A Predictive Energy-Efficient Technique to Support Object-Tracking Sensor Networks

Samer Samarah, Muhannad Al-Hajri, and Azzedine Boukerche, *Senior Member, IEEE*

***Abstract*—In recent years, we have witnessed an increasing interest in deploying wireless sensor networks (WSNs) for real-life applications. However, before WSNs become a commodity, several challenging issues remain to be resolved. Object-tracking sensor network (OTSN)-based applications are widely viewed as being among the most interesting applications of WSNs. OTSN is mainly used to track certain objects in a monitored area and to report their location to the application's users. However, OTSNs are well known for their energy consumption when compared with other WSN applications. In this paper, we propose a prediction-based tracking technique using sequential patterns (PTSPs) designed to achieve significant reductions in the energy dissipated by the OTSNs while maintaining acceptable missing rate levels. PTSP is tested against basic tracking techniques to determine the appropriateness of PTSP under various circumstances. Our experimental results have shown that PTSP outperforms all the other basic tracking techniques and exhibits significant amounts of savings in terms of the entire network's energy consumption.**

***Index Terms*—Data mining, object tracking, wireless sensor networks (WSNs).**

I. Introduction

MAJOR advances in creating cost-effective, energy-efficient, and versatile microelectromechanical systems (MEMS) over the past few years has significantly contributed to the vast amount of research and development projects related to the field of wireless sensor networks (WSNs) and smart sensors. The sensor node is a very small device that represents the building blocks of the WSN. These nodes are being produced at a very low cost and yet with high levels of sophistication in terms of computing power, energy consumption savings, and multipurpose functionalities when compared with earlier generations of sensor nodes. WSNs are created by deploying a large number of sensor nodes in a certain area, which is usually called the *monitored region*, for monitoring purposes. These nodes are interconnected and are used together as a monitoring and reporting device to acquire specific types of data as desired by the application requirements [1]–[3], [19].

Object tracking, which is also called target tracking, is a major field of research in WSNs and has many real-life applications such as wild life monitoring, security applications for buildings and compounds to prevent intrusion or trespassing, and international border monitoring for illegal crossings. Furthermore, object tracking is considered one of the most demanding applications in WSNs due to its application requirements, which place a heavy burden on the network resources, particularly energy consumption. The main task of an object-tracking sensor network (OTSN) is to track a moving object and to report its latest location in the monitored area to the application in an acceptable timely manner, and this dynamic process of sensing and reporting keeps the network's resources under heavy pressure [4], [8].

OTSN is considered one of the most energy-consuming applications of WSNs. Due to this fact, there is a necessity to develop energy-efficient techniques that adhere to the application requirements of an object-tracking system, which reduce the total energy consumption of the OTSN while maintaining a tolerable missing rate level. Many researches have attempted to solve the issue of energy consumption by addressing the hardware design and the issue of optimizing the physical design of sensor nodes to produce an energy-saving sensor node. Another side of energy conservation in OTSNs was achieved by minimizing the energy consumed by the radio component (radio-frequency (RF) radio), in which these techniques had attempted to reduce the number of the messages transmitted and received, thus creating an energy-saving tracking scheme [4], [8]. However, there has been a very limited focus on the energy lost by the computing components, which are referred to as microcontroller unit (MCU) and the sensing components [4]. Although the energy dissipated by the MCU and the sensing component is less than what is consumed by the radio component, it still represents a significant source of energy consumption in the sensor node. Therefore, our concern in this paper is to develop an energy-efficient OTSN that should make remarkable reductions in the energy consumed by both the MCU and the sensing components.

In this paper, we propose the prediction-based tracking technique using sequential pattern (PTSP), which is an object-tracking technique that revolves around the ability to predict the objects' future movements to track it with the minimum number of sensor nodes while keeping the other sensor nodes in the network in sleep mode. This would achieve our goals while significantly reducing the network's energy consumption. The PTSP is based on the inherited patterns of the objects'

Manuscript received July 21, 2010; revised November 13, 2010; accepted November 19, 2010. Date of publication December 30, 2010; date of current version February 18, 2011. The review of this paper was coordinated by Prof. Y. Zhang.

S. Samarah is with the Department of Computer Information Systems, Yarmouk University, Irbed 211-63, Jordan, and also with the University of Ottawa, Ottawa, ON K1N 6N5, Canada (e-mail: samers@yu.edu.jo; ssamarah@site.uottawa.ca).

M. Al-Hajri is with Saudi Aramco, Dhahran 31311, Saudi Arabia (e-mail: malha072@site.uottawa.ca).

A. Boukerche is with the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON K1N 6N5, Canada (e-mail: boukerch@site.uottawa.ca).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2010.2102375

movements in the network and the utilization of sequential patterns (a data-mining technique) to predict to which sensor node that the moving object will be heading next. Since the PTSP totally depends on prediction, it is possible to have some missing objects during the tracking process. In addition to the main prediction technique, we propose several mechanisms to locate missed objects.

The main contribution of this paper is the prediction technique that will be used to predict the future location of a moving object. For tracking and recovery mechanisms, we adopt the framework proposed in [4].

The remainder of this paper is organized as follows: Section II reviews the related work regarding the algorithms proposed for object tracking in WSNs. Section III presents our proposed tracking framework for moving objects. Section IV provides a performance study for our proposed schemes. Section V concludes this paper.

II. Related Work

Different classification techniques have been proposed in the literature for object tracking in WSNs. The most appropriate classification is that presented by Jin *et al.* [7]. In their work, they classified object-tracking techniques into five main classes, which are naive, schedule monitoring, continues monitoring, dynamic clustering, and prediction based. However, in this section, we will focus on the first three techniques as they will be used to compare the proposed technique in this paper. For more details, see [7].

