

we choose the polynomial of order 5 to characterize the relation, as shown below.

$$DC = f(D_{detection}) = \sum_{i=0}^{5} a_i D_{detection}^i \tag{7}$$

The coefficients of the polynomial calculated from the curve fitting are $a_5 = 1.0999, a_4 = -13.1138, a_3 = 51.3443, a_2 = -73.2343, a_1 = 54.6671, a_0 = 0.2402$. The polynomial $f(D_{detection})$ characterizes the relation of the detection confidence and the imposed sub-deadline $D_{detection}$ when the vehicle is moving at a relatively low speed. In the scenarios where the vehicles move faster, the detection delay tends to be shorter and detection confidence will be higher because the targets impose a faster change to the sensor readings. Hence, $f(D_{detection})$ represents a conservative estimation of the detection confidence, given a certain amount of time available to the sensor node to capture and process the target signals.

We note that in the analysis of the time-related properties of the sensing algorithms, we choose such a conservative-case approach instead of a worst-case approach. In many cases, the worst-case scenario is a rare event that the system is not designed to handle well. For example, with the magnetic sensing algorithm, the worst case of detection delay is infinity – if a vehicle moves extremely slowly, it provides a low-frequency signal just as the background noise, resulting in a non-detection for that target. We note that an analysis with such a worst-case scenario provides little insight into the system. To represent a reasonably practical scenario, we study a conservative case in which a target can be detected.

In conclusion, we must provide a detection algorithm that finishes before a given sub-deadline $D_{detection}$. According to Equations 6 and 7, when running a detection algorithm with a sub-deadline $D_{detection}$, one node can detect $P(t > D_{detection})$ percent of targets with DC percent of the confidence in detection. This analysis justifies the benefits of fast detection algorithms and the need for group aggregation to improve the detection confidence.

4.3 Wake-up Delay and Its Tradeoff

Once a target is detected in Phase B, we need more nodes to join in order to increase the confidence in detection. We design a wake-up service to activate the non-sentry nodes after the sentry nodes detect the incoming targets. Different target speeds impose different sub-deadlines D_{wakeup} to the wake-up services.

Normally the wake-up service can be supported either through hardware or software. Several hardware solutions have been proposed in [Dutta et al. 2005; Gu and Stankovic 2004]. Since the wake-up circuits accumulate the ambient energy slowly, the current hardware solutions are not fast enough for the real-time target tracking. Therefore, we propose a software-based wake-up strategy, which has a short average delay and a predictable worse-case delay.

The wake-up operation goes as shown in Figure 6. A non-sentry actually does not sleep all the time. It periodically wakes itself up, quickly senses the radio activity at a particular frequency. If no radio activity is detected, this node goes back to sleep, otherwise it remains active and starts to sample the environment. We control the non-sentry operation through two parameters: *Toggle Period (TP)* and *Channel Clear Access duration (CCA)*. The toggle period is defined as the time interval

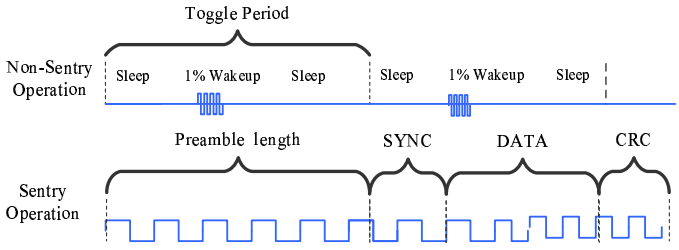

Fig. 6. The Wake-up Operation

between two consecutive wake-up instances. The CCA is defined as the minimal time for a radio module to detect the existence of the radio signal. For example, the CC2420 radio transceiver takes at least $2ms$ (8 symbol periods, as specified by 802.15.4 [IEEE]) to access the radio activity. Based on TP and CCA, we can get the Non-Sentry Duty Cycle ($NSDC$) as $\frac{CCA}{TP}$. At the sentry side, once a sentry detects a target, it broadcasts a radio message with a long preamble. This long preamble is guaranteed to be sensed by neighboring non-sentry nodes as long as this preamble has a length equal to or longer than the toggle period of non-sentry nodes. The worst case wake-up delay WC_{Delay} equals TP. In other words, the sub-deadline D_{wakeup} can be ensured trivially in our design by setting $TP = D_{wakeup}$. Let the power consumption for an active node during a unit of time be E, the energy consumption for a non-sentry node is $\frac{E \times CCA}{TP}$. Since the amount of time to check the radio activity (CCA) is constant for a specific radio hardware, the length of the toggle period determines the energy consumption rate in non-sentry nodes. In general, a long toggle period TP leads to a low energy consumption, however, it also leads to a long delay in waking up the non-sentry nodes. Figure 7 shows such a tradeoff, using the CC1000 radio transceiver for MICA2/XSM motes as an example. As shown in Figure 7, a sub-deadline of 200ms lead to a 99% energy saving for the non-sentry nodes.

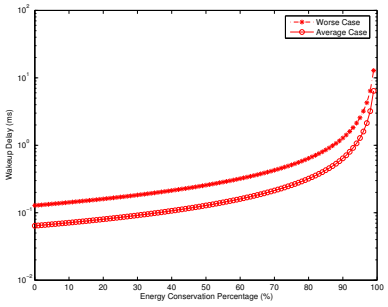

Fig. 7. Wakeup Delay Vs. Non-Sentry Energy Saving

4.4 Aggregation Delay and Its Tradeoffs

Once all nodes near the target are awakened in Phase C, the group-based tracking begins. To avoid an excessive power consumption, instead of relaying every detection message back, VigilNet sends only aggregates to the base stations for further processing. Such an online aggregation process is subject to a certain sub-deadline $D_{aggregation}$ determined by the target speed and the node density.

Specifically, we organize the nodes in the vicinity of a target into one group. We use a semi-dynamic leader election [Luo et al. 2005] to minimize the delay. Nodes that detect the target become the group members, which, upon detection, immediately report their own locations and sensing data to a leader. The leader then averages the locations of members as the estimates of the target positions, and sends such estimates to a base station. To filter out the sporadic false alarms of individual nodes, we introduce a configurable parameter, DOA (Degree of Aggregation), which forces the leader to withhold reports to a base station until the number of received member reports reaches DOA. To achieve a high confidence in target detection, one should set a high DOA value (e.g., 4). On the other hand, a higher DOA value inevitably introduces a longer group aggregation delay since the leader waits longer to expect more member reports. This tradeoff allows us to choose appropriate DOA to meet the sub-deadline $D_{aggregation}$.

