POOT: An efficient object tracking strategy based on short-term optimistic predictions for face-structured sensor networks

Jenq-Muh Hsu, Chao-Chun Chen, Chia-Chi Li

Abstract

The advance of wireless sensor networks has enabled the development of a great number of applications in various fields, such as biology, military, and environmental surveillance. Among these applications, object tracking systems have particularly useful functions, and have been studied by many researchers in recent years. In the design of a sensor network system, energy consumption is a critical consideration. In this paper, we propose a short-term Prediction-based Optimistic Object Tracking strategy (POOT) to reduce energy consumption and prolong the lifetime of sensor nodes while sacrificing only minimal tracking precision. Furthermore, we present two schemes, a Time-efficient Object Recovery Scheme (TORS) and a Communication-efficient Object Recovery Scheme (CORS), to improve object recovery. We also derive cost models for POOT. Through a set of experiments, our proposed prediction-based optimistic object tracking scheme can save up to 23% energy consumption compared to the related scheme, DOT. Meanwhile, the accuracy of POOT is still higher than 97.5% which reveals the optimistic design does not affect the tracking accuracy. Hence, POOT is shown to effectively conserve energy and achieve the objective of tracking of moving objects.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The advance of wireless sensor networks has enabled the development of a great number of applications in various fields, such as biology, military, and environmental surveillance. Among these applications, object tracking systems are particularly useful [1]. Since current sensor nodes are battery powered, energy consumption is the most critical consideration in design of object tracking systems. If energy consumption of an object tracking system is low, service life of a wireless sensor network can be prolonged. Hence, object tracking systems have become a popular research topic in recent years [2].

Many researchers have studied the object tracking issues in the related literature [2–11]. These studies can be classified into two categories: one category of research has focused on collection and transmission of data in order to update locations of moving objects, and the other category has focused on discovering object location in a distributed sensor network. In the first category, researchers have focused on reducing the communication cost (i.e., the number of messages) in object tracking networks [7,12]. Lin et al. proposed two schemes, DAT and Z-DAT [7] based on a tree structure. Chen et al. proposed DOT [12] to optimize energy consumption based on a face structure. These studies aimed at reducing communication costs, however, they did not discuss the effect of reducing the number of wake-up faces on object tracking. For the second category, researchers proposed schemes that track moving objects while consuming less energy in a sensor network. The Frisbee
model was proposed to activate all of an object's nearby nodes in order to track object location [13,14]. The idea of Frisbee is to dynamically organize a cluster to track object location. However, Frisbee consumes a lot of energy since the nodes of the cluster are always awake. Xu et al. proposed PES [10] to track objects with a prediction mechanism. The idea of PES is to track objects with less frequency, and to use a prediction mechanism to predict the next location of moving objects. In this way, the sensor nodes do not need to be awake in order to track objects and energy is thus saved. If the prediction mechanism misses the object location, PES has recovery mechanisms to relocate the object. However, the energy expended to relocate objects is quite high. Thus, PES is not suitable for tracking objects that move in an irregular fashion. Some other schemes have also been proposed to efficiently track moving objects [15]. However, they only address the tracking issue, and don't touch on the location maintenance issue. From these studies, we can know the object tracking problem in sensor networks contains the two fundamental sub-issues mentioned above, and so far no research has proposed a total solution to solve both sub-issues simultaneously.

In this paper, we proposed a short-term Prediction-based Optimistic Object Tracking Scheme (POOT) to integrate gathering and maintaining of tracking information and to obtain better performance in terms of communication costs. The basic concept behind POOT is to minimize routing distance while gathering location data. Thus, the design of our location gathering method is considered the underlying data routing protocol. More specifically, we adopted a face routing method to maintain updating of locations, while object detection was based on a face structure network. When detecting an object, the active nodes are all in the face, so data communication can be reduced. In addition, our tracking strategy was based on an optimistic method. That is, POOT will predict the next object location based on the timing of when POOT is going to lose the object, and so activates the nodes of nearby faces for tracking. Since the number of active nodes is less than other related methods such as DOT, POOT spends less communication time on detecting the object, and thus the wireless sensor network consumes less energy. Furthermore, we present two schemes, a Time-efficient Object Recovery Scheme (TORS) and a Communication-efficient Object Recovery Scheme (CORS), to improve object recovery. The two schemes can be adopted according to different system requirements. We also derived cost models for POOT, and those of DOT as well. Finally, we conducted a set of experiments to compare performance in terms of communication. The results reveal that POOT consumes less energy in tracking objects, while our optimistic design provides acceptable tracking accuracy.