The naive scheme is considered a basic object-tracking technique, where all the sensors in the OTSN are kept active all the time, and therefore, each moving object in the network will be detected by the sensor nodes and reported to the base station every T ms [7], [9].

In the scheduled monitoring (SM) technique, all the sensor nodes in the network are allowed to stay in sleep mode; they change their status to active mode for a brief period of time where they start sensing their monitored area and report their findings to the base station, given that both the sensor nodes and the base station are well synchronized. A major concept introduced in this technique is the fact that an OTSN does not need to report its collected data all the time. Based on our application requirements, then all the sensor nodes, in this technique, are set to active mode for X ms, followed by a sleep mode period of $(T - X)$ ms. In other words, the sensor nodes wake up only when they need to report to the base station. This alternating behavior of the network between sleep and active modes is set to take place during the whole network operating time. It is obviously clear that, with the implementation of this technique, a major savings in the consumed energy is accomplished compared with the naive technique; this is because all the nodes in the OTSN network are scheduled to stay for a very short time in active mode and continue to operate in sleep mode as long as possible. On the other hand, the main drawback of this technique is that, to avoid missing reports, there are many nodes being activated to participate in the object detection process while those nodes are not actually required [7], [10], [11].

The continuous monitoring (CM) technique differs the naive and schedule monitoring techniques; it does not involve all the sensor nodes in the OTSN network to track a moving object. As an alternative, this technique allows only one sensor node to continuously stay in an active mode while it detects and tracks a moving object as long as this object stays inside the node's detection area, and the rest of the sensor nodes in the network are kept in sleep mode. The current active node will continue to monitor the moving object in its monitoring area until it leaves its coverage area and begins to enter a neighboring node detection area. At that time, the current active node will wake up the destination node to start monitoring the moving object; then, the current active node will change to sleep mode while the destination node will become the current active node. The current active node sends its activation message to the destination node W ms before the object enters the destination node's detection area. Determining the duration of the W ms is based on the transmission rate and the moving object velocity, which usually means a very small value for W. Hence, once a moving object reaches the edge of the detection area of the current active node, it will be able to determine the destination node by calculating the velocity and direction of the object. A handover will then take place where the current active node wakes up the destination node. Therefore, when an object starts moving between sensor nodes, then the sensor nodes will have to work together to keep tracking the object's movement, sending their reports to the base station on schedule for reporting. This technique's key benefit is that only one sensor node is used to track each moving object. This allows the rest of the nodes in the network to maintain their sleep mode status as long as possible, preserving their energy. Nonetheless, to avoid any missing reports, the current active node will have to keep its active mode status during the time when the moving object is still inside its detection area. This eventually will lead to the depletion of the current active node's energy [7], [12].

Several works have been proposed in the literature for using data-mining techniques, such as association rules and sequential patterns, in WSNs. Most of these works aim to generate hidden patterns in the domain of phenomena under monitoring such as temperature and medical data [17], [18], [20]. However, there are some works that have been proposed to generate patterns regarding the sensors' behavior. In [13], the authors proposed sensor association rules in an attempt to capture the temporal correlations between the sensor nodes in a particular WSN. Sensor association rules generate patterns regarding the sensor nodes; thus, these patterns can be used to enhance the quality of service (QoS) of the network, predicting the source of missed objects or estimating the value of a missed reading.

III. Prediction-Based Tracking Technique Using Sequential Pattern Framework

In this paper, we assume that the sensor nodes are static and that the network topology, including the position of each sensor node in the network, is well known to the base station. In addition, we assume that the communications between the sensor nodes in the network and base station will be based on single-hop communications.

Each sensor node in the network is required to monitor its detection area in anticipation of an intrusion by a moving object. Thus, when the sensor nodes start sensing their detection area to retrieve the object's attributes, this sensing task lasts for a specified period of time called the sampling duration ($\boldsymbol{X}$). Furthermore, while the sensor node is going through the sampling duration, its MCU and sensing components are turned on to obtain the sensed data and then process it. However, the communication component in the sensor node (RF radio) is most likely kept inactive when there are no communication requirements. The reporting frequency is controlled by the application requirements where the sensor nodes report to the application the presence of an object in their detection area every specified period of time ($\boldsymbol{T}$) [4].

To allow an active sensor node to wake up a sleeping sensor node, we assume the existence of a low-energy messaging channel [5], [6]. Due to its insignificance, we will not be considering the energy consumed as an overhead whenever a sensor node changes its mode of operation since this amount of energy is less than what is consumed when having all the nodes working in active mode and is inherited with all the techniques that use the sleep mode.

The proposed PTSP is based on two stages: 1) sequential pattern generation and 2) object tracking and monitoring. In the sequential pattern generation stage, the prediction model is built based on a huge log of data collected from the sensor network and aggregated at the sink in a database, producing the inherited behavioral patterns of object movement in the monitored area. Based on these data, the sink will be able to generate the sequential patterns that will be deployed by the sink to the sensor nodes in the network. This will allow the sensor nodes to predict the future movements of a moving object in their detection area. In the second stage, the actual tracking of moving objects starts. This stage has two parts: 1) *activation mechanism*, which entail the use of the sequential patterns to predict which node(s) should be activated to continually keep tracking of the moving object, and 2) *missing object recovery mechanism*, which will be used to find missing objects in case the activated node is not able to locate an object in its detection area. In the following, we explain these stages in more detail.

A. Sequential Pattern Generation

In this section, we highlight the main steps required for generating the sequential patterns. We start by providing a formal definition for the sequential patterns through Section III-A1 and A2, followed by a detailed explanation for the remaining steps.

1) Sequential Patterns: Formal Definition: The definition of the sequential patterns used in our prediction model following the definition provided by Samarah *et al.* [14] is inspired by the definition of sequential patterns by Agrawal *et al.* [15].