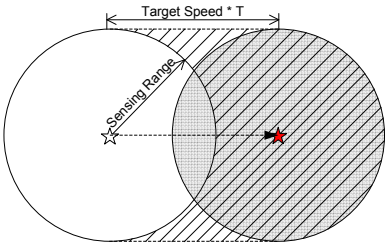

Fig. 8. The Detection Areas Before and After Movement

The relation between DOA and the group aggregation delay is complicated by various factors, e.g., the sensing range, the target speed, and the node density. Therefore, we make several assumptions to simplify the analysis, including a circular sensing range, a straight target trajectory and randomly distributed nodes. Based on these assumptions, Figure 8 depicts the movement of a target with a speed TS for a time period T. Again, the sensing range of a sensor node is SR. The white circle and the grey circle denote the detection area of the target before and after movement, respectively. Nodes located in the diagonally lined area are the new detectors of the target, which contribute to DOA by sending reports to the leader. To guarantee a certain sub-deadline $D_{aggregation}$, the number of new detectors must exceed or equal DOA before the sub-deadline $D_{aggregation}$:

$$D_{aggregation} \geq T_{aggregation} = \frac{DOA}{2 \cdot SR \cdot TS \cdot D} \tag{8}$$

where D represents the node density. Note that after the wake-up process, not only the sentry nodes but also the non-sentry nodes participate in the tracking. Equation 8 quantitatively reveals a feasible region for us to guarantee the sub-deadline $D_{aggregation}$. For example, if the network density (D) and the sensing range (SR) are fixed, we can exploit a feasible solution, using different DOA values under different target speeds. Figure 9 gives a more concrete design space by depicting the group aggregation delay for varied DOA values and target speeds when the sensing range is 10m, the node density is 1 per 100 m^2. We note that this result is consistent with the results obtained from the large-scale simulations presented in Section 7.

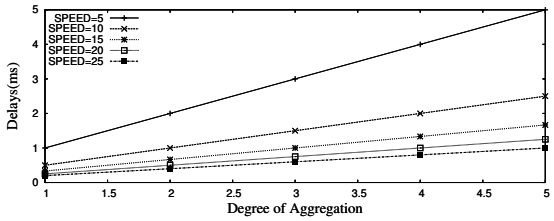

Fig. 9. Minimal Group Aggregation Delay for Varying DOA and Target Speed

4.5 Communication Delay and Its Tradeoff

After group aggregation in Phase D, the leader delivers the aggregated tracking reports to a nearby base. Suppose the end-to-end communication sub-deadline is D_{e2e} and one-hop worst case communication delay is T_{WC_MAC} [He et al. 2003], we need to ensure that the number of hops is smaller than D_{e2e}/T_{WC_MAC}. For a given node density, the hop length L_{hop} can be estimated through Kleinrock-Silvester formula [L.Kleinrock and J.Slivester 1978], which gives the correlation between the hop length L_{hop}, the communication range CR and the number of neighbors N as:

$$L_{hop} = CR \times (1 + e^{-N} - \int_{-1}^{1} e^{-\frac{N}{\pi}(arccos(t) - t\sqrt{1-t^2})} dt) \tag{9}$$

Therefore, to guarantee a sub-deadline D_{e2e}, when we deploy the network, we should ensure that every node can reach a base within a radius of L_{e2e}:

$$L_{e2e} = \frac{D_{e2e} \cdot L_{hop}}{T_{WC\_MAC}} \tag{10}$$

In VigilNet, the sub-deadline D_{e2e} is guaranteed by partitioning the whole network into multiple sections based on the Voronoi diagram [Okabe et al. 2000]. Specifically, a network with n bases is partitioned into n Voronoi sections such that each section contains exactly one base and every node in that Voronoi section is closer to its base than to any other base inside the network.

4.5.1 *Base Deployment Strategy.* We have shown that an ideal deployment should ensure that each node is able to reach a base within a distance of L_{e2e}, so that the sub-deadline D_{e2e} can be satisfied. This possesses an implicit requirement on the number of base stations and their positions. We therefore provide a detailed analysis regarding this requirement and compare the performances of different strategies.

We model the area S with each side as D. Suppose the total number of deployed base stations is N, each serving nodes located within a radius of L. We assume that a large number of other non-base nodes are deployed in the area as well. The problem is, what is probability that every non-base node can reach a base within a distance of L, given a certain deployment strategy? Furthermore, what is the best deployment strategy available? We shall analyze three different deployment strategies: random, grid and optimal. In particular, we show that the optimal strategy is a special case of the grid deployment.

We first consider random deployment. We derive the probability in question as follows. Consider an arbitrary point Q under question. Since each base can serve a radius of L, once a base station is deployed, we know that the point Q has a probability of $\frac{\pi L^2}{S}$ of being located inside this base's service radius. Therefore, once N bases are deployed, the coverage probability for an arbitrary point Q is $1-(1-\frac{\pi L^2}{S})^N$. Notice that this derivation does not take into account the boundary effect. This approximation is valid when $S \gg \pi L^2$, as verified in our experiment.

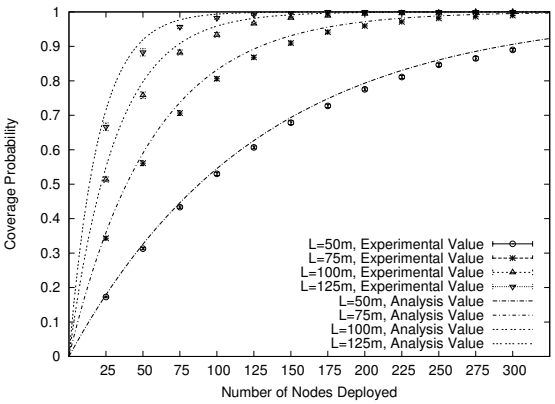

Fig. 10. Random Deployment Performance

Figure 10 validates our analysis on the performance of the random deployment strategy. In particular, we assume that $D = 1000m$. The number of bases N and the serving distance L are both adjustable. All simulation results are plotted based on 50 rounds of data, and the confidence interval is 95%. As shown in this figure, the analysis result roughly fits the experimental data, with certain inconsistencies. These inconsistencies are introduced by the boundary effect: the bases deployed near the boundary have a service area less than πL^2, therefore, the observed coverage probability is slightly lower than the predicted coverage probability.

An interesting problem regarding deployment strategies is *redundancy*. Since typically more bases than needed are provided, it is interesting to consider the

ratio between the number of base stations deployed to the minimum number of base stations required. For example, when $L = 100m$, using the random deployment, we observe that roughly 150 bases are needed to provide each potential node real-time service (the coverage probability is more than 98%). The redundancy can be calculated at $\frac{\pi L^2 \times N}{D^2}$, which is 4.71. This is indeed quite high. We, therefore, discuss more efficient deployment strategies, assuming we can position the base stations at desired places accurately.