In summary, our contributions in this paper are as follows:

- We propose a short-term Prediction-based Optimistic Object Tracking Scheme (POOT) to integrate gathering and maintenance of object location and then obtain better performance in terms of communication costs.
- Object detection is based on an optimistic method in order to conserve energy. Hence, POOT can intelligently determine the best time for next detection of the object. In this way, sensor energy is saved.
- We created two schemes, a Time-efficient Object Recovery Scheme (TORS) and a Communication-efficient Object Recovery Scheme (CORS), to improve object recovery, in the event the object is lost.
- We derived cost models for both POOT and DOT with different object recovery schemes.
- We conducted comprehensive experiments to study performance. In these POOT performed better than other related object tracking schemes.

The rest of this paper is organized as follows. Section 2 briefly describes the system environment of an object-tracking sensor network. We present the proposed Prediction-based Optimistic Object Tracking Scheme in Section 3. We then present two object recovery schemes in Section 4. In Section 5, we derive cost models for POOT and DOT. We conduct the experiment in Section 6. Finally, we draw conclusions and present future work in Section 7.

2. Environment

Fig. 1 illustrates an object tracking scenario in a wireless sensor network. A wireless sensor network is deployed with a large number of stationary sensor nodes over the interested field. Each sensor has sensing, computing, and communication capabilities, and has prior knowledge of its location in the wireless sensor network [16,17]. With these capabilities, the sensor nodes can determine the object location through the localization techniques [18]

Transmitting the data (e.g., the object trajectory) between any two nodes in the wireless sensor network is based on a multi-hop routing protocol, and many existing methods have been proposed to solve the routing problem, such as GPRS [19]. Past studies have proposed methods [20,21] to efficiently transmit data in the sensor network, however, they only considered the discrete and independent data (e.g., light and pressure) and are not optimally suitable for continuous and related data (e.g., object locations). Huang et al. investigated this problem, and proposed a face-structure routing method [22] to efficiently store and access locations of a moving object.

The issue of tracking moving objects inside wireless sensor networks has been much studied in recent years [23]. In order to reduce the high energy consumption of non-prediction based schemes, prediction-based schemes have been designed to reduce the number of activated nodes and active time on monitoring the moving object based on the movement knowledge of the object [10,24]. Prediction-based schemes use prediction techniques to predict the location of the moving object in the near future, and then activate the nodes nearby the predicted location at a specific time. Hence, the number of activated nodes and monitoring time can be greatly reduced. However, if the prediction mechanism misses the object location, it becomes necessary to find the object again, which is a high-cost operation. Since the prediction mechanism has a great impact on tracking performance, the issue of how to predict the location of a moving object is of critical importance [25].
3. Prediction-based optimistic object tracking scheme

3.1. Face-structure network

A well-organized network structure can greatly assist in the requesting and reporting of sensor data in wireless sensor networks. More specifically, a network structure can save energy and balance the operating load for distributed nodes. Many related network structures have been proposed, such as tree structure [7], cluster structure [26,27], and face structure [22]. In this paper, we extended face structure [22] to the object tracking issue under consideration, because adoption of a face structure can help reduce the communication load. Fig. 2 is a comparison among the above-mentioned structures.

Fig. 3 is an example of object tracking in a face-structure sensor network. When a group of nodes detect the approach of a tank, these nodes will issue a wake-up message along the direction that the tank moves in. Based on the face structure, nodes that receive a wake-up message and are in the direction the tank is moving in (i.e., solid line nodes) would be activated to continuously track the tank in the upcoming periods. Those nodes that receive the wake-up message and that are not located in the tank-moving direction (i.e., the dashed line nodes) will stay asleep after receiving the wake-up message. In this manner, only a moderate number of nodes need to join the tracking task and the communication costs should be acceptable.

