Energy Efficient Sleep Scheduling in Sensor Networks for Multiple Target Tracking

Bo Jiang*, Binoy Ravindran*, and Hyeonjoong Cho‡

*ECE Dept., Virginia Tech
Blacksburg, VA 24061, USA
{bjiang,binoy}@vt.edu

‡ETRI
Daejoen, South Korea
raycho@etri.re.kr

Abstract

This paper presents an energy-aware, sleep scheduling algorithm called SSMTT to support multiple target tracking sensor networks. SSMTT leverages the awakening result of interfering targets to save the energy consumption on proactive wake-up communication. For the alarm message-miss problem introduced by multiple target tracking, we present a solution that involves scheduling the sensor nodes' sleep pattern. We compare SSMTT against three sleep scheduling algorithms for single target tracking: the legacy circle scheme, MCTA, and TDSS. Our experimental evaluations show that SSMTT achieves better energy efficiency than handling multiple targets separately through single target tracking algorithms.

1. Introduction

Target tracking in surveillance systems is one of the most important applications of wireless sensor networks (WSNs) [3]. Some of the earlier applications in this domain have focused on rare-event tracking where targets typically enter the surveillance field one by one [6, 14]. Recently, many interesting applications have emerged, which require concurrent tracking of multiple targets—e.g., search and rescue, disaster response, pursuit evasion games [23]. As one of the critical mechanism for WSNs' energy efficiency, sleep scheduling is still a challenging problem for multiple target tracking systems. This is because, it is generally difficult to optimize energy efficiency and simultaneously track all targets without missing any, when multiple targets concurrently intrude the surveillance field.

In this paper, we present an energy-aware, Sleep Scheduling algorithm for Multiple Target Tracking (or SSMTT). Our objective is to improve the energy efficiency through a sleep scheduling approach that is conscious of concurrently tracking multiple targets, in contrast to an approach which is not.

In the proactive wake-up mechanism for target tracking [13], a node that detects a target (i.e., the "root" node) broadcasts an alarm message to activate its neighbor nodes (i.e., member nodes) toward preparing them to track the approaching target. When the routes of multiple targets interfere with each other, some neighbor nodes of a root node may have already been activated by another root node's alarm broadcast. If the wake-up mechanism is carefully designed, such overlapping broadcasts can be saved, thereby saving associated energy costs on communication. SSMTT uses such a mechanism, and builds upon the tracking subarea management and sleep scheduling algorithms in [18]. A consequence of this multi-target-conscious energy efficiency mechanism is that some sensor nodes may be put into the sleep state by the alarm broadcast for a target, and thereby, they may miss the alarm broadcast for another. We present a solution to this problem by modifying the node sleep patterns.

Most of the past works on multiple target tracking aim at differentiating multiple targets from each other i.e., identifying "who is who" [19, 24], or improving data fusion [7, 12]. Some of those works which also consider improving energy efficiency do not consider sleep scheduling [25]. On the other hand, most efforts on sleep scheduling do not explicitly support concurrent tracking of multiple targets [8, 10]. Liu et al. utilize multiple targets as tracking objects in their simulation studies [21]. However, this work aims at guaranteeing the quality of traffic towards the base station instead of tracking along the target's route. To the best of our knowledge, this work is the first effort to enhance energy efficiency through sleep scheduling for multiple target tracking systems.

The paper makes the following contributions: (1) We present a sleep scheduling algorithm for concurrently tracking multiple targets; (2) we further enhance energy efficiency by leveraging the awakened sensor nodes to save more energy than tracking multiple targets separately with single target tracking algorithms; and (3) we provide a simple solution for the alarm message-miss problem.

The results from our experimental evaluations show that, compared with sleep scheduling algorithms for single target tracking, the SSMTT algorithm saves alarm transmission energy by $10\% \sim 15\%$.

The rest of the paper is organized as follows: In Section 2, we describe our assumptions and formulate the problem. Section 3 describes the rationale and design of SSMTT. In Section 4, we present detailed algorithm descriptions. Section 5 reports our evaluation results. In Section 6, we conclude and discuss future work.

2. Assumptions and Problem Formulation

2.1. Assumptions

Our assumptions include the following:

- *Architecture.* All sensor nodes are assumed to be homogeneous, and the network is assumed to have a flat (i.e., non hierarchical) architecture. Also, we assume that the tracking field is flat and can be described with two-dimensional Cartesian coordinates.

- *Time synchronization.* All nodes are time-synchronized using a protocol such as RBS [11].

- *Node location.* Each node knows, a priori, its own position and that of its neighbors within one hop. This knowledge can be obtained during the system's initialization phase via GPS [16] or using algorithmic strategies such as [26].

- *Target's instantaneous status.* Nodes can determine a target's movement status including its position, instantaneous velocity magnitude, and direction, either by sensing or by calculating—e.g., [4, 27, 29].

- *Target identification.* Multiple targets are assumed to be distinguished from each other using a multiple target tracking algorithm such as [19].

- *Radio transmission power.* We assume that the transmission power of sensor nodes' communication radio can be adjusted to reach different distances based on popularly used radio hardware such as CC1000 [1] and CC2420 [2]. The energy consumption with variable transmission power is determined with a model developed using curve fitting based on the empirical measurements in [28].

- *Sleep level.* Nodes are assumed to operate in two states: S_0 (active) and S_1 (sleep). A node may sample and process signals, and transmit and receive packets in state S_0. In state S_1, all modules/devices of a node will be in the sleep state, except a wake-up timer that has very low energy consumption [13].