We focus on two types of grid strategies, square based and hexagon based. These strategies are shown in Figure 11.

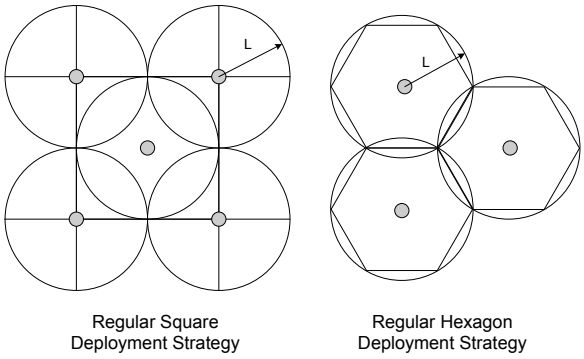

Fig. 11. Regular Deployment Strategies

In the first type of grid deployment, the base stations form a regular square structure. The redundancy can be determined to be about 1.57. The second type of grid deployment forms the *honeycomb* structure, where regular hexagons are used. Notice that this figure also shows the Voronoi diagram partitions associated with the honeycomb structure. The second grid deployment has a redundancy of $\frac{2\pi}{3\sqrt{3}}$, or approximately 1.208.

Previous literature [Williams 1979] has proved that the optimal redundancy ratio for any circle covering is exactly $\frac{2\pi}{3\sqrt{3}}$. This result indicates that the honeycomb based node deployment is the optimal strategy. Indeed, boundary effect also exists in this type of deployment, however, when the area is considerably large, the actual redundancy ratio should approach the optimal bound.

4.6 Base Processing Delay and Its Tradeoffs

After a base receives the reports delivered in phase E, it performs the high-level processing such as the velocity estimation. In order to do so, a base node needs to accumulate several reports from the network. The delay to accumulate the reports T_{base} is subject to its sub-deadline D_{base}. We defined the minimal number of reports needed by the base as K. This value can be one, if the in-networking processing is sufficient. The frequency of reports depends on the speed of the target and the aggregation of locations from nodes at different locations. From the analysis in Section 4, we know that after the target enters the system for time t, the expected number of nodes that can sense the target is $(\pi \cdot SR^2/2 + 2SR \cdot TS \cdot t)D$.

Obviously, if the target goes further for Δt, the expected number is increased by $2SR \cdot TS \cdot \Delta t$. Considering the detection delay $T_{detection}$, only nodes that are $\sqrt{SR^2 - (T_{detection} \cdot TS/2)^2}$ meters away from the target trajectory can recognize the target. Therefore, we can estimate the number of reports (NR) generated before the sub-deadline D_{base} as:

$$NR = (2TS \cdot D \cdot \sqrt{SR^2 - (T_{detection} \cdot TS/2)^2}) \cdot D_{base} \tag{11}$$

Alteratively, to guarantee D_{base}, we need to select K, the minimal number of reports needed by the base, to be a value smaller than NR.

Now we consider how the selection of K impacts the accuracy in velocity estimation. Since each location report is an approximation of the target location, there is an error in the result of velocity estimated using the least square method. Without loss of generality, we first consider the velocity along the x-axis. Statistics has established the variance of the estimated slope in a two-variable least square linear regression as:

$$\frac{\sigma^2}{\sum_{i=1}^{K}(x_i - \bar{x}^2)},$$

where σ is the standard deviation of the disturbance, which in our case is the detection error of a single report; x_i in our case is a timestamp. It is hard to get the distribution of $\sum_{i=1}^{K}(x_i - \bar{x})^2$, but a rough estimation can be obtained by a simplification so that the values of x_i are evenly distributed and $x_i = i/(2D \cdot SR \cdot TS \cdot P_R)$. Thus we can get an estimation of the standard deviation of the velocity:

$$\frac{4\sigma \cdot D \cdot SR \cdot TS \cdot P_R}{\sqrt{3K(K+1)(K-1)}}, \tag{12}$$

where σ is the standard deviation of the location error of a single report. Equation 12 reveals the tradeoff between the accuracy in tracking and the delay in base processing. In brief, T_{base} increases linearly with the number of reports required and the standard deviation of the velocity estimation reduces approximately linearly with $K^{-3/2}$.

4.7 Summary of the Analysis and Tradeoffs

Dealing with the physical world, many sensor-based systems must respond to external stimuli within certain time constraints. Such constraints could change over time with the changes of the application objectives. For example, a surveillance system should be able to track fast vehicles at a high energy budget as well as slow personnel at a smaller budget. So it is desirable for a system designer to have the ability to trade off the system parameters to satisfy certain real-time constraints. In this section, we use the deadline partition method to guarantee the sub-deadline of each phase, consequently guaranteeing the end-to-end deadline. This approach makes the real-time design for a complex sensor network manageable. Since VigilNet aims at various tracking scenarios, for a given end-to-end deadline, the actually partition among the phases would vary significantly. Our analysis is independent of how the sub-deadlines are assigned, which gives the designer more flexibility to choose

appropriate partition. Currently, the deadline partition is done statically, and we shall investigate the solutions that allow dynamic online partition in the future.

We note our analysis provides a set of generic design guidelines for other tracking systems with or without certain features. For example, the tracking system presented in [Arora et al. 2004] does not consider the power management, which makes the analytical results of $T_{initial}$ and T_{wakeup} trivially zero, while other analytical results are still applicable. Other notable insights from our analysis are: First, to guarantee the same sub-deadline, a higher node density is desired in the slow-target case, however a slower duty cycle can be tolerated without jeopardizing the detection. Second, it is very beneficial to increase the wake-up delay, when possible, in exchange for the energy saving. Third, fast detection algorithms are essential. Fourth, a low network density increases the group aggregation delay, which indirectly reduces the detection confidence. Fifth, theoretically, honeycomb is the optimal base placement strategy.

We also note due to the unpredictable and statistical nature of environmental inputs (e.g., a target could move infinitely slowly, and sensing and communication ranges could be highly irregular), VigilNet is not quite amenable to the traditional precise worst-case real-time analysis. Nevertheless, the analytical results we provide can assist the designer to provide soft real-time guarantee and make guided decisions on system configurations. In the Section 6 and Section 7, we validate our real-time design and analysis through a physical test-bed with 200 XSM motes as well as a large-scale simulator with 10,000 nodes, respectively.