Construction of a face-structure network can be classified into two parts. The first part is to generate planarized graphs; the second part is to construct faces in the networks. The purpose of the first part is to avoid the scenarios where communications of two pairs of nodes is crossing. In the following we will describe these two parts in details.

Planarized graphs

Wireless networks can be modeled as an undirected graph \( G = \{ V, E \} \), where \( V \) is the set of nodes, and \( E \) is the set of edges which makes a pair of nodes can directly communicate with each other. For any two nodes that can directly communicate it is necessary to check whether another intermediate node exists such that the communication distance between the two nodes can be divided into two shorter communication distances through the intermediate node. Fig. 4 illustrates this scenario. Two nodes \( u \) and \( v \) can directly communicate with each other, and \( w \) is an intermediate node. We can see that \( \text{dist}(u, v) \) is shorter
than both \( \text{dist}(u, w) \) and \( \text{dist}(w, v) \). If such an intermediate node exists, then the edge of the two nodes (i.e., \((u, v)\)) will be eliminated from \( E \). Algorithm 1 shows the details involved in the creation of a planarized network topology slightly modified from the Relative Neighborhood Graph (RNG) Algorithm [19].

\[
\text{Algorithm 1: RNG Algorithm}
\]

\[
\begin{array}{l}
\text{Input: the set of sensor nodes } N; \\
\text{for all the } v \in N \text{ do} \\
\quad \text{for all the } u \in N \text{ do} \\
\quad \quad \text{/* } u \text{ and } v \text{ are two communicable nodes in } N, w \text{ is an intermediate node between } u \text{ and } v. \text{ */} \\
\quad \quad \text{if } w == v \text{ then} \\
\quad \quad \quad \text{continue;} \\
\quad \quad \quad \text{else if } \text{dist}(u, v) > \max(\text{dist}(u, w), \text{dist}(w, v)) \text{ then} \\
\quad \quad \quad \quad \text{eliminate edge}(u, v); \\
\quad \quad \quad \quad \text{break;} \\
\quad \quad \quad \text{end if} \\
\quad \quad \text{end if} \\
\quad \text{end for all} \\
\text{end forall}
\end{array}
\]

\textbf{Face construction}

Face construction in a sensor network can be achieved by two face identification methods. We use the example in Fig. 5 to illustrate face construction as follows. For a sensor node, the right-hand rule [19,22] is used to identify all nodes in the same face.

- \textit{Inner-face identification}: Each node transmits an IN-FACE message to its neighbors in a clockwise direction, in order to identify other nodes in the same face. We use node 1 as an example to illustrate inner-face identification. Node 1
transmits an IN-FACE message to its neighbors in a clockwise direction, i.e., node 2. Note that the clockwise direction can be determined by using the outer product of vectors. When node 3 selects the next node, nodes 10 and 4 both lie in a clockwise direction. In this situation where multiple nodes can be selected, the node that has maximal angle to the node 3 will be selected. This node can be discovered out by using the inner product of vectors. Thus, node 4 is selected as the next node. After node 1 has received the IN-FACE message, all nodes in the message belong to the same face as node 1.

- **Outer-face identification**: Outer-face identification is similar to inner-face identification. Again, we use node 1 as an example to illustrate outer-face identification. Node 1 transmits an OUT-FACE message to its neighbors in a clockwise direction, i.e., node 2. Similar to inner-face identification, when multiple nodes can be considered as the next node, the node which has the smallest angle to the transmitted node will be selected. For example, node 3 can select node 4 or 10 as the next node. Since node 10 has a smaller angle to node 3, node 3 selects node 10 as the next node.

In the above example, nodes 1, 2, 3, 4, 5, 6, 7, 8, and 9 are in the same face. Nodes 3, 10, 11, 12, and 4 form the second face, and nodes 4, 12, 13, 14, 15, 16, 17, 6, and 5 form the third face. Finally, nodes 1, 2, 3, 10, 11, 12, 13, 14, 15, 16, 17, 8, 7, 6, and 9 form the outer face.