- *Sleep pattern.* We assume that the default sleep pattern is "random" i.e., before sleep scheduling is triggered by a target detection event, all the nodes switch between active and sleep modes with the same *toggling cycle* (TC) and the same *duty cycle* (DC). However, the boundaries of each node's toggling cycle are random. In each period, a node wakes up and remains active for $TC * DC$, and then sleeps for $TC * (1 - DC)$ [13]. Upon detecting a target, SSMTT may change some nodes' sleep pattern by modifying either their toggling cycle or duty cycle.

2.2. Problem Formulation

First, we discuss the modeling of the sensor network and multiple interfering targets. We use the term *tracking subarea* to describe a sensor node set consisting of a root node and some member nodes that are awakened by the root node. Let a tracking subarea be denoted as A, a root node as γ, and member nodes as $m_0, m_1, ..., m_k$. Thus, $A = \{\gamma, m_0, m_1, ..., m_k\}$. Let the number of nodes in a tracking subarea be denoted as $|A| = k+2$. The root node γ will broadcast an alarm message to schedule the sleep pattern of its neighbor nodes (i.e., member nodes) upon detecting a target.

Let's assume that n targets $\{T_i | i \in [0, n-1]\}$ interfere with each other. Each target will trigger an alarm message and thus form a tracking subarea. Let the tracking subareas triggered by target T_i be denoted as $\{A_{ip} | p \in N_0\}$, and their root nodes as $\{\gamma_{ip} | p \in N_0\}$, where N_0 is the non-negative integer set. Then, at the time that A_{ip} is formed, A_{ip} may overlap with $\{A_jq | j \in [0, i-1] \cup [i+1, n-1], q \in N_0\}$.

Now, we define the criteria for deciding whether or not SSMTT improves energy efficiency compared with tracking multiple targets separately with a single target tracking algorithm, and how much it improves.

Since our basic idea is to enhance energy efficiency by leveraging the targets' interference, the energy consumption is a critical aspect in making this decision. When two targets are far away from each other such that there is no node that can detect both of them at the same time, they can be handled as two single targets with single target tracking and sleep scheduling algorithms. Therefore, we only consider the difference in energy efficiency during the period when multiple targets interfere.

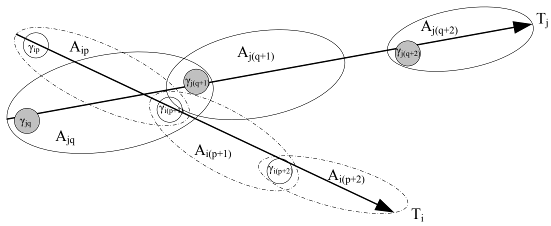

Figure 1. Interference Period ϵ_{ij}

Let the interference period of two targets, T_i and T_j, be denoted as $\epsilon_{ij} = \epsilon_{end} - \epsilon_{start}$. Here, ϵ_{start} is the time when the root nodes for T_i and T_j move close enough to hear each other for the first time ($|\gamma_{ip}\gamma_{jq}| \leq R, (p, q \in N_0)$). ϵ_{end} is the time when the root nodes of T_i and T_j move away from each other ($|\gamma_{ip}\gamma_{jq}| \geq R, (p, q \in N_0)$). Figure 1 describes the definition of ϵ_{ij} in which T_i and T_j interfere (in the figure, ellipses are used to describe tracking subareas for simplification, and the shape in Figure 6 is more close to the actual case). Assume that $|\gamma_{i(p-1)}\gamma_{j(q-1)}| > R$, $|\gamma_{ip}\gamma_{jq}| \leq R$, $|\gamma_{i(p+1)}\gamma_{j(q+1)}| < R$ and $|\gamma_{i(p+2)}\gamma_{j(q+2)}| \geq R$. Now, ϵ_{start} is the time when both A_{ip} and A_{jq} are formed (no matter which is formed later), and ϵ_{end} is the time when both $A_{i(p+2)}$ and $A_{j(q+2)}$ are formed (no matter which is formed later).

Let the tracking energy consumed during ϵ_{ij} be denoted as E_s when tracking multiple targets through single target tracking algorithms, and as E_m when tracking with SSMTT. We define the benefit that can be obtained by SSMTT as Energy Saving Ratio (or ESR), where $ESR = \frac{E_s - E_m}{E_s}$.

Detection delay is one of the most important performance feature of many surveillance sensor networks [15, 22]. Figure 2 describes the definition of detection delay, in which $\overrightarrow{AB}$ is target T_i's route in the interfering period, solid circles (or successful nodes) and dotted circles are the nodes that succeed and fail on detecting T_i respectively. As an example, t_1 and t_2 are the delays from when the target leaves the sensing range of a successful node to when it is detected again by another. Since concurrently tracking multiple targets requires to guarantee the overall quality for all the targets, we use the metric, Average Detection Delay (or ADD) for measuring the average delay per target, where $ADD = \frac{1}{i}\sum_i \sum_u t_{iu}$. Furthermore, to describe SSMTT's improvement over single target tracking algorithms, we measure ADD during ϵ_{ij} only.

We define another metric, Tracking Degree (or TD), to evaluate tracking performance. TD is defined as the percentage of the route length of a target that is covered by successful nodes divided by the target's total route length. TD can be used to measure the probability for detecting the target (the overall detection probability is generally difficult to directly measure). Higher the TD is, higher would therefore be the detection probability. Similar to ADD, we only count the route length during ϵ_{ij}. Since TD is a relative ratio, we use TD to measure the overall tracking degree of all the targets. Therefore TD is given by $TD = \frac{\sum_i \sum_v u_{iv}}{\sum_i L_i}$, where u_{iv} is T_i's route length which is not covered by successful nodes, and L_i is the total route length of target T_i in ϵ_{ij}. TD is also described in Figure 2.