5. SYSTEM IMPLEMENTATION

A large portion of code of VigilNet is written in NesC [Gay et al. 2000], a module oriented extension of the C programming language. Since the concept of traditional OS kernels does not exist in TinyOS [Hill et al. 2000], a NesC programmer can directly access the hardware devices, which facilitates the time analysis within a single node [Mohan et al. 2004]. The network infrastructure in VigilNet is a multipath diffusion tree rooted at bases. The contention-based B-MAC protocol [Polastre and Culler 2004] is the default media access control protocol, which has certain uncertainty in the communication delay. Three detection algorithms are designed separately for acoustic, magnetic and motion sensors. They identify the target signatures through a lightweight classification scheme as described in [Gu et al. 2005]. VigilNet consists 40,000 lines of code, supporting multiple existing mote platforms including MICA2, MICA2dot and XSM. The compiled image occupies 83,963 bytes of code memory and 3,586 bytes of data memory.

Among 30 protocols implemented within VigilNet, we only describe the time-related services here. Other information can be found at [He et al. 2006; He et al. 2006; He et al. 2004]. VigilNet needs a millisecond-level synchronization to coordinate the operations among the nodes. In addition, to obtain precise timing measurements in the experiments, we need a network-wide synchronization between a base and other nodes within the field. Several well-known schemes are able to achieve a high synchronization precision, however they do not match well with VigilNet requirements. GPS-based schemes [Wellenhoff et al. 1997] 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 [Elson and Romer 2002] maintains information relating the phase and frequency of each pair of clocks in the neighborhood of a node. 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 believe that fine-grained clock synchronization achieved by costly periodic beacon exchanges may not be suitable for the energy-constrained surveillance system. Therefore, we modified the FTSP time synchronization protocol [Maroti et al. 2004] to synchronize the motes only during the 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. The synchronization beacons are propagated across the network through limited flooding with timestamp values reassigned at intermediate motes immediately prior to the transmission of the timestamp. This eliminates the uncertainty in MAC contention delay. Receivers take the timestamp from the beacon plus a fixed hardware delay as their own local time. The timer drift that accumulates over time is rectified by a new system cycle (i.e., a repeated initialization phase). The frequency of re-initialization is a configurable parameter, which can be calculated based on the rate of clock drift and the desired accuracy of time synchronization. As for the current VigilNet system, the accuracy of tens of milliseconds is sufficient, which leads to about once per day synchronization.

6. EVALUATION OF REAL SYSTEM PERFORMANCE

In the evaluation, we validate the analytical results as well as provide more insights into the timing issues from the real system and simulation perspectives.

6.1 Experimental Settings

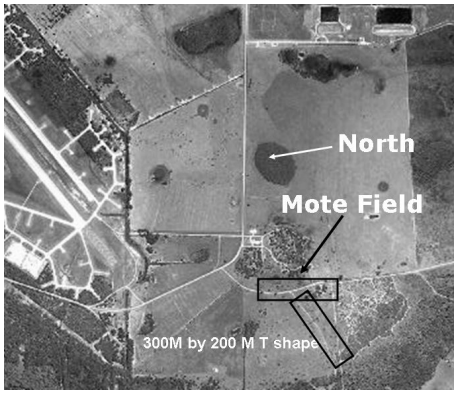

Fig. 12. Deployment Site

As a real-time online tracking system, VigilNet is designed to complete detection, classification and velocity estimation within 4 seconds. The field test was done

on a T-shape dirt road in Florida as shown in Figure 12 from the aerial view. We deployed 200 XSM motes which are equipped with CC1000 radio, magnetic, acoustic, photo, temperature and passive infrared sensors (PIR). Along the road, nodes were randomly placed roughly 10 meters apart, covering one 300-meter road and one 200-meter road. Through a certain localization [Stoleru et al. 2004; He et al. 2003; Stoleru et al. 2005], nodes were aware of their positions. In order to measure various kinds of delay, all nodes within VigilNet synchronized with the base within 1∼10 milliseconds using the techniques described in [Maroti et al. 2004]. The time stamps of various actions such as initial detection were sent back to the base, so that we can calculate the delay. We used a Ford Explorer that weighted about 4000 lbs. as the target.

6.2 Delay Measurements

When a car enters the surveillance area at about 10 meters per second (22 mph), a detection report is issued first, followed by classification reports. Finally, after sufficient information is gathered, velocity reports are issued. Figure 13 illustrates the cumulative distribution of different delays. The communication delay (leftmost curve) is much smaller compared with other delays. About 80% of detections are done within 2 seconds. Over 80% of the classification and velocity estimations are made within 4 seconds. The empirical results from most runs are consistent with our analysis in Section 4 and the simulation results in Section 7.

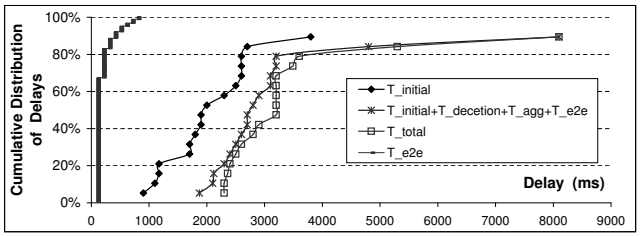

Fig. 13. Various Delays Measurements from Field Test

We emphasize here that field experiments indicate that VigilNet meets its real-time requirement and our real-time analysis can approach the reality with a reasonable precision, despite the amount of complexity within the VigilNet (30 protocols integrated). On the other hand, we acknowledge that due to various physical constraints, field experiments can only exploit a very limited design space and obtain a limited amount of data. Therefore, to understand the real-time properties in VigilNet at scale with a much larger context, we provide a large-scale simulation in the next section.

7. LARGE-SCALE SIMULATION

Our simulator is a discrete simulator, written in C++. It emulates the tracking operations as shown in Figure 1. We distribute 10,000 nodes randomly within a 100,000 m^2 rectangle area, assuming nominal circular sensing and communication

ranges. We run each experiment 30 times with different random numbers. The figures are plotted with the average value as well as the 95% confidence interval.

7.1 Experiment Setup

We note that our evaluation does not choose deadline/sub-deadline miss ratios as the major metrics, because such an approach reveals less information about the tradeoff between actual delays and other system performance parameters. Since the mean value and 95% confidence intervals of the delays are plotted in the figures, one can determine the appropriate system settings for a given deadline requirement.

In our experiments, we study several system-wide parameters that directly affect the real-time properties of VigilNet. These parameters are: 1) the target speed (TS), 2) the physical delay in detection ($T_{detection}$), 3) the sentry duty cycle (SDC), 4) the non-sentry duty cycle (NSDC), 5) the required degree of aggregation (DOA), 6) the sensing range (SR) and 7) the required number of reports for base processing (K). We match the simulations with the analysis to see how well they fit with each other.