### 3.2. Short-term prediction techniques

Many prediction mechanisms for object mobility have been researched and presented [10,24,25,28]. The prediction techniques are designed according to different application demands. For example, if the scale of the field of interest is large, regression-based predictions will be adopted. On the other hand, if the field of interest is small, some techniques such as Kalman filter, particle filter, and moving horizon estimation (MHE) will be used.

A simple and efficient regression-based prediction method is the linear mobility model, and it is suitable for sensor networks due to its low computational complexity. Assume the current position is \((x_i, y_i)\), then the predicted position after time \(\Delta t\) based on the linear mobility model can be expressed as follows:

\[
\begin{align*}
x_{i+1} &= x_i + \text{vel} \times \Delta t \times \cos(\theta), \\
y_{i+1} &= y_i + \text{vel} \times \Delta t \times \sin(\theta),
\end{align*}
\]

where \(\text{vel}\) is the velocity of the target, \(\Delta t\) is the period of time between \(t_i\) and \(t_{i+1}\), and \(\theta\) is the angle of the movement direction. The velocity \(\text{vel}\) and the angle of movement direction \(\theta\) can be obtained by using the historical movement pattern. The velocity \(\text{vel}\) for time instant \(t_i\) can be obtained as

\[
\text{vel} = \frac{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}{\Delta t}.
\]

The angle of movement direction \(\theta\) for time instant \(t_i\) can be obtained as

\[
\theta = \cos^{-1} \frac{|x_i - x_{i-1}|}{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}.
\]

The advantage of the model is that it is easy to implement, and the lightweight computation means it is suitable for poor-resource sensor devices. In addition, the performance is acceptable if prediction time is short.

Fig. 6 illustrates an example of estimation of velocity and angle. In the example, we assume the target position is \((0, 0)\) at time \(t_{i-1}\), and \((2, 2)\) at time \(t_i\). The velocity of the target used in the short-term prediction can be estimated as follows:

\[
\text{vel} = \frac{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}{\Delta t} = \frac{\sqrt{(2 - 0)^2 + (2 - 0)^2}}{1} = 2.8 \text{ m/s}.
\]
The angle $\theta$ can be estimated using trigonometry as follows:

$$\theta = \cos^{-1} \frac{|x_{t} - x_{t-1}|}{\sqrt{(x_{t} - x_{t-1})^2 + (y_{t} - y_{t-1})^2}}$$

$$= \cos^{-1} \frac{2}{\sqrt{(2)^2 + (2)^2}} = 44.7^\circ.$$  

After obtaining the two parameters $\text{vel}$ and $\theta$, the target position can be predicted by using the linear mobility model mentioned above.

One other prediction method used in small scale sensor fields is the Kalman filter and its variants \[29,30\]. The Kalman filter and its variants use observed measurements containing noise and other inaccuracies to generate values that tend to be close to true values. From the system design aspect, the Kalman filter is used to correct noise and generate better-quality locations. In addition, when time is updated, the Kalman filter can be used to predict the new location as well.

The estimation process of the Kalman filter can be classified into two groups: the time update equations used for prediction; and the measurement update equations used for location stability. Fig. 7 shows a relationship diagram of prediction and location stability. From the figure, we can see that the estimation of the Kalman filter is a process of feedback control. The filter predicts new locations at some time slots and then obtains feedback from measurements. The specific equations for time updates (from step $k - 1$ to step $k$) are described as follows:

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_{k-1}$$

$$P_k = AP_{k-1}A^T + Q$$

where the matrix $A$ is the state at the previous time step $k - 1$ to the state at the current step $k$, the matrix $B$ is the optional control input to the state $\hat{x}$, and $Q$ is the process noise covariance. The first equation above is used to predict the next
location (i.e., $\hat{x}_k$), and the second equation is used to predict the next error covariance (i.e., $P_k^{-}$). The specific equations for measurement updates (for step $k$) are described as follows:

$$K_k = P_{k-1}^{-}H^T(HP_{k-1}^{-}H^T + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k + K_k(z_k - H\hat{x}_k)$$

$$P_k = (I - K_kH)P_{k-1}^{-}$$

where the matrix $H$ is the state to the measurement $z_k$, and $R$ is the measurement error covariance. The first equation above is used to compute the Kalman gain (i.e., $K_k$). The second equation is used to update location (i.e., $\hat{x}_k$) with measurement $z_k$. The last equation is used to update the error covariance (i.e., $P_k$).