Definition 1: Let S represent the set of sensors in a certain WSN. The list $L = [(s_1, t_1), (s_2, t_2), (s_3, t_3), \ldots, (s_m, t_m)]$ denotes a list that contains the pairing of each sensor detection (represented by the sensor ID) and its equivalent time of detection, where $s_i \in S$ and $t_i <= t_{i+1}$ for all $1 <= i < m$.

The pair (s_i, t_i) represents the detection of a certain event by the sensor s_i, which took place at time t_i. $O(L) = [s_1, s_2, s_3, \ldots, s_m]$ represents the list of sensors shown in L, except that it was chronologically ordered based on the time of detection recorded by each sensor. O called the list's sequential pattern.

Definition 2: The pattern $O = [s_1, s_2, s_3, \ldots, s_m]$ is considered a subpattern of the pattern $O' = [s'_1, s'_2, s'_3, \ldots, s'_n]$. This can be interpreted as $O \sqsubseteq O'$, if $m <= n$, and a list of incremental integers exists as $i_1, i_2, i_3, \ldots, i_m$, where $1 \leq i_1 < i_2 < \cdots < i_m \leq n$, and $s_k = s'_{ik}$ for $k = 1, 2, .., m$. Thus, O is designated as the "subpattern," whereas O' is called the "superpattern."

Definition 3: Let DS be a database of sensor lists. The support of the pattern O in DS is defined by the number of lists in DS, where the sequential pattern O represents a subpattern of the lists' sequential patterns. Thus, $Support(O) = |\{L \in DS | O \sqsubseteq O(L)\}|$.

Therefore, the main goal in mining the sensors' sequential patterns with a database of lists (DS) and a minimum support (min_sup) is to identify all the sensor sequential patterns that satisfy the user-defined minimum support; these patterns are called frequent sensor sequential patterns.

2) Sequential Patterns and Trisensor Patterns: We have discussed the sequential patterns in the previous section. However, in our proposed prediction model, we are going to use a special form of sequential patterns, i.e., trisensor patterns. Trisensor patterns can be represented as follows: $[SourceSensor, CurrentSensor, DestinationSensor]$.

This represents the chronological ordering of the sensors' detection of a certain moving object in the network. Therefore, the $source\ sensor$ represents the sensor ID of the sensor, which detected a particular moving object o before it moved to the next sensor designated as the $current\ sensor$; later on, the object o moves to the next sensor denoted by $destination\ sensor$.

Furthermore, to evaluate the statistical significance of a certain trisensor pattern, we have to calculate the *confidence*, in addition to its support, of each particular trisensor pattern. This can be viewed as the estimate of the probability $P(Y|X)$, which is the probability of finding the right-hand side of the pattern in sequences while these sequences also contain the left-hand side of the pattern. Therefore, the confidence of trisensor pattern $[sourcesensor, currentsensor] \Longrightarrow [destinationsensor]$ is calculated as follows:

$$\frac{supp\,([\text{source sensor, current sensor, destination sensor}])}{supp\,([\text{source sensor, current sensor}])}.$$

To commence the patterns' discovery process, the first phase begins by aggregating the data of the sensors' detections and arranging them in a chronological order. As explained in Section III-A-6, our methodology describes the generation of sequential patterns to use them in our prediction model.

3) Movement Pattern Acquisition: The first major part of the proposed prediction technique is generating object movement patterns. In the real world and depending on the terrain being monitored and types of objects that move (animals, enemy movements, etc.), certain paths tend to more frequently occur. For example, in a forest area, there may be only two or three available routes for animals to reach a water source, because

TABLE I
OBJECTS MOVEMENT LOG

Time	Sensor ID	Object ID
00:01	S3	Obj1
00:01	S9	Obj4
00:01	S4	Obj2
00:01	S12	Obj3
00:02	S6	Obj1
..:..	..	..

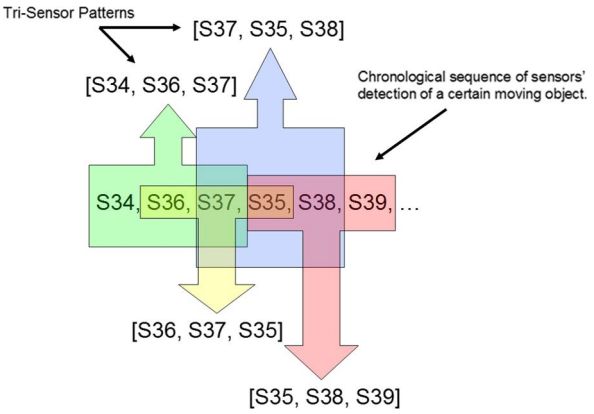


Fig. 1. Trisensor pattern generation.

other paths are blocked by heights or other obstacles. The proposed scheme relies on deploying the WSN and simply turning it on for the whole time to record movement patterns collected from sensors (basically the naive technique). As every object moves, its movement path and the sequence of sensors detecting it get recorded into a database. Each tuple in the database will contain the object ID and the ID of the sensor that detects this object, along with the time of detection. Table I shows an example of a movement log.

After having sufficiently executed this monitoring and recording process, the next step is to analyze these data and convert it into movement patterns. The trisensor movement patterns discussed in the previous section will be used. A pattern simply records the $source\ sensor$ of a certain object (the sensor where the object used to be before moving toward the current sensor), the $current\ sensor$, and the sensor the object traveled to next, which is also called the $destination\ sensor$. For example, if an object moved through the sequence of sensors S_3, S_6, S_8, and S_7, we record an instance of the pattern S_3, S_6, S_8 and pattern S_6, S_8, S_7. Therefore, we can utilize the data collected in the object history movement log file (Table I) and transform it into sequences of sensors triggered by each object movement; here, each sequence of sensors will be transformed into trisensor patterns in the following form: $[SourceSensor, CurrentSensor, DestinationSensor]$

These patterns are generated in an overlapping manner where the $Current\ Sensor$ becomes the $Source\ Sensor$ and the $Destination\ Sensor$ becomes the $Current\ Sensor$, whereas the $Destination\ Sensor$ will be the next sensor in the sequence. Fig. 1 demonstrates how the trisensor patterns are generated in an overlapping manner from the sequence of detections by the sensor nodes of the OTSN for a particular moving object. Table II shows an example of aggregated trisensor patterns.