We use the settings from the VigilNet system as the default values for these system parameters, which are listed in Table I. Unless mentioned otherwise, the default values in Table I are used in all experiments. The metrics used to measure the system performance are mainly the six types of delays discussed in Section 2, the end-to-end delay and the energy consumption per day per node.

Table I. Key System Parameters

Parameter	Definition	Default Value
TS	Target Speed	10 m/s
SDC	Sentry Duty Cycle	50%
NSDC	Non-Sentry Duty Cycle	1%
DOA	Degree of Aggregation	1%
SR	Sensing Range	10
K	Reports required by the base	1
D	Node Density	0.01 m^2

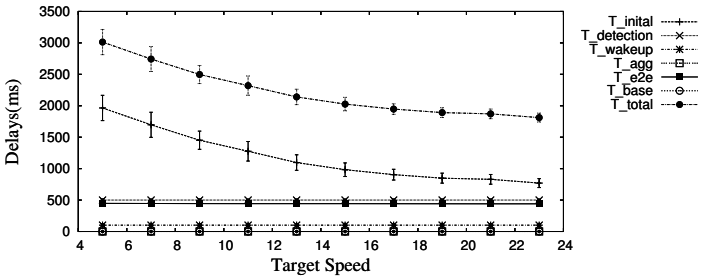

Fig. 14. Delays vs. Target Speed

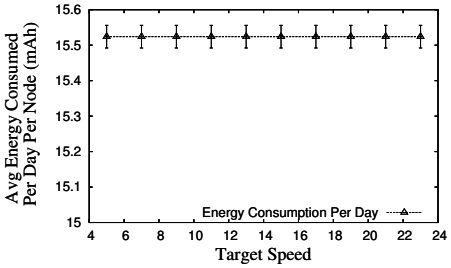

Fig. 15. Energy Consumption vs. Target Speed

7.2 Performance vs. Target Speed

The target speed determines the spatiotemporal distribution of events over a certain time period. It is crucial to understand its impacts on the tracking performance. In this experiment, we incrementally increase the target speed (TS) from 5m/s to 15m/s in steps of 1 meter. As expected from our analysis in Section 4, $T_{initial}$ and $T_{aggregation}$ decrease with the target speed, as shown in Figure 14. One interesting observation is that the descend rate of $T_{initial}$ diminishes when TS becomes larger. This is because a node needs a sufficient sensing time to ensure detection. It is possible that a quick target passes one sensor without detection, which negatively affects the $T_{initial}$. Since VigilNet deals with a rare event model, the energy consumed during the tracking is not perceptibly affected by the target speeds as shown in Figure 15.

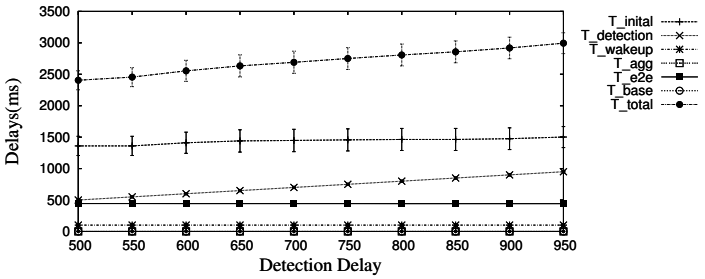

Fig. 16. Delays under Varying Detection Delay

7.3 Performance vs. Detection Delay

Different tracking systems use different sensing devices and detection algorithms, which have various detection delays $T_{detection}$. In this experiment, we increase the delay in the detection algorithm $T_{detection}$ from 500 ms to 1000 ms in steps of 50 ms. It is interesting to observe in Figure 16 that at a speed of 10m/s, the detection delay has a small impact on the initial delay, however it contributes most significantly to the overall increase of the total tracking delay. Again, since the detection time is relatively small, this system parameter does not noticeably affect the overall energy consumption, as shown in Figure 17.

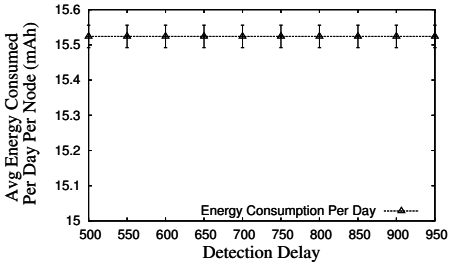

Fig. 17. Energy Consumption vs. Detection Delay

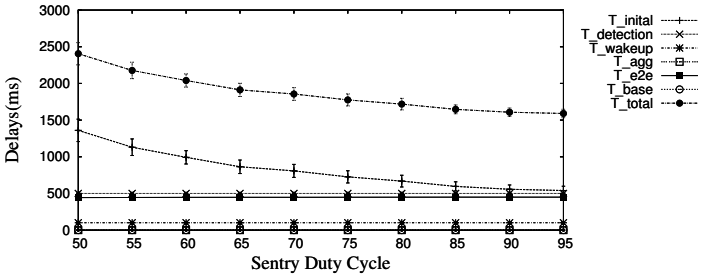

Fig. 18. Delays vs. Sentry Duty Cycle

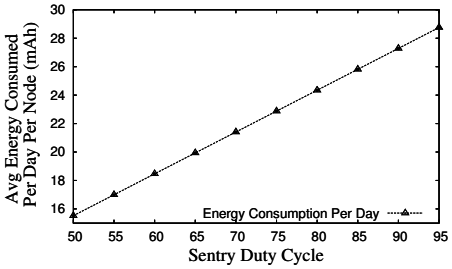

Fig. 19. Energy Consumption vs. Sentry Duty Cycle

7.4 Performance vs. Sentry Duty Cycle

From the analytical results in Section 4, we obtain an analytical delay curve between $T_{initial}$ and SDC in Figure 4. In this experiment, we obtain a set of other curves (Figure 18) through the simulation. By comparing these two results, we conclude that they are consistent with each other. For example, at a default 50% duty cycle, $T_{initial}$ obtained from the analysis in Figure 4 is 1600ms, while $T_{initial}$ obtained from the simulation (Figure 18) is 1360ms (Note that our analysis is relatively conservative). In addition, Figure 19 reveals that the energy consumption escalates linearly with the SDC, which indicates that an efficient sentry scheduling algorithm is beneficial.