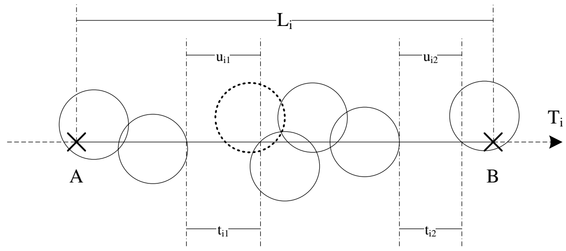

Figure 2. Detection Delay and Tracking Degree

Given these metrics, we formulate our problem as: *how to schedule the node sleep patterns and leverage the overlapping broadcasts for multiple targets, to achieve better ESR with acceptable ADD and TD loss.*

3. SSMTT Algorithm Design

The basic steps of the SSMTT algorithm include the following: (1) Describe the target movement, especially its potential moving directions, with a probabilistic model; (2) manage tracking subareas to reduce the number of proactively awakened nodes; (3) leverage the overlapping broadcasts for multiple targets to reduce the energy consumed on proactive wake-up alarm transmission; and (4) schedule the sleep patterns of the subarea member nodes to shorten their active time.

Next we discuss these ideas and their utilization in the design of the SSMTT algorithm one by one.

3.1. Target Movement Model

In the real world, a target's movement follows certain rules of kinematics in a short term, although in a long term it may still be subject to uncertainty. In fact, even for a short term, it is difficult to accurately predict a target's movement based on a physics-based model. Instead, we consider a probabilistic model, Linear Distribution Model (or LDM) to approximately imitate the actual target motion. This model is based on an assumption that a target has a higher probability to keep its current direction than to change to another, and turning around (making a 180^o U-turn) has the least probability. Since an intruding target will typically have a purpose and move towards a certain direction, this assumption is reasonable.

Based on the target's instantaneous moving direction θ_0, we define the probability $p(\theta)$ with which the target moves along the direction $\theta_0 + \theta$ ($\theta \in (-\pi, +\pi]$) as follows.

$$p(\theta) = \begin{cases} a\theta + b, \theta \in (-\pi, 0] \\ -a\theta + b, \theta \in (0, -\pi] \end{cases} \tag{1}$$

Here, a and b ($a > 0, b > 0$) are constants which can be determined by specific $(\theta, p(\theta))$ value pairs or certain constraints for a given application.

Figure 3 describes the relationship of θ_0, θ and $p(\theta)$. In the figure, "γ" is the root node, "m" is a member node, v is the target's instantaneous speed, and the dashdotted line is an alarm message sent from the root node to the member node. Figure 4 shows the probabilistic density function $p(\theta)$.

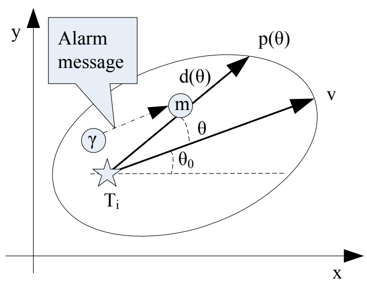

Figure 3. Target Moving Probabilities Along Different Directions

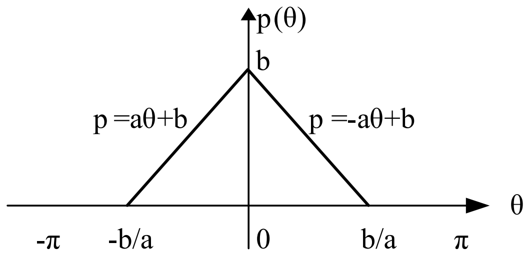

Figure 4. Linear Distribution Model

3.2. Tracking Subarea Management Mechanism

Usually, a sensor node's communication radio range R is far longer than its sensing range r. Thus a broadcast alarm message may reach all the active neighbors within the communication range. However, only some of them can detect the target, and others' energy consumed for being active is wasted. A more effective approach is to determine a node subset among all the neighbors and form a tracking subarea to reduce the number of awakened nodes.

We present a subarea management mechanism as follows, which is dispersedly implemented in the algorithm procedures in Section 4.

- *Creation.* On detecting a target, a sensor node will check if a new subarea needs to be formed. If yes, the node runs a root node election algorithm e.g., [20]. If this node is elected as root, it broadcasts an alarm message. Each neighbor who receives this alarm message will decide if it is in this subarea's scope based on the information carried in the alarm message. Finally, a new subarea gets created which will include the root node and all the neighbor nodes who decide to be part of it;

- *Recovery.* As time progresses, a subarea member's sleep pattern will recover back to the default pattern in a step-by-step manner;

- *Dismissal.* A subarea will be dismissed as each subarea member's sleep pattern individually returns to the default mode with the recovery mechanism.

Each receiver node decides a subarea's scope by determining whether or not it is in the subarea's scope. When a sensor node receives an alarm message, it computes its distance D from the root node and compares it with $d(\theta)$, where $d(\theta)$ is the subarea's radius along the direction θ. If $D < d(\theta)$, the node knows that it is a member of the subarea. Figure 3 also describes the definition of $d(\theta)$.