As the patterns accumulate, we can find the most repeated patterns and draw conclusions there. For example, if we found that the pattern S_3, S_6, S_8 too frequently occurs, we can tell that, if an object is at S_6, coming from S_3, then, most probably, it will head to S_8. We call this process sequential pattern generation. Table III shows the collection of sequential patterns, based on their occurrence from the generated trisensor patterns in the previous step, as shown in Table II.

By applying simple mathematics, we can derive the following sequential patterns in Table IV from the aforementioned patterns, along with their confidences.

Therefore, prediction is achieved using these sequential patterns. The higher the confidence, the better the chance of predicting the next sensor right. As we run our technique, we should also keep updating these patterns as new objects move. This makes our scheme adaptive to changing movement patterns, which we refer to as "active learning."

Furthermore, the sequential patterns generated in Table IV can also be represented using the implication form, which can be shown as follows:

$$
\begin{aligned}
&[S77, S87] \Longrightarrow S86, Conf = 79.5\% \\
&[S77, S87] \Longrightarrow S88, Conf = 20.5\% \\
&[S96, S87] \Longrightarrow S86, Conf = 2.5\% \\
&[S96, S87] \Longrightarrow S88, Conf = 97.5\% \\
&[S92, S90] \Longrightarrow S93, Conf = 35.2\% \\
&[S92, S90] \Longrightarrow S91, Conf = 64.8\%.
\end{aligned}
$$

4) Sensor Sequential Patterns' Deployment: In this step, we will embed all the generated sensor sequential patterns to their respective sensor nodes, which resembles the second and final step in building the prediction model of our tracking technique. However, these sequential patterns will be continuously evaluated and updated to maintain a high level of accuracy in terms of our technique's prediction abilities.

B. Object Tracking and Monitoring

Object tracking consists of two stages: sensor node activation occurs when the next sensor that should wake up is decided, whereas missing object recovery, which is the second stage, involves the location of missing objects. In this section, we explain these stages in more detail.

1) Sensor Node Activation Mechanism: After the completion of the first stage of the PTSP, i.e., sequential pattern generation, the second stage, which is object tracking and monitoring, starts. The key objective of this stage is to keep in sleep mode, for the longest possible period, any sensor node that has no object moving in its detection area, thus saving its energy. Moreover, in the case of a moving object in the vicinity of a certain sensor node, this sensor node will not be awake all the time. It ought to switch to sleep mode as long as possible while not impairing the tracking process; this sensor node will be called the $current\ sensor$. The current sensor switches to active mode for X ms, and during this time, the current sensor senses its detection area. It reports the findings to the base station by the end of X, which also matches the end of T (the time period between reports). Fig. 2 shows the relation between X, T, and $(T - X)$ [4]. The last action the current sensor performs before

TABLE II
GENERATING THE TRISENSOR PATTERNS FROM THE SEQUENCE OF DETECTIONS BY SENSOR NODES FOR EACH OBJECT

Object ID	Chronological Sequence of Sensors' Detection	Tri-Sensor Patterns Generated
Obj1	S3, S6, S8, S7, S5, S6, S4, S5,...	[S3, S6, S8], [S6, S8, S7], [S8, S7, S5], [S7, S5, S6], [S5, S6, S4], [S6, S4, S5],...
Obj2	S12, S15, S16, S17, S18, S19, S21, S20, S18,...	[S12, S15, S16], [S15, S16, S17], [S16, S17, S18], [S17, S18, S19], [S18, S19, S21], [S19, S21, S20], [S21, S20, S18],...
Obj3	S52, S51, S54, S53, S50, S48, S46, S47, S49, S51,...	[S52, S51, S54], [S51, S54, S53], [S54, S53, S50], [S53, S50, S48], [S50, S48, S46], [S48, S46, S47], [S46, S47, S49], [S47, S49, S51],...
Obj...	S..., S..., S...,...	[S..., S..., S...], ...

TABLE III
AGGREGATING TRISENSOR PATTERNS BASED ON THEIR OCCURRENCE

Source Sensor	Current Sensor	Destination Sensor	Occurrence
S77	S87	S86	739
S77	S87	S88	191
S96	S87	S86	24
S96	S87	S88	936
S92	S90	S93	571
S92	S90	S91	1052

TABLE IV
SEQUENTIAL PATTERNS GENERATED FROM TRISENSOR PATTERNS' FREQUENCY OF OCCURRENCE

Source Sensor	Current Sensor	Destination Sensor	Confidence
S77	S87	S86	79.5%
S77	S87	S88	20.5%
S96	S87	S86	2.5%
S96	S87	S88	97.5%
S92	S90	S93	35.2%
S92	S90	S91	64.8%

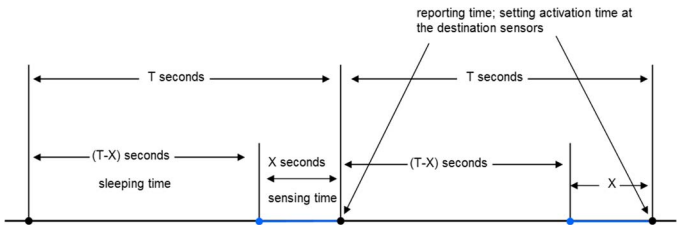


Fig. 2. Demonstrating the relation between X, T, and $(T - X)$. [4].

switching to sleep mode is predicting the location of the moving object during the $(T - X)$ milliseconds and sending a message to the destination sensors to wake up after the $(T - X)$. The current sensor will utilize the embedded sequential patterns to decide which destination sensor it will activate. The decision to apply sequential pattern(s) on a particular sensor is based on the minimum confidence level required by the application, which would be required from any sequential pattern before it could be used to predict the future movements of a moving object. Therefore, if the application sets the minimum confidence level at 70%, then any sequential pattern with a confidence level lower than 70% will not be executed. Determining the most appropriate minimum confidence level requires a thorough analysis of the prediction model performance, which will be discussed in future research.