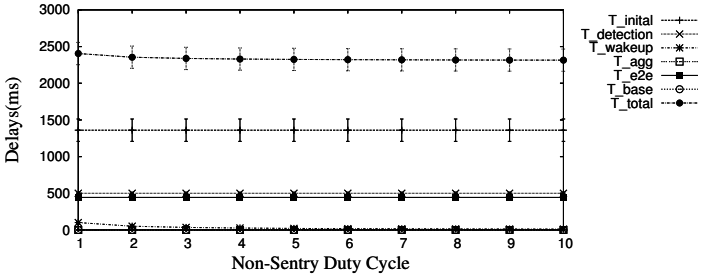

Fig. 20. Delays vs. Non-Sentry Duty Cycle

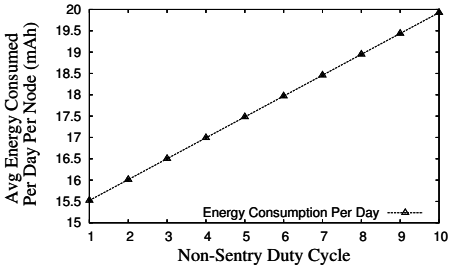

Fig. 21. Energy Consumption vs. Non-Sentry Duty Cycle

7.5 Performance vs. Non-sentry Duty cycle

Here, we evaluate the impact of the wake-up operation on the delay and energy consumption. First, the simulation results confirm that the average wake-up delay is approximately half of the toggle period as predicted in Section 4.3. Since the wake-up delay T_{wakeup} is one order of magnitude smaller than other delays such as $T_{initial}$, a slight decrease in the wake-up delay, shown in Figure 20, does not noticeably impact the overall delay. However, interestingly a slight increase of the Non-Sentry Duty Cycle leads to a significant increase in energy consumption as shown in Figure 21. This is because the non-sentry nodes are by far the majority, so a duty-cycle increase for the non-sentry nodes leads to a quick increase in the total energy. This result indicates that it is beneficial to increase the wake-up delay, when possible, in exchange for the energy saving.

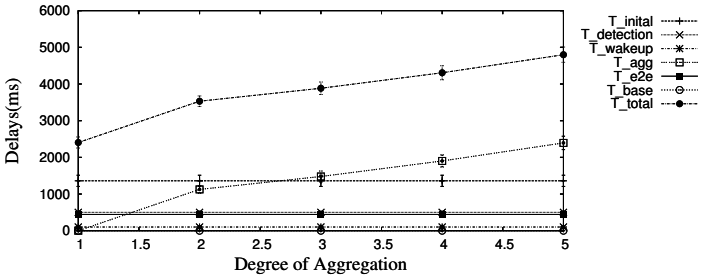

Fig. 22. Delays vs. DOA

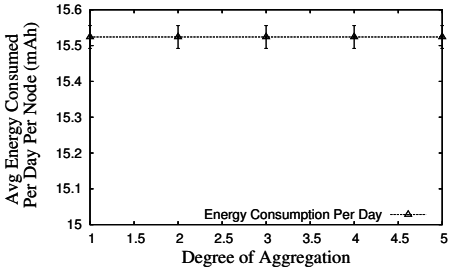

Fig. 23. Energy Consumption vs. DOA

7.6 Performance vs. DOA

In-network processing through data aggregation can reduce the amount of data transmitted over the network and can increase the confidence in target detection. However, to accumulate enough report, it inevitably introduces a certain delay. This experiment studies the effects of data aggregation. We gradually increase the DOA threshold for a leader to report to the base. Since the DOA value only affects the tracking phase, which has a small energy consumption, DOA's impact on the energy consumption is not noticeable. On the other hand, with a larger DOA value, it takes more time for a leader to collect the member reports. For example as shown in Figure 22, it takes as long as 2.39 seconds to achieve DOA value of 5. We note that this simulation result is again consistent with the analytical results shown in Figure 9, which has an estimated delay of 2.5 seconds.

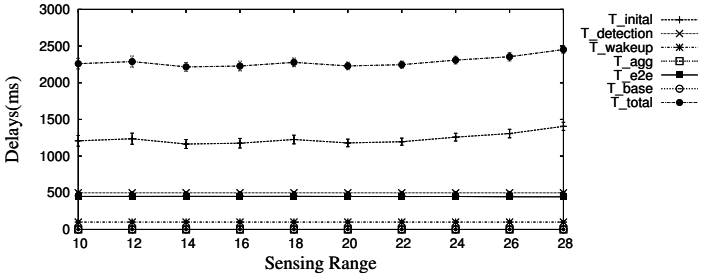

Fig. 24. Delays vs. Sensing Range

7.7 Performance vs. Sensing Range

To accommodate various requirements in detection and classification, different tracking systems use sensors with different ranges. Figure 24 and Figure 25 investigate the impact of sensing range to the tracking performance and energy consumption. With a large sensing range, a smaller number of sentries is required. Therefore, the total energy consumption decreases quickly. For example in Figure 25, the energy reduces by 75% when the sensing range increases from 10m to 28m. It is interesting to see that the initial delay $T_{initial}$ actually slightly increases. This is because the number of sentry nodes reduces while the coverage per sensor

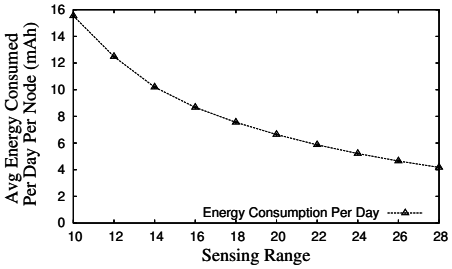

Fig. 25. Energy Consumption vs. Sensing Range

increases, the total coverage by all sentry nodes remains the same. We can derive from Equation 3 that the expected $T_{initial}$ is higher when the sensing range is smaller, given the same coverage in both cases. This analytic result is confirmed by the simulation results shown in Figure 24. Due to the space constraints, we omit the detailed derivation here.

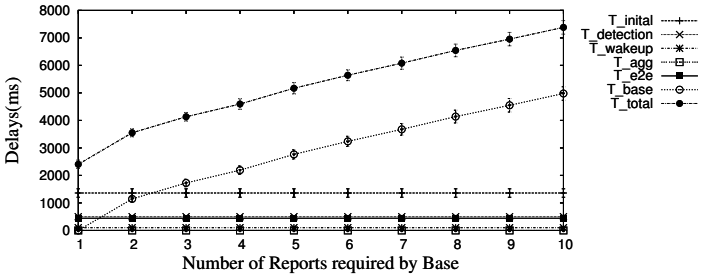

Fig. 26. Delays vs. Num of Required Reports

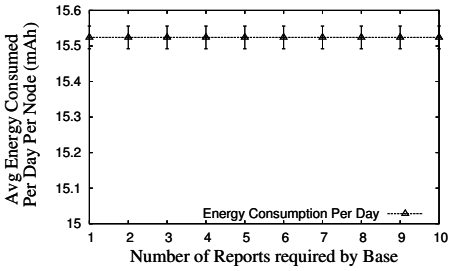

Fig. 27. Energy Consumption vs. Num of Required Report

7.8 Performance vs. Number of Reports

To improve the estimation of target velocity and to classify targets with a high confidence, a base node normally needs to accumulate a certain number of spatiotemporally related reports from the same logic tracking group. This experiment investigates the impact of the number of reports required by a base on the tracking

delays. Obviously, this only affects T_{base}. Figure 26 shows that T_{base} approximately increases linearly with the number of reports, which is expected from our analytical results in Section 4.6. Since the operation is done at the base, there is no energy impact to the sensor network, as shown in Figure 27.