Notice that the prediction mechanism can be customized according to the application’s demands. If the system administrator has enough mobility knowledge for the tracking targets, the linear mobility model can be replaced by another suitable prediction model.

3.3. Target tracking mechanism

During tracking of the target, the sensor node that detects the targets and is closest to the target transmits a wake-up message to other nodes of the same face in order to track the target. In addition, this node needs to estimate the face that the target will visit next based on the target information obtained from the sensor nodes of the same face, such as velocity, movement direction, etc. If the target is going to leave the territory of the resident face, the sensor node that detects and is closest to the target of that face will transmit the wake-up message to nodes in the next face based on the prediction mechanism mentioned above. Thus, the sensor network can continuously track the target. The detailed steps of target tracking are shown in Algorithm 2.

### Algorithm 2: POOT Algorithm

```plaintext
/* Stage 1: initialization */
/* The source uses the flooding method to find the current location. */
targetDiscovery();

/* Stage 2: location prediction */
/* assume current location is $(x_c, y_c)$, predicted location $(x_p, y_p)$. */
$x_p = x_c + vel \times \Delta t \times \cos(\theta)$;
$y_p = y_c + vel \times \Delta t \times \sin(\theta)$;

/* Stage 3: active nodes in associate faces */
earest_node($x_c, y_c$) asks the indicator of faces face($x_c, y_c$) and face($x_p, y_p$) to activate nodes to track the target;

/* Stage 4: tracking target */
if nearest_node($x_p, y_p$) is found then
  nearest_node($x_c, y_c$) gives the tracking right to nearest_node($x_p, y_p$);
  nearest_node($x_p, y_p$) become the beacon node;
  if source can reach the beacon node then
    beacon node set Active_Node(0);
  end if
  goto Stage 2;
else
  object_recovery();
end if
```

4. Improvement of object recovery

Two object recovery schemes are proposed, a Time-efficient Object Recovery Scheme (TORS) and a Communication-efficient Object Recovery Scheme (CORS), in order to improve the number of nodes used in the flooding that recovery schemes require. TORS is designed to find a lost object with the least amount of elapsed time. CORS uses the least amount of communication to find the lost object by using the object’s direction of movement and distance.

4.1. Time-efficient object recovery scheme

The Time-efficient Object Recovery Scheme (TORS) wakes up all nodes whose distances to the lost location are less than the longest distance obtained by using the maximum moving speed. Let $R$ represent the circular area centered on the lost location and its radius is equal to the longest distance obtained by using the maximum moving speed. $R$ can be obtained as

$$R = \pi \times (v_{max} \times \Delta t)^2$$
where $v_{\text{max}}$ is the maximum moving speed and $\Delta t$ is the difference from the recovery time to the object lost time. Sensor nodes in area $R$ will be woken up to find the lost object. The consumed energy for TORS is proportional to the number of wake-up nodes $N_{\text{TORS}}$. If the sensor nodes are uniformly deployed, the number of wake-up nodes $N_{\text{TORS}}$ depends on the size of area $R$, and can be estimated by the following formula:

$$N_{\text{TORS}} = \frac{R}{\delta}$$

where $\delta$ is the average node density in the uniformly-deployed sensor network. In addition to time efficiency, another advantage of TORS is that the range of flooding of the recovery message is limited and can be evaluated by the above formulas.

Fig. 8 illustrates an example of TORS. In the figure, $s_i$ is the node that last detected the object. $s_i$ calculates the radius to determine the object-recovery region. Then, if the center of a face is within the circle formed by the above radius, $s_i$ sends object-recovery messages to those nodes of that face. In this example, nodes in the faces 7, 8, 9, 11, 12, 13, 16, 17, and 18 are woken up to find the lost object. Hence, TORS can find the lost object in a short amount of time, and the area of flooding area for sending of recovery messages can be restricted.