The subarea radius $d(\theta)$ is correlated with both v and $p(\theta)$. But we cannot setup a mapping from v and $p(\theta)$ to $d(\theta)$ for each direction θ, because the mapping parameters may vary significantly along different directions due to the irregularity of $p(\theta)$. A simplified approach is to setup a mapping for the direction $(\theta = 0)$ and only calculate $d(\theta = 0)$. Then, for all the other directions, the radius $d(\theta)$ can be derived from $d(\theta = 0)$ by proration. We describe the correlations between v, $p(\theta = 0)$, and $d(\theta = 0)$ with the simplest linear mapping shown as $d(\theta = 0) = m \cdot v \cdot p(\theta = 0) + n$, where m and n are both constants which can be determined with example value groups of v, $p(\theta = 0)$, and $d(\theta = 0)$. Substituting $p(\theta = 0)$ with $p(\theta)|_{\theta=0}$ from Equation 1, we obtain $d(\theta = 0) = m \cdot v \cdot b + n$. Then, $d(\theta)$ can be derived as $d(\theta) = \frac{p(\theta)}{p(\theta=0)} d(\theta = 0) = \frac{(-a|\theta|+b)}{b} \cdot (m \cdot v \cdot b + n)$.

We can save the energy consumption on proactive wake-up communication by leveraging the overlapping broadcasts. Therefore based on the assumption that the radio transmission power is adjustable, we present an approach to shorten $d(\theta = 0)$ for saving the transmission energy of the alarm messages in Section 3.4.

3.3. Sleep Scheduling For Subarea Member Nodes

In a tracking subarea, not all the awakened sensor nodes need to be active all the time. By scheduling their sleep pattern, we can save more energy than a single approach of reducing the number of awakened nodes. Figure 5 describes the scheduled sleep pattern that is executed on each sensor node in the tracking subarea, where X-axis represents time, and Y-axis represents duty cycle. In the figure, DC_{max} is the maximum scheduled duty cycle, DC_{min} is the default duty cycle, $t_{start} = \frac{D-r}{v_{max}}$ is the minimum time that the target can enter this node's sensing range from the current position, $t_{end} = \frac{D+r}{v_{min}}$ is the end time of the scheduled sleep pattern when the duty cycle recovers to the default value, and TC_{schd} is the scheduled toggling cycle. Here v_{max} and v_{min} are respectively the maximum and minimum speeds that are supported by a specific implementation. DC_{min} and TC_{schd} are also dependent on the detailed implementation.

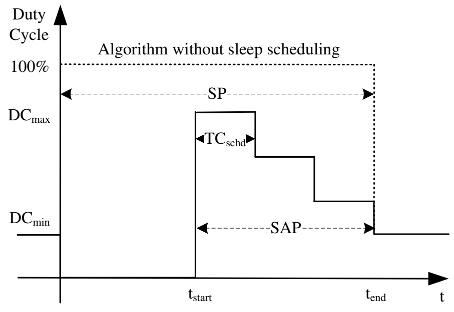

Figure 5. Scheduled Sleep Pattern of Each Subarea Member

Similar to $d(\theta = 0)$, we setup a linear mapping from v and $p(\theta)$ to DC_{max}. For LDM, DC_{max} is determined by $DC_{max} = i \cdot v \cdot (-a\,|\theta| + b) + j$, where i and j are constants which can be determined by specific $(v, p(\theta), DC_{max})$ value groups or certain constraints for a given application.

Sections 3.1, 3.2, and 3.3 are built upon the TDSS algorithm [18].

3.4. Energy Saving for Proactive Wake-up Alarm Transmission

Let the interfered target be denoted as T_i, and the interfering targets be denoted as $\{T_j | j \in [0, i-1] \cup [i+1, n-1]\}$. If just before a detection event, the root node γ_{ip} received alarm messages triggered by T_j, its tracking subarea A_{ip} may overlap with subareas $\{A_j q | j \in [0, i-1] \cup [i+1, n-1], q \in N_0\}$. If most of the member nodes in the overlapped area have already been awakened by $\{\gamma_{jq}\}$, then it will not be necessary for γ_{ip} to awaken them once again.

Now, we discuss the detailed definition and calculation of the criteria for γ_{ip} on saving this transmission energy. First, we start with the case of two targets.

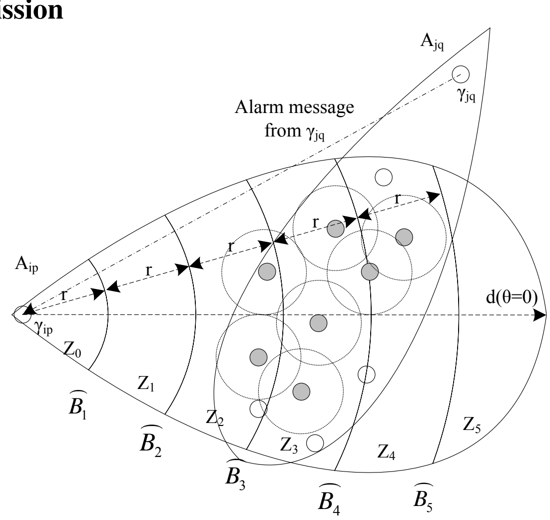

Figure 6. Interference between Two Targets

Figure 6 shows the interference between two targets T_i and T_j. In the figure, the smallest circles are sensor nodes, and the dotted circles around them denote their sensing range. Assume that γ_{ip} just received an alarm message from γ_{jq} before it detects T_i, and it finds that there would be an overlapping area between A_{ip} and A_{jq} (the dashdotted line describes the alarm message from γ_{jq} to γ_{ip}). Then γ_{ip} divides A_{ip} into zones $\{Z_k | k \in N_0\}$ based on the distance from γ_{ip}, i.e., a point $P \in Z_k \Leftrightarrow |\overrightarrow{\gamma_{ip}P}| \in (kr, (k+1)r]$, where r is the sensing range of the sensor nodes. In the figure, the arcs

$\{\widetilde{B_k}|k \in N\}$ are the boundaries between adjacent zones. The reason we divide the zones by the distance r is to facilitate the discussion on the criterion for saving the transmission.