Both the current sensor and the destination sensors will activate after the passing of $(T - X)$ ms and start tracking the moving object. In addition, once one of those active sensor nodes detects the object in its vicinity, it will send an acknowledgment message (ACK) to the former current sensor, unless it is the current sensor. This action is performed by the sensor node to become the new current sensor so that it may start the previously illustrated process. However, the rest of the active sensor nodes will turn to sleep mode.

2) Missing Object Recovery Mechanisms: Since we are using a prediction technique to determine the future movements of a moving object, it is possible to have some missing objects during the tracking process. Therefore, there is a need to develop a solution to find any missing object and return the network to the prediction tracking process (or the normal state, as we call it). Toward this, we have implemented three recovery mechanisms adopted from [4] to determine which recovery mechanism would generate the lowest energy consumption. The developed techniques are source recovery, destination recovery, and all neighbors recovery. In the following, we will explain these mechanisms in more detail.

1) *Source recovery mechanism:* In this recovery mechanism, the current sensor will activate all its neighboring sensor nodes if the object is not in its detection area and if it did not receive an ACK message from the destination sensor(s) after the passing of a certain timeout period.
2) *Destination recovery mechanism:* This recovery mechanism is similar to the previous mechanism, except that the current sensor will activate all the neighboring sensors of the destination sensor, instead of the neighboring sensors of the current sensor.
3) *All neighbors recovery mechanism:* This recovery mechanism combines both previous recovery mechanisms since the current sensor will activate all its neighboring sensor nodes in addition to the destination sensor neighboring sensor nodes.

No matter which recovery mechanism is used, if the missing object is not found, then the current sensor will initiate the second step of recovery, which is called *all network recovery*. In this step, the current sensor will send an activation message to all the sensor nodes in the network to start looking for the missing object. If no sensor node finds the missing object during the second phase of recovery, then the object must have left the network monitored region. In both recovery phases, the current sensor uses an ultralow-energy messaging channel to activate the sensor nodes for recovery. The missing object recovery message includes the time at which the sensor nodes will activate and start looking for the missing object. During the first phase of recovery, if any sensor node locates the missing object, it then must inform the current sensor through an ACK message to avoid initiation of the second phase of recovery,

TABLE V
SIMULATION SETTINGS

Number of Sensors	100 sensor nodes
Monitored Region	$150 \times 150\ meter^2$
Sensor Coverage Range	15 m
Network Topology	2D Grid topology
Default Object Speed	5 m/s
Minimum Confidence	50%
Sampling Duration	100 ms
Time between reports	1000 ms
Simulation Duration	140 seconds
Number of trails	500

TABLE VI
SENSOR NODE ENERGY CONSUMPTION [4]

Component	Mode	Energy Consumption (mW)
MCU	Active Mode	360
MCU	Sleep Mode	0.9
Sensing Component	Active	23

which involves all the sensor nodes in the network. In the second phase of recovery, the ACK message is not required: if the object is not found, then it must have left the network; if it is found, then the sensor node that located the missing object will automatically become the current sensor [4].

IV. PERFORMANCE EVALUATION

To evaluate the PTSP in a comprehensive manner, different scenarios and settings have been implemented using a stand-alone simulator. The simulation experiments carried out in an OTSN of 100 logical sensor nodes in a 150×150 m^2 monitored region. It is assumed that each sensor node will have a coverage range of 15 m. The implementation of the OTSN network is based on the 2-D grid topology [16]. We have assumed that there are a certain number of key paths that an object may follow; in addition, we have assumed that an object may choose a random path. The ratio of the key paths to the random paths is 3 : 1. The sampling duration will be 100 ms (X), and the OTSN will send a report regarding the location of the moving object every 1000 ms (T) to the application. Each simulation experiment will last for 140 s, which will include more than one object. We have conducted 500 trails (i.e., experiments) to get an average value of the results, particularly for that of the energy consumption of the network and the missing rate. See Table V for a summary of the simulation settings.

As for the energy consumption, we have adopted the WINS energy consumption for sensor nodes used in PES [4] (see Table VI). We have only included the MCU and the sensing components' energy consumption since our proposed tracking technique focuses on reducing the energy consumption on those two components. The energy consumption for the radio component will be the subject of our future research.

We will show different charts depicting performance under variations of the preceding parameters, and we will thoroughly analyze their effect on the results.

Two metrics have been used in the performance analysis.

1) *Total energy consumed:* the amount of energy consumed by the whole network to monitor moving objects, which include the active and sleep modes during the simulation;

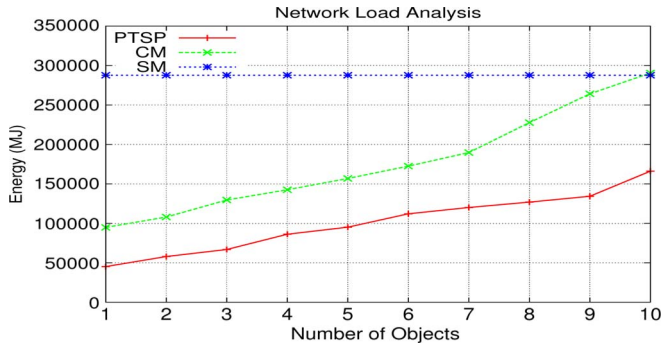


Fig. 3. Network workload analysis.