4.2. Communication-efficient object recovery scheme

The basic idea of the Communication-efficient Object Recovery Scheme (CORS) is to sequentially search for the lost object according to the probability of it being in certain faces; this probability is denoted as the priority value. We observe that the face where the lost object may reside has the following properties:

- The lost object possibly resides in nearby faces, instead of faces far away from the location in which it went missing.
- The angle between the face that the lost object current resides in and the location where it went missing should be small, because objects do not generally suddenly change to a different direction with a great angle.

Hence, from the above observations, the priority value (denoted as $pv_i$) of face $F_i$ is inversely proportional to $\text{dist}(\text{loc}_{\text{lost}}, \text{centroid}(F_i)) \times \theta_i$, where $\text{centroid}(F_i)$ stands for the center of face $F_i$ and $\theta_i$ means the angle between the center of face $F_i$ and the past trajectory of the lost location. Hence, priority value $pv_i$ can be designed as

$$pv_i = \frac{1}{\text{dist}(\text{loc}_{\text{lost}}, \text{centroid}(F_i)) \times \theta_i}$$

Algorithm 3 depicts the detailed steps of the CORS algorithm. The first step of CORS is to determine possible faces where the lost object could be residing, and these possible faces are within a circular area centered on the last known location with radius $v_{\text{max}} \times \Delta t$ (line 1). Then, the priority value of each face that satisfies the above condition is calculated as the formula mentioned in the last paragraph (lines 2–4). After computing the priority values of related faces, these faces are sorted and maintained in an array $\text{order\_list}$ (line 5). Finally, faces in the $\text{order\_list}$ are searched sequentially until the lost object is found (lines 6–10).

Fig. 9 illustrates an example of CORS. Assume five faces, $\{F_1, F_2, F_3, F_4, F_5\}$, are within the search range radius. The distances $\text{dist}(\text{loc}_{\text{lost}}, \text{centroid}(i))$ are $4.25, 7.43, 7.66, 8.60, 5.16$, respectively. The angles between the center of face $F_i$ and the past trajectory of the last known location are $19.22^\circ, 30.73^\circ, 59.15^\circ, 79.61^\circ, 104.6^\circ$, respectively. The priority values for these faces can be calculated as follows:
Algorithm 3: CORS Algorithm

/* Determine search range for search. */
1 radius = v_max × Δt;
/* Compute priority value pv_i for face i. */
2 foreach face i that satisfies dist(loc_lost, i) ≤ radius do
3  \( p_{v_i} = \frac{1}{\text{dist}(\text{loc}_{\text{lost}}, \text{centroid}(i)) \times \theta_i} \);
4 end foreach
/* Determine the order of the face. */
5 sort the above faces according to their priority values, and maintain the sorting result in order_list;
/* Find out the lost object. */
6 for (i = 0; i < |order_list|; i + +) do
7    if (the lost object is found in the i-th face of order_list) then
8        break;
9    end if
10 end for

Fig. 9. Example of CORS.

\[ p_{v_1} = \frac{1}{\theta_1 \times d_1} = 0.0122 \]
\[ p_{v_2} = \frac{1}{\theta_2 \times d_2} = 0.0044 \]
\[ p_{v_3} = \frac{1}{\theta_3 \times d_3} = 0.0022 \]
\[ p_{v_4} = \frac{1}{\theta_4 \times d_4} = 0.0019 \]
\[ p_{v_5} = \frac{1}{\theta_5 \times d_5} = 0.0015 \]

Next, sorting the priority values estimated above, we can obtain the order for searching these faces as \( F_1, F_2, F_3, F_5, F_4 \).

5. Cost analysis

In this section, we present a cost analysis for POOT and DOT [12] which is also an object tracking scheme for application in a face-structure sensor network. The metrics of evaluating object tracking schemes is based on the number of wake-up nodes necessary for tracking mobile targets. The parameters used in our cost models are listed in Fig. 10.

Let \( F \) be the face set in the sensor network. Since POOT is based on the prediction method, the probabilities of successful and lost tracking are taken into consideration in the total cost. Hence, the total cost is represented as follows:

\[ C_{\text{POOT}}^{\text{success}} \times (|F_{\text{current}}| + |F_{\text{next}}|) + (1 - P_{\text{POOT}}^{\text{success}}) \times N \]

where \(|F_{\text{current}}| + |F_{\text{next}}|\) is the number of nodes during the handoff in tracking when the short-term prediction successfully tracks the object, and \( N \) is the number of nodes in the sensor network. Notice that when the object is lost, POOT will wake up all nodes to locate the object. Thus, the number of wake-up nodes for finding the lost object is \( N \).