Figure 5 also shows the definition of Scheduled Period (or SP) and Scheduled Active Period (or SAP): SP is the period that the node takes the scheduled sleep pattern instead of the default one, and SAP is the period that the node does not sleep completely in a SP. Assume that in the overlapping area of A_{ip} and A_{jq}, there are m nodes, called Overlapping Nodes (or ON) and denoted as $U_{ON} = \{ON_k|k \in [0, m-1]\}$. Now for each ON_k, we define the metric Node Overlapping Ratio (or NOR), as $NOR_k = \frac{t_{overlap}}{SAP_i}$, to describe the overlapping degree of SAP_i and SAP_j. Here SAP_i and SAP_j are the SAPs scheduled respectively for the two targets, $t_{overlap}$ is the overlapping time of SAP_i and SAP_j. Figure 7 shows their relationship.

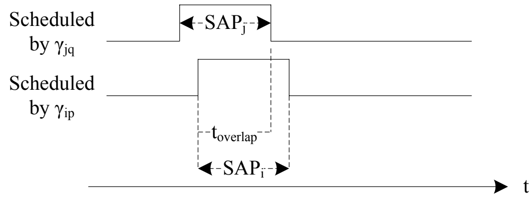

Figure 7. Node Overlapping Ratio

We call those ONs whose $NOR > THS_{NOR}$ as Reusable Nodes (or RN) and denote them as $U_{RN} = \{RN_k|k \in [0, m-1]\}$, where THS_{NOR} is a threshold specific for a given implementation. In Figure 6, the RNs are shown with solid gray circles. If the sensing coverage area of RNs in a zone Z_k is large enough so as to cover most of Z_k's area, the member nodes in Z_k may be omitted in γ_{ip}'s alarm broadcasting. For each zone Z_k, we define the metric Zone Overlapping Ratio (or ZOR) to describe the overlapping degree of two targets in this zone. The overlapping ratio of Z_k, denoted ZOR_k, is calculated as:

$$ZOR_k = \frac{\bigcup S_{RN}}{S_{all}} \tag{2}$$

Here S_{RN} is the covered area in Z_k of all the Z_k's RNs, and S_{all} is the total area of Z_k. Figure 8 shows the definition of ZOR_k, in which we use Z_3 in Figure 6 as the example. In Figure 8, $\bigcup S_{RN}$ is shown as the dotted area, and S_{all} equals to the area of $\overline{ABCD}$. We call those zones whose $ZOR_k > THS_{ZOR}$ as Reusable Zones (or RZ) and denote them as $U_{RZ} = \{Z_k|k \in N_0\}$, where THS_{ZOR} is a threshold specific for a given implementation.

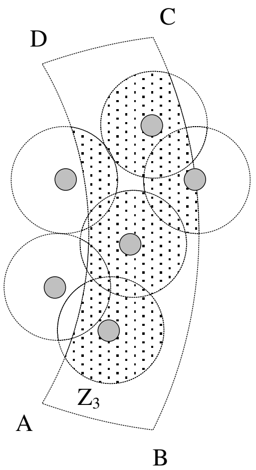

Figure 8. Definition of ZOR

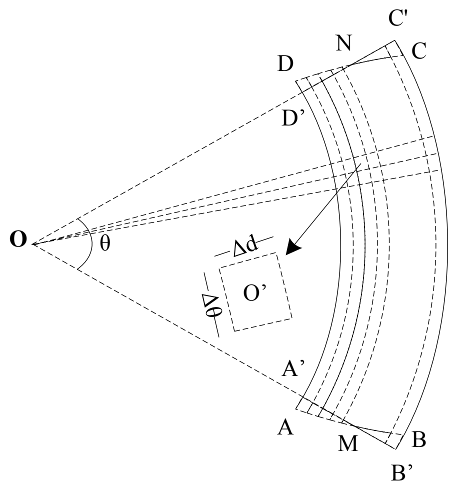

Figure 9. Calculation of Covered Area

The core idea of the energy saving effort for proactive wake-up alarm transmission is (1) to cancel the alarm broadcast completely if a zone that is close to the root node is reusable, and (2) to reduce the transmission power of the alarm broadcast if a zone that is far from the root node is reusable and all the zones that are closer to the root node than it are not reusable.

Based on the case of two targets interference, we calculate ZOR_k using Equation 2 for the multiple target case, too. However, the difference is that the RNs for calculating $\bigcup S_{RN}$ includes the reusable nodes in all of the overlapping areas of A_{ip} and each interfering target.

Next, we present the calculation of $\bigcup S_{RN}$. To reduce the computational complexity, we adopt an approximate approach and again discuss with Z_3 in Figure 6 as the example. In Figure 9, $Z_3 = \overline{ABCD}$ is determined by the tracking subarea A_{ip}'s edges, and the boundaries $\widetilde{B_3}$ and $\widetilde{B_4}$. O is the position of γ_{ip}. M and N are points on the intersection of A_{ip}'s edges and a circle with center O and radius $3.5r$ (i.e., $|OM| = |ON| = 3.5r$). θ is the central angle corresponding to the arc $\widetilde{MN}$. Now, A', B', C', and D' are points on the intersection of line OM, line ON, and the boundaries $\widetilde{B_3}$ and $\widetilde{B_4}$.