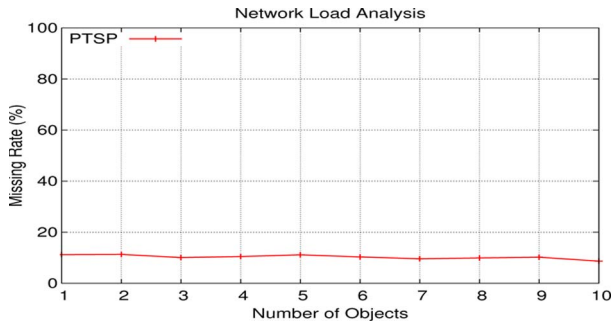


Fig. 4. Missing rate levels against the network workload.

2) *Missing rate:* The missing reports' percentage to the total number of reports that should have been received by the application; as mentioned earlier, the location of the next sensor node that the object will be heading is predicted using sequential patters. Since there is no 100% prediction, it is necessary to have this metric as a basic for comparison.

A. Energy Consumption Analysis

In this experiment, we have tested and compared PTSP to the other basic tracking schemes, i.e., SM and CM, in the context of network workload, which is represented by the number of moving objects in the network. The experiments were conducted by increasing the number of moving objects from one to ten objects.

Fig. 3 shows the energy consumption of the three schemes. Conversely, we noticed a dramatic reduction in the energy consumption gap between SM and both CM and PTSP. This occurred when the number of objects increases to the point where CM matches SM in regard to the energy consumed, when the object count reaches ten objects. These results were due to the fact that SM is always activating all its sensor nodes, regardless of the number of objects, and the only increase of energy would be attributed to the extra reports sent to the base station. As a result, if the network has a large number of moving objects, then the most ideal scheme would be SM. In the case of CM and PTSP, the increase in the number of objects means an increase in the number of active sensor nodes and, thus, higher energy consumption levels.

For the missing rate analysis, we will only consider PTSP since the missing rate for the other basic tracking techniques (SM and CM) is always 0%. We can notice in Fig. 4 that the missing rate levels are not impacted by increases in the number

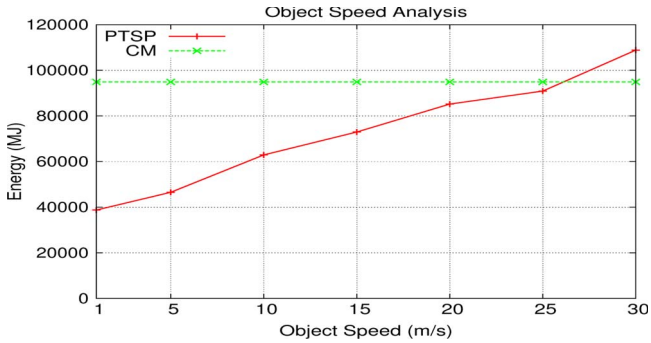


Fig. 5. Object movement speed energy consumption analysis.

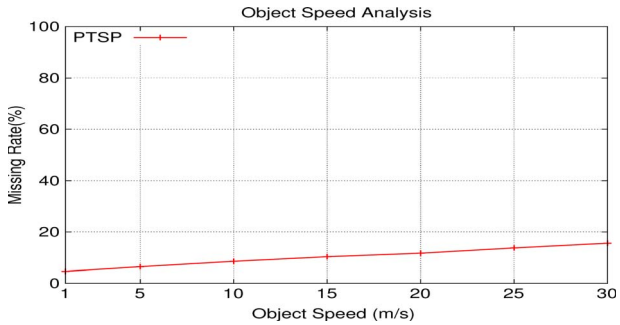


Fig. 6. Object movement speed and missing rate analysis.

of moving objects as expected since the missing rate is the ratio of the missing reports to the total number of reports. Therefore, this ratio is not affected by the increase in the number of objects. Even though it is likely that the number of missing reports will increase, this number will be matched with an increase in the number of total reports. Thus, the ratio remains unchanged.

Another experiment has also been conducted to evaluate how the changes in the speed of a moving object could affect the energy consumed by a tracking technique. Additionally, since the CM is better than the SM, when there is a low number of objects in the network, we will only use the CM for the purposes of comparison against PTSP. As shown in Fig. 5, we can notice a linear growth, for PTSP, in the energy consumption levels when the object speed also increases. This results from the fact that, when the object moves with a faster speed, the prediction for the destination sensor will be harder; thus, a greater recovery process is required and, eventually, an increase in the overall energy consumption of the network. We noticed that PTSP was outperforming CM when the object speed was below 30 m/s. This is considered an excellent performance when compared with the speed of the object; tracking an object moving at a speed of 25 m/s, for example, is not an easy task, if energy saving is also a factor of the tracking technique. The energy consumption of the CM is not effected by the object's speed, as it will be always able to locate the object.

It is apparent that, in the context of the missing rate levels, PTSP has kept an acceptable level of missing rate, although it did increase along with the increases in the object speed (see Fig. 6).

B. Recovery Mechanism Analysis

In this experiment, we have evaluated the three previously explained recovery mechanisms (source recovery, destination recovery, and all neighbors recovery) in terms of energy consumption. We can notice in Fig. 7 that the source recovery mechanism is the best in terms of energy consumption, which is based on activating the neighboring sensor nodes of the current sensor.