8. RELATED WORK

Real-time protocols have been designed at different layers to guarantee the effectiveness of the interactions between wireless sensor networks and the physical world. At the MAC layer, RAP [Lu et al. 2002] uses a novel velocity monotonic scheduling to prioritize the real-time traffic and enforce such prioritization through a differentiated MAC Layer. Woo and Culler [Woo and Culler 2001] propose an adaptive rate control scheme to achieve fairness among the nodes with different distances to a base station. Li [Li et al. 2005] proposes a SLF message scheduling algorithm to exploit spatial channel reuse, so that deadline misses can be reduced. Carley [Carley et al. 2003] designs a periodic message scheduler to provide a contention-free predicable medium access control. At the network layer, He [He et al. 2003]et al. support a soft real-time communication service with a desired delivery speed across the sensor network, using feedback-based adaptation algorithms that enforce per-hop speed in face of unpredictable traffic. Felemban [Felemban et al. 2005] presents a novel packet delivery mechanism, called multi-path and multi-speed routing protocol, for probabilistic QoS guarantee in wireless sensor networks. At the aggregation layer, Vasudevan [Vasudevan et al. 2003] proposes an application-specific compression for time delay estimation in sensor networks, and He [He et al. 2004] adaptively performs application independent data aggregation in a time sensitive manner. The real-time solutions at the application is highly diversified. Huang [Huang et al. 2003] et al. propose the Mobicast protocol to provide just-in-time information dissemination to nodes in a mobile delivery zone. Given the complete knowledge of traffic pattern, Somasundara [Somasundara et al. 2004] proposes a mobile agent scheduling algorithm to collect the buffered sensor data, before the buffer overflow occurs at the sensor nodes. Nam [Nam et al. 2005] proposes time-parameterized sensing task model for real-time tracking. Yang [Yang and Vaidya 2004] proposes a wakeup scheme that assists balancing energy saving and end-to-end delay. The Lightning protocol [Wang et al. 2004] localizes the acoustic source with a bounded delay regardless of the node density.

Besides the real-time protocol design, several researchers have focused on the time analysis for sensor networks. Lu [Lu et al. 2005] studies how to minimize the communication latency given that each sensor has a duty cycling requirement of being awake for only $\frac{1}{k}$ time slots on average. In [Mohan et al. 2004], Mohan et al. provides a cycle-accurate WCET analysis tool for the applications running on the Atmega Processor Family. Abdelzaher [Abdelzaher et al. 2004] derives a real-time capacity bound for multi-hop wireless sensor networks. It is a sufficient schedulability condition for a class of fixed priority packet scheduling algorithms. Using this bound, one can determine whether a certain traffic pattern can meet its real-time requirement beforehand.

With advances in the sensor techniques, several large-scale sensor systems have been built recently. The GDI Project [Szewczyk et al. 2004] provides an environ-

mental monitoring system to record animal behaviors for a long period of time. The shooter localization system [Simon et al. 2004] collects the time-stamps of the acoustic detection from different nodes within the network to localize the positions of the snipers. These systems mention some timing issues, however they do not treat real-time as a major concern. Our previous publications on VigilNet [He et al. 2004; He et al. 2006] focus on the middleware services and overarching system integration. To the best of our knowledge, this work is the first to analyze the real-time performance and its tradeoffs in a real-world large-scale wireless sensor system.

9. CONCLUSION

In this paper, we demonstrate the feasibility to design a complex real-time sensor network, using the deadline partition method, which guarantees an end-to-end tracking deadline by satisfying a set of sub-deadlines. We also analytically identify the tradeoffs among system properties while meeting the real-time requirements. We validate our design and analysis through both a large-scale simulation with 10,000 nodes as well as a field test with 200 XSM nodes. We contribute a set of tradeoffs that are useful for the future development of real-time sensor systems. Given real-time constraints, a system designer can make guided engineering judgments on the system parameters. Here we just name a few. First, to guarantee the same sub-deadline, a higher node density is desired in the slow-target case, however a slower duty cycle can be tolerated without jeopardizing the detection. Second, it is beneficial to increase the wake-up delay, when possible, in exchange for the energy saving. Third, fast detection algorithms are essential. Fourth, a low network density increases the group aggregation delay, which indirectly reduces the detection confidence. Fifth, theoretically, honeycomb is the optimal base placement strategy to meet the communication sub-deadline.

Finally, we acknowledge that although it is amenable to provide the worst-case real-time analysis for a certain protocol such as the wake-up protocol in Section 4.3, however, due to the dynamic and unpredictable nature of the sensor networks, it is a long-term research goal for us to achieve precise worst-case real-time analysis across the whole system.

10. ACKNOWLEDGEMENTS

This work was supported in part by the DAPRPA IXO offices under the NEST project (grant number F336615-01-C-1905), the MURI award N00014-01-1-0576 from ONR, NSF grant CCR-0329609, CCR-0325197 and CNS-0435060. The authors specially thank the NEST program manager Dr. Vijay Raghavan for his valuable contributions.

REFERENCES

Abdelzaher, T. F., Prabh, S., and Kiran, R. 2004. On Real-Time Capacity Limits of Multihop Wireless Sensor Networks. In *IEEE RTSS*.

Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., Mittal, V., Cao, H., Demirbas, M., Gouda, M., Choi, Y., Herman, T., Kulkarni, S., Arumugam, U., Nesterenko, M., Vora, A., and Miyashita, M. 2004. A Wireless Sensor Network for Target Detection, Classification, and Tracking. *Computer Networks (Elsevier)*.

Arora, A. and et al. 2005. Exscal: Elements of an extrem scale wireless sensor network. In *11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2005)*.

Cao, Q., Yan, T., Abdelzaher, T., and Stankovic, J. 2005. Analysis of target detection performance for wireless sensor networks. In *DCOSS'05*.

Carley, T. W., Ba, M. A., Barua, R., and Stewart, D. B. 2003. Contention-free periodic message scheduler medium access control in wireless sensor/actuator networks. In *IEEE RTSS*.