We use the area $\overline{A'B'C'D'}$ as the approximation of $\overline{ABCD}$, and divide $\overline{A'B'C'D'}$ evenly into small "sectors" with concentric circles centered at O and lines through O. Here, sectors are not mathematical sectors but more like disk sectors in the context of computer disk storage. Each sector has a segment with the radius $\Delta d = \frac{r}{a}$ and corresponds to a central angle $\Delta\theta = \frac{\theta}{b}$, where a and b are constants specific for a given implementation. Let a sector be denoted as $C(d, \theta)$. In polar coordinates with the radial coordinate ρ and the polar angle α, $C(d, \theta)$ is determined by the circle $\rho = d$, circle $\rho = d + \Delta d$, line $\alpha = \theta$, and line $\alpha = \theta + \Delta\theta$. Sector $C(d, \theta)$'s central point O' (i.e., $(d + \Delta d/2, \theta + \Delta\theta/2)$ in polar coordinates) is used as the representative of the sector. If O' is covered by a sensor node's sensing range, we consider that $C(d, \theta)$ is covered by this node.

Now, we approximate the calculation of ZOR_k in Equation 2 as $ZOR_k = \frac{SN_{RN}}{SN_{total}}$, where SN_{RN} is the number of sectors that are covered by reusable nodes, and SN_{total} is the total number of sectors in a zone.

3.5. Preventing Alarm Messages from being Missed

The consequence of the SSMTT mechanism is that, when a sensor node is scheduled to sleep by an alarm message, it may miss the alarm message broadcast for other approaching targets. Our solution to this problem is to force the member node, which has been scheduled to sleep until the expected target arrival time, to wake up with the default toggling cycle and an extremely short duty cycle. The only purpose of this is to check alarm messages from other approaching targets.

4. SSMTT Algorithm Description

To record each approaching target, a sensor node needs to manage an Alarm Message Database (or AMD). Each entry in the AMD records all the information transmitted by an alarm message including target ID, subarea ID, γ's position, the target's instantaneous movement status, and TTL (i.e., time to live) et al. The AMD is updated whenever the node receives an alarm message, irrespective of whether it will become a member of the tracking subarea.

When a node wakes up, it changes its sleep pattern according to the scheduled result and sets the wake-up timer for the subsequent wake-up. During this active period, it may detect a target or receive an alarm message, and corresponding interrupt handlers for them will be released for execution. The main functions of the SSMTT algorithm are implemented in Algorithm 1 and Algorithm 2.

5. Performance Evaluation

A. Simulation Environment. We evaluated SSMTT algorithm against three sleep scheduling algorithms for single target tracking: the legacy circle-based proactive wake-up scheme (or CIRCLE) [13], MCTA

Algorithm 1: Calculation of Reusable Zones

for *each entry in AMD* **do**
  Decide U_{ON};
$U_{RN} = \phi$;
for *each* ON_k *in* U_{ON} **do**
  Calculate NOR_k;
  if $(NOR_k > THS_{NOR})$ **then**
    $U_{RN} = U_{RN} + ON_k$;
$U_{RZ} = \phi$;
for *each zone* Z_k *in* A_{ip} **do**
  $SN_{RN} = 0$;
  $SN_{total} = 0$;
  for *each sector* $C(\lambda, \alpha)$ *in* Z_k **do**
    $SN_{total} + +$;
    if *central point* O' *of* $C(\lambda, \alpha)$ *is covered by a* RN_k *(*$RN_k \in U_{RN}$*)* **then**
      $SN_{RN} + +$;
  $ZOR_k = SN_{RN}/SN_{total}$;
  if $(ZOR_k > THS_{ZOR})$ **then**
    $U_{RZ} = U_{RZ} + Z_k$;
return U_{RZ};

Algorithm 2: Alarm Transmission Reduction

$tmp = d(\theta = 0)$;
Calculate U_{RZ} with Algorithm 1;
if $(Z_1 \in U_{RZ})$ **then**
  $tmp = 0$;
else
  for $($*int* $i = 2$*;* $i < R/r$*;* $i + +\ )$ **do**
    if $(Z_i \in U_{RZ})$ **then**
      $tmp = i * r$;
      break;
return tmp;

awakened node reducing algorithm [17], and TDSS. As discussed in Section 1, the reason that we did not compare SSMTT with other sleep scheduling algorithms with multiple target tracking support is because, we are not aware of any previous works in this area.

Parameters of our simulation environment included the following:

- *Network scale.* 400 nodes were deployed in a grid structure of 20 rows and 20 columns. The distance between two adjacent nodes was 5 (m).
- *Targets.* We tracked $2 \sim 10$ targets concurrently.
- *Target motion.* Uniform Rectilinear Motion (URM) was used to describe the target movement.
- *Target speed.* Nine target speeds were used ($\{3, 6, 9, 12, 15, 18, 21, 24, 27\}$) for the two targets case. For the case of more than two targets, the target speed was random.
- *Interfering angle.* For the two targets case, we experimented six interfering angle options ($\frac{\pi}{6}, \frac{\pi}{3}, \frac{\pi}{2}, \frac{2\pi}{3}, \frac{5\pi}{6}, \pi$). For the case of more than two targets, the interfering angles were random.
- *Number of Samples* For each combination of algorithm, number of target, target speed, and interfering angle, we simulated 50 cases.