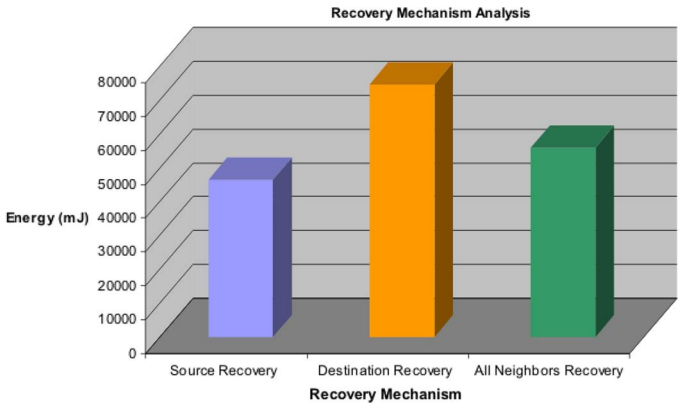


Fig. 7. Recovery mechanisms and energy consumption analysis.

The next best recovery mechanism is the all neighbor recovery mechanism, which combines the source and destination recovery mechanisms by activating all the neighboring sensor nodes of both the current sensor and the destination sensor. As for the worst recovery mechanism in terms of energy consumption, it was the destination recovery mechanism. This recovery mechanism became the worst because it required the network to go through the second phase of recovery more often than the other recovery mechanisms. Since the second phase of recovery involves activating all the sensor nodes in the network, it incurs more energy consumption in comparison with the first phase of recovery. Therefore, we have chosen the first recovery mechanism (source recovery) as our recovery mechanism of choice in all the previous experiments since it was the most energy-conservative recovery mechanism.

V. Conclusion

In this paper, we have proposed an energy-efficient PTSP, which implements a novel approach in its prediction mechanism. PTSP utilizes the sensor sequential patterns to produce accurate predictions of the future movements of a certain object. These sequential patterns are continuously evaluated and updated to provide the prediction mechanism with the latest and most accurate predictions. We have simulated our proposed tracking scheme (PTSP), along with two basic tracking schemes for comparison purposes. The results generated by the experiments were mainly through testing the performance of PTSP and the other tracking schemes against two main metrics: 1) total energy consumed by the network during the simulation period, including the active and sleep mode energy consumption for each sensor node in the network, and 2) missing rate, which represents a ratio of the missing reports to the total number of reports received by the application. Moreover, it has been proven by the simulation results that PTSP outperformed all the other tracking schemes by keeping a low energy consumption level while maintaining an acceptable level of missing rate.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE Commun. Mag.*, vol. 40, no. 8, pp. 102–114, Aug. 2002.

[2] F. Zhao and L. J. Guibas, *Wireless Sensor Networks. An Information Processing Approach*. San Mateo, CA: Morgan Kaufmann, 2002.

[3] W. Alsalih, H. Hassanein, and S. Akl, "Placement of multiple mobile data collectors in wireless sensor networks," *Ad Hoc Netw.*, vol. 8, no. 4, pp. 378–390, Jun. 2010.

[4] Y. Xu, J. Winter, and W.-C. Lee, "Prediction-based strategies for energy saving in object tracking sensor networks," in *Proc. IEEE Int. Conf. Mobile Data Manage.*, Berkeley, CA, 2004, pp. 346–357.

[5] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, and T. Tuan, "Picoradios for wireless sensor networks: The next challenge in ultra-low-power design," in *Proc. Int. Solid-State Circuits Conf.*, San Francisco, CA, 2002, pp. 200–201.

[6] J. M. Rabaey, M. J. Ammer, J. L. da Silva, Jr., D. Patel, and S. Roundy, "Picoradio supports ad hoc ultra-low power wireless networking," *Computer*, vol. 33, no. 7, pp. 42–48, Jul. 2000.

[7] G.-Y. Jin, X.-Y. Lu, and M.-S. Park, "Dynamic clustering for object tracking in wireless sensor networks," in *Proc. 3rd Int. Symp. UCS*, Seoul, Korea, 2006, pp. 200–209.

[8] Y. Xu and W.-C. Lee, "On localized prediction for power efficient object tracking in sensor networks," in *Proc. 1st Int. Workshop Mobile Distrib. Comput.*, 2003, pp. 434–439.

[9] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proc. 33rd Hawaii Conf. Syst. Sci.*, 2000, pp. 3005–3014.

[10] A. Manjeshwar and D. P. Agrawal, "TEEN: A routing protocol for enhanced efficiency in wireless sensor networks," in *Proc. 15th Int. Parallel Distrib. Process. Symp.*, San Francisco, CA, 2001, pp. 2009–2015.

[11] A. Manjeshwar and D. P. Agrawal, "APTEEN: A hybrid protocol for efficient routing and comprehensive information retrieval in wireless sensor networks," in *Proc. 16th Parallel Distrib. Process. Symp.*, 2002, pp. 195–202.

[12] S. Balasubramanian, I. Elangovan, S. K. Jayaweera, and K. R. Namuduri, "Distributed and collaborative tracking for energy-constrained ad-hoc wireless sensor networks," in *Proc. Wireless Commun. Netw. Conf.*, Atlanta, GA, 2004, pp. 1732–1737.

[13] A. Boukerche and S. Samarah, "A novel algorithm for mining association rules in wireless ad hoc sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 7, pp. 865–877, Jul. 2008.

[14] S. Samarah and A. Boukerche, "Chronological tree—A compressed structure for mining behavioral patterns in wireless sensor networks," *J. Interconnection Netw.*, vol. 9, no. 3, pp. 255–276, Sep. 2008.

[15] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. 11th Int. Conf. Data Eng.*, Taipei, Taiwan, 1995, pp. 3–14.

[16] H. G. Goh, M. L. Sim, and H. T. Ewe, "Energy efficient routing for wireless sensor networks with grid topology," in *Proc. Int. Fed. Inf. Process.*, Santiago de Chile, Chile, 2006, pp. 834–843.