CrossBow 2003. *Mica2 data sheet*. CrossBow. Available at `http://www.xbow.com`.

Dutta, P., Grimmer, M., Arora, A., Biby, S., and Culler, D. 2005. Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events. In *IPSN'05*.

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

Felemban, E., Lee, C., Ekici, E., Boder, R., and Vural, S. 2005. Probabilistic qos guarantee in reliablility and timeliness domains in wireless senosr networks. In *IEEE INFOCOM 2005*.

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

Gu, L., Jia, D., Vicaire, P., Yan, T., Luo, L., A.Tirumala, Cao, Q., T. He, J. A. S., T.Abdelzaher, and Krogh., B. 2005. Lightweight Detection and Classification for Wireless Sensor Networks in Realistic Environments. In *SenSys'05*.

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

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

He, T., Huang, C., Blum, B. M., Stankovic, J. A., and Abdelzaher, T. 2003. Range-Free Localization Schemes in Large-Scale Sensor Networks. In *MOBICOM'03*.

He, T., Krishnamurthy, S., Luo, L., Yan, T., Stoleru, R., Zhou, G., Cao, Q., Vicaire, P., Stankovic, J. A., Abdelzaher, T. F., Hui, J., and Krogh, B. 2006. VigilNet: An Integrated Sensor Network System for Energy-Efficient Surveillance. *ACM Transaction on Sensor Networks*.

He, T., Krishnamurthy, S., Stankovic, J. A., Abdelzaher, T., Luo, L., Stoleru, R., Yan, T., Gu, L., Hui, J., and Krogh, B. 2004. An Energy-Efficient Surveillance System Using Wireless Sensor Networks. In *MobiSys'04*.

He, T., Stankovic, J., Lu, C., and Abdelzaher, T. 2003. SPEED: A Stateless Protocol for Real-Time Communication in Ad Hoc Sensor Networks. In *ICDCS'03*.

He, T., Vicaire, P., Yan, T., Cao, Q., Zhou, G., Gu, L., Luo, L., Stoleru, R., Stankovic, J. A., , and Abdelzaher, T. 2006. Achieving Long-Term Surveillance in VigilNet. In *IEEE Infocom*.

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

Huang, Q., Lu, C., and Roman, G.-C. 2003. Spatiotemporal Multicast in Sensor Networks. In *SenSys 2003*.

IEEE. IEEE Wireless Medium Access Control(MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs).

Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L., and Rubenstein, D. 2002. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *Proc. of ASPLOS-X*.

Li, H., Shenoy, P., and Ramamritham, K. 2005. Scheduling Messages with Deadlines in Multi-hop Real-time Sensor Networks. In *RTAS'05*.

L.Kleinrock and J.Slivester. 1978. Optimum transmission radii for packet radio networks or why six is a magic number. In *national Telecomm conference*.

Lu, C., Blum, B. M., Abdelzaher, T. F., Stankovic, J. A., and He, T. 2002. Rap: A real-time communication architecture for large-scale wireless sensor networks. In *IEEE RTAS*.

Lu, G., Sadagopan, N., Krishnamachari, B., and Goel, A. 2005. Delay efficient sleep scheduling in wireless sensor networks. In *IEEE INFOCOM 2005*.

Luo, L., He, T., Abdelzaher, T., and Stankovic, J. 2005. Design and comparison of lightweight group management strategies in envirosuite. In *DCOSS '05: International Conference on Distributed Computing in Sensor Networks*.

Maroti, M., Kusy, B., Simon, G., and Ledeczi, A. 2004. The Flooding Time Synchronization Protocol. In *SenSys'04*. 39–49.

Mohan, S., Mueller, F., Whalley, D., and Healy, C. 2004. Timing Analysis for Sensor Network Nodes of the Atmega Processor Family. In *IEEE RTSS*.

Nam, M.-Y., Lee, C.-G., Kim, K., and Caccamo, M. 2005. Time-parameterized sensing task model for real-time tracking. In *IEEE RTSS 2005*.

Okabe, A., Boots, B., Sugihara, K., and Chiu, S. N. 2000. *Spatial Tessellations:Concepts and Applications of Voronoi Diagrams*. Wiley.

Polastre, J. and Culler, D. 2004. Versatile Low Power Media Access for Wireless Sensor Networks. In *Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*.

Simon, G., Maroti, M., Ledeczi, A., Balogh, G., Kusy, B., Nadas, A., Pap, G., Sallai, J., and Frampton, K. 2004. Sensor Network-Based Countersniper System. In *SenSys'04*.

Somasundara, A. A., Ramamoorthy, A., and Srivastava, M. B. 2004. Mobile Element Scheduling for Efficient Data Collection in Wireless Sensor Networks with Dynamic Deadlines. In *IEEE RTSS*.

Stoleru, R., He, T., and Stankovic, J. A. 2004. Walking GPS: A Practical Solution for Localization in Manually Deployed Wireless Sensor Networks. In *EmNetS-I*.

Stoleru, R., He, T., Stankovic, J. A., and Luebke, D. 2005. High-Accuarcy, Low-Cost Localization System for Wireless Sensor Networks. In *Third ACM Conference on Embedded Networked Sensor Systems (SenSys 2005)*.

Szewczyk, R., Mainwaring, A., Anderson, J., and Culler, D. 2004. An Analysis of a Large Scale Habit Monitoring Application. In *SenSys'04*.

Vasudevan, L., Ortega, A., and Mitra, U. 2003. Application-Specific Compression for Time Delay Estimation in Sensor Netowrks. In *ACM SenSys 2003*.

Wang, Q., Zheng, R., Tirumal, A., Liu, X., and Sha, L. 2004. Lightning: A Fast and Lightweight Acoustic Localization Protocol Using Low-End Wireless Micro-Sensors. In *IEEE RTSS*.

Wellenhoff, B. H., Lichtenegger, H., and Collins, J. 1997. *Global Positions System: Theory and Practice,Fourth Edition*. Springer Verlag.

Williams, R. 1979. Circle Coverings. In *The Geometrical Foundation of Natural Structure: A Source Book of Design, pp. 51-52*.

Woo, A. and Culler, D. 2001. A Transmission Control Scheme for Media Access in Sensor Networks. In *Proc. of Mobile Computing and Networking (Mobicom)*.

Xu, N., Rangwala, S., Chintalapudi, K. K., Ganesan, D., Broad, A., Govindan, R., and Estrin, D. 2004. A Wireless Sensor Network for Structural Monitoring. In *SenSys 2004*.

Yang, X. and Vaidya, N. H. 2004. Awakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay. In *IEEE RTAS 2004*.