Table 1. Energy Consumption Rates

Node status	Energy consumption rate (unit)	Node status	Energy consumption rate (unit)
Active (P_{sense})	8 (mA)	Send message (P_{send})	$\frac{2}{625}d^2 + 8.6$ (mA)
Sleep (P_{sleep})	16 (μA)	Receive message (P_{rcv})	10 (mA)

The system configuration parameters used in the simulation followed the default values given in Table 1, and the example values used in the algorithm design in Section 3. The energy consumption data in Table 1 comes from the actual Mica2 platform [9,28]. As discussed previously, the transmission power is developed from the curve fitting based on the empirical measurements, and d is the transmission distance.

B. Simulation Results. Figure 10 shows ESR on alarm communication of SSMTT over the other three reference single target tracking algorithms under different numbers of targets. We can observe that the energy saved increases as the number of interfering targets increases. This is because that the more targets interfere, the more overlapping broadcasts can be saved. Although TDSS algorithm's energy consumption on alarm transmission is the most, its overall energy consumption is still better than CIRCLE and MCTA [18].

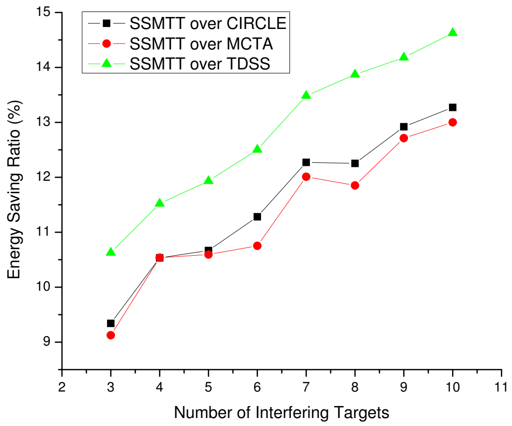

Figure 10. ESR vs. Number of Targets

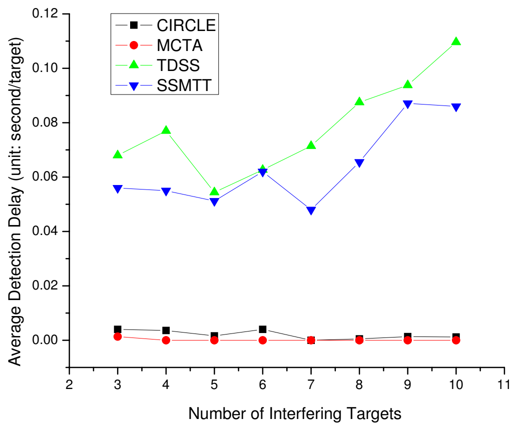

Figure 11. ADD vs. Number of Targets

Figure 11 shows ADD of the four simulated algorithms under different numbers of targets. Both TDSS and SSMTT introduce an increasing detection delay. However, this performance loss is acceptable, since in most of the cases the increased delay is within 0.1 second for each target per interference.

Figure 12 shows TD of the four simulated algorithms under different numbers of targets. SSMTT introduces a little decrease on the tracking coverage. Similar to ADD, this performance loss is negligible compared with the ESR enhancement.

We also studied the correlation between ESR and the target speed. In the two targets case, the correlation is shown in Figure 13. Basically ESR decreases as the target speed increases. This is because that the randomness of targets will increase significantly as the speed increases, therefore the overlap among interfering targets' scheduling will decrease.

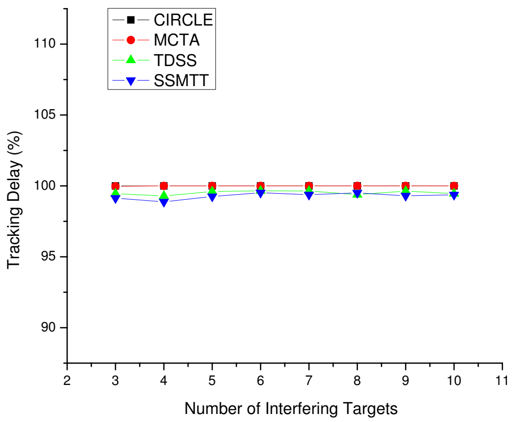

Figure 12. TD vs. Number of Targets

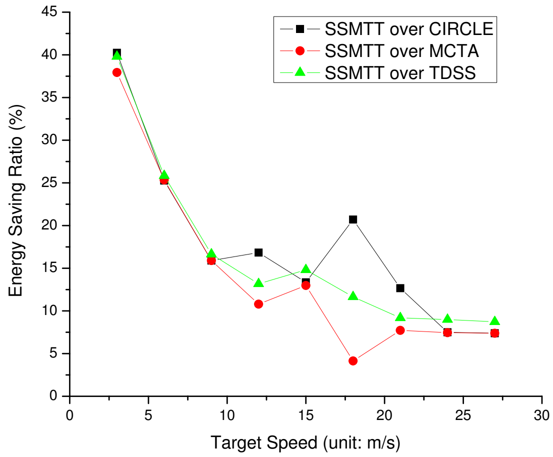

Figure 13. ESR vs. Target Speed

Figure 14 shows the correlation between ESR and the interfering angle for two targets case. We can observe that little interfering angle for two targets moving on the same direction presents the best energy saving ratio, and the worst case occurs when the interfering angle is close to $\frac{2\pi}{3}$.

6. Conclusions

In this paper, we present an energy-aware, Sleep Scheduling algorithm for Multiple Target Tracking (or SSMTT). Our objective is to improve the energy efficiency through a sleep scheduling approach that is conscious of concurrently tracking multiple targets, in contrast to an approach which is not. We introduced a linear target movement model as the foundation for energy efficiency optimization. Based on the movement model, we presented a tracking subarea management mechanism and sleep scheduling for nodes. We introduced an energy saving approach to reduce the transmission energy for alarm broadcasts. Our experimental evaluation shows that SSMTT can achieve better energy efficiency and suffer less performance loss than single target tracking algorithms.