[17] K. Loo, I. Tong, and B. Kao, "Online algorithms for mining inter-stream associations from large sensor networks," in *Proc. 9th Pacific-Asia Conf. Knowl. Discovery Data Mining*, Hanoi, Vietnam, 2005, pp. 143–149.

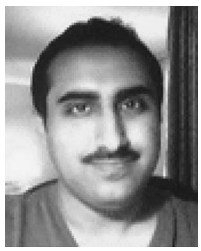
[18] K. Romer, "Distributed mining of spatio-temporal event patterns in sensor networks," in *Proc. 2nd IEEE Int. Conf. Distrib. Comput. Sensor Syst.*, San Francisco, CA, 2006, pp. 103–116.

[19] M. Naderan, M. Dehghan, and H. Pedram, "Mobile object tracking techniques in wireless sensor networks," in *Proc. Int. Conf. Ultra Modern Telecommun.*, St. Petersburg, Russia, 2009, pp. 1–8.

[20] M. Halatchev and L. Gruenwald, "Estimating missing values in related sensor data streams," in *Proc. 11th Int. Conf. Manage. Data*, Goa, India, 2005, pp. 83–94.

Samer Samarah received the Ph.D. degree in computer science from the University of Ottawa, Ottawa, ON, Canada, in 2008.

He is currently an Assistant Professor with the Department of Computer Information System, Yarmouk University, Irbed, Jordan, and a Research Associate with the School of Information Technology and Engineering, University of Ottawa. His research interests are wireless networks, wireless ad hoc and sensor networks, and data mining for both distributed systems and wireless sensor networks.



Muhannad Al-Hajri received the M.S. degree in computer science from the University of Ottawa, Ottawa, ON, Canada, in 2010.

He is currently with Saudi Aramco, Dhahran, Saudi Arabia. His research interests are object-tracking wireless sensor networks.

Azzedine Boukerche (SM'09) received the Ph.D. degree in computer science from McGill University, Montreal, QC, Canada.

He is currently a Full Professor and holds a Canada Research Chair position with the School of Information Technology and Engineering, University of Ottawa, Ottawa, ON, Canada. He is a fellow of the Canadian Academy of Engineering and the founding Director of the PARADISE Research Laboratory, School of Information Technology and Engineering, Ottawa. Prior to this, he held a faculty position with the University of North Texas, Denton, and was a Senior Scientist with the Simulation Sciences Division, Metron Corporation, San Diego, CA. He was also a faculty member with the School of Computer Science, McGill University, and taught at the Polytechnic of Montreal. He spent a year at the Pet Propulsion Laboratory, National Aeronautics and Space Administration (JPL/NASA), California Institute of Technology, Pasadena, where he contributed to a project centered around the specification and verification of the software used to control interplanetary spacecraft operated by the JPL/NASA Laboratory. He has published several research papers in these areas. He serves as an Associate Editor for *Elsevier Ad Hoc Networks*, the *Wiley International Journal of Wireless Communication and Mobile Computing*, *Wiley's Security and Communication Network Journal*, the *Elsevier Pervasive and Mobile Computing Journal*, *Elsevier's Journal of Parallel and Distributed Computing*, and *SCS Transactions on Simulation*. He served as a Guest Editor for the *Journal of Parallel and Distributed Computing* (Special Issue on Routing on Mobile Ad Hoc, Special Issue on Wireless Communication and Mobile Computing, and Special Issue on Mobile Ad Hoc Networking and Computing), *ACM/Kluwer Wireless Networks*, *ACM/Kluwer Mobile Networks Applications*, and the *Journal of Wireless Communication and Mobile Computing*. His current research interests include wireless ad hoc and sensor networks, wireless networks, mobile and pervasive computing, wireless multimedia, quality-of-service (QoS) service provisioning, performance evaluation and modeling of large-scale distributed systems, distributed computing, large-scale distributed interactive simulation, and parallel discrete-event simulation.

Dr. Boukerche served as the General Chair for the Eighth ACM/IEEE Symposium on Modeling, Analysis, and Simulation of Wireless and Mobile Systems and the Ninth ACM/IEEE Symposium on Distributed Simulation and Real-Time Application (DS-RT). He was the Program Chair for the ACM Workshop on QoS and Security for Wireless and Mobile Networks, the ACM/IFIPS Europar 2002 Conference, the IEEE/SCS Annual Simulation Symposium (ANNS 2002), ACM WWW 2002, IEEE MWCN 2002, IEEE/ACM MASCOTS 2002, IEEE Wireless Local Networks WLN (2003–2004), IEEE WMAN (2004–2005), and ACM MSWiM (1998–1999). He served as Vice General Chair for the Third IEEE Distributed Computing for Sensor Networks (DCOSS) Conference in 2007 and Program Cochair for GLOBECOM 2007–2008 Symposium on Wireless Ad Hoc and Sensor Networks and for the 14th IEEE ISCC 2009 Symposium on Computer and Communication Symposium. He serves as an Associate Editor for the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, the IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and *IEEE Wireless Communication Magazine*. He also serves as a Steering Committee Chair for the ACM Modeling, Analysis, and Simulation for Wireless and Mobile Systems Conference and the ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks. He is a cofounder of the International Conference on Quality of Service for Wireless/Wired Heterogeneous Networks (QShine 2004). He was the recipient of the Best Research Paper Award at IEEE/ACM PADS 1997, ACM MobiWac 2006, ICC 08-09, and IWCMC 2009. He received the Third National Award for Telecommunication Software in 1999 for his work on distributed security systems on mobile phone operations, the Ontario Early Research Excellence Award (previously known as Premier of Ontario Research Excellence Award), the Ontario Distinguished Researcher Award, and the Glinski Research Excellence Award.