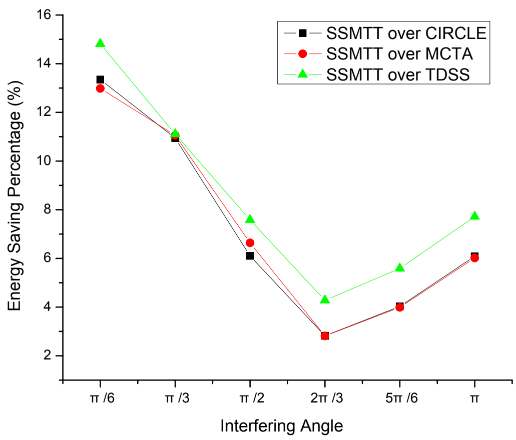

Figure 14. ESR vs. Interfering Angle

Directions for future work include: (1) Further enhance energy efficiency on the alarm message transmission with collaboration among the subareas of multiple targets; and (2) discuss the energy efficiency given specific tracking performance requirements.

References

[1] Cc1000 a unique uhf rf transceiver. http://www.chipcon.com.

[2] Cc2420 2.4 ghz ieee 802.15.4 / zigbee-ready rf transceiver. http://www.chipcon.com.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38(4):393–422, 2002.

[4] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, , and H. Z. et al. A line in the sand: A wireless sensor network for target detection, classification,and tracking, 2004.

[5] M. Athanassoulis, I. Alagiannis, and S. Hadjiefthymiades. Energy efficiency in wireless sensor networks: A utility-based architecture. In *European Wireless*, 2007.

[6] Q. Cao, T. Abdelzaher, T. He, and J. Stankovic. Towards optimal sleep scheduling in sensor networks for rare event detection. In *IPSN*, number 4, 2005.

[7] L. Chen, M. Cetin, and A. Willsky. Distributed data association for multi-target tracking in sensor networks. In *International Conference on Information Fusion*, 2005.

[8] Y. Chen and E. Fleury. A distributed policy scheduling for wireless sensor networks. In *INFOCOM*, 2007.

[9] CrossBow. Mica data sheet. http://www.xbow.com.

[10] J. Denga, Y. S. Hanb, W. B. Heinzelmanc, and P. K. Varshney. Balanced-energy sleep scheduling scheme for high density cluster-based sensor networks. In *Computer Communications: special issue on ASWN04*, volume 28, pages 1631–1642, 2005.

[11] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Oper. Syst. Rev.*, pages 147–163, 2002.

[12] L. Fan, H. Wang, and H. Wang. A solution of multi-target tracking based on fcm algorithm in wsn. In *IEEE International Conference on Pervasive Computing and Communications Workshops*, page 290, 2006.

[13] C. Gui and P. Mohapatra. Power conservation and quality of surveillance in target tracking sensor networks. In *MOBICOM*, pages 129–143, 2004.

[14] T. He, P. Vicaire, T. Yan, Q. Cao, and G. Z. et al. Achieving long-term surveillance in vigilnet. 2006.

[15] T. He, P. Vicaire, T. Yan, and L. L. et al. Achieving real-time target tracking using wireless sensor networks. *ACM TECS*, 2007.

[16] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66, 2001.

[17] J. Jeong, T. Hwang, T. He, and D. Du. Mcta: Target tracking algorithm based on minimal contour in wireless sensor networks. In *INFOCOM*, pages 2371–2375, 2007.

[18] B. Jiang, K. Han, B. Ravindran, and H. Cho. Energy efficient sleep scheduling based on moving directions in target tracking sensor network. *IEEE IPDPS*, 2008.

[19] J. Liu, M. Chu, and J. Reich. Multitarget tracking in distributed sensor networks. *Signal Processing Magazine, IEEE*, 24(3):36–46, May 2007.

[20] J. Liu, J. Liu, J. Reich, P. Cheung, and F. Zhao. Distributed group management in sensor networks: Algorithms and applications to localization and tracking. In *Telecommunication Systems*, volume 26, pages 235–251, 2004.

[21] S. Liu, K.-W. Fan, and P. Sinha. Dynamic sleep scheduling using online experimentation for wireless sensor networks. In *SenMetrics*, 2005.

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

[23] S. Oh, L. Schenato, P. Chen, and S. Sastry. A scalable real-time multiple-target tracking algorithm for sensor networks. *Memorandum*, 2005.

[24] S. Oh, L. Schenato, and S. Sastry. A hierarchical multiple-target tracking algorithm for sensor networks, 2005.

[25] J. Shin, L. Guibas, and F. Zhao. A distributed algorithm for managing multi-target identities in wireless ad-hoc sensor networks, 2003.

[26] R. Stoleru, J. A. Stankovic, and S. Son. Robust node localization for wireless sensor networks. In *EmNets*, 2007.

[27] X. Wang, J.-J. Ma, S. Wang, and D.-W. Bi. Cluster-based dynamic energy management for collaborative target tracking in wireless sensor networks. *Sensors*, 7:1193–1215, 2007.

[28] G. Xing, C. Lu, Y. Zhang, Q. Huang, and R. Pless. Minimum power configuration in wireless sensor networks. In *MobiHoc*, pages 390–401, 2005.

[29] L. Yang, C. Feng, J. W. Rozenblit, and H. Qiao. Adaptive tracking in distributed wireless sensor networks. In *Engineering of Computer Based Systems, IEEE International Symposium and Workshop*, page 9, 2006.