For the sake of simplicity, we assume that sizes of all faces are similar. The values of \(|F_{\text{current}}|\) and \(|F_{\text{next}}|\) can be estimated by the average number of nodes of a face (denoted as \( \text{AVG}(|F|) \)) [19]. The value of \( \text{AVG}(|F|) \) is equal to

\[ \text{AVG}(|F|) = \frac{e_1 + e_2 + \cdots + e_n}{\text{COUNT}(F)} \]
Fig. 10. Parameters and their meanings.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C^X$</td>
<td>The total cost of strategy $X$, where $X \in {\text{POOT, DOT}}$</td>
</tr>
<tr>
<td>$P^X_{\text{success}}$</td>
<td>The probability of successfully tracking object of strategy $X$.</td>
</tr>
<tr>
<td>$F$</td>
<td>The face set in the sensor network.</td>
</tr>
<tr>
<td>$</td>
<td>F_{\text{current}}</td>
</tr>
<tr>
<td>$</td>
<td>F_{\text{next}}</td>
</tr>
<tr>
<td>$AVG(</td>
<td>F</td>
</tr>
<tr>
<td>$COUNT(F)$</td>
<td>The number of faces in the sensor network.</td>
</tr>
<tr>
<td>$N$</td>
<td>The number of nodes in the sensor network.</td>
</tr>
<tr>
<td>$e_i$</td>
<td>The number of edges for face $i$, $i = 1, 2, \ldots, N$.</td>
</tr>
<tr>
<td>$E$</td>
<td>The number of face edges in the sensor network.</td>
</tr>
<tr>
<td>$\Delta t$</td>
<td>The time difference between two location updates.</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>The number of faces waked up by DOT in the next prediction.</td>
</tr>
<tr>
<td>$df_i$</td>
<td>The number of faces that contains node $i$.</td>
</tr>
<tr>
<td>$de_i$</td>
<td>The number of edges that connect to node $i$.</td>
</tr>
</tbody>
</table>

where $e_i$ stands for the number of edges for face $i$ ($i = 1, 2, \ldots, N$), and $COUNT(F)$ is the number of faces in the sensor network. Note that each edge is shared by two faces. Thus, we obtain the following formula for estimating the upper bound of total face edges

$$e_1 + e_2 + \cdots + e_N \leq 2 \times E$$

where $E$ is the number of total face edges. From the above two equations, $AVG(|F|)$ can be rewritten as

$$AVG(|F|) \leq \frac{2 \times E}{COUNT(F)}.$$  

Using the above formula, we can estimate the maximal value of $AVG(|F|)$. Notice that the above estimation of $AVG(|F|)$ is conservative. Thus, the cost models can be seen as the estimation of the upper bound of the total cost. In other words, the total cost estimated by the cost models is higher than in real cases.

In Section 4 we discussed ways to improve object recovery. When an improved object recovery (e.g., TORS) is adopted, the total cost can be reduced to

$$C^{\text{POOT}} = P^{\text{POOT}}_{\text{success}} \times (|F_{\text{current}}| + |F_{\text{next}}|) + (1 - P^{\text{POOT}}_{\text{success}}) \times (\pi \times \Delta t)^2.$$  

If CORS is adopted instead, the total cost can be derived in a similar manner.

The total cost of DOT can be obtained by using the similar models of POOT mentioned above, and can be represented as follows.

$$C^{\text{DOT}} = P^{\text{DOT}}_{\text{success}} \times (|F_{\text{current}}| + \alpha \times |F_{\text{next}}|) + (1 - P^{\text{DOT}}_{\text{success}}) \times N$$

where $|F_{\text{current}}| + \alpha \times |F_{\text{next}}|$ is the number of nodes during the handoff in tracking when DOT successfully tracks the object, and $\alpha$ is the number of faces woken up by DOT in the next prediction. We refer to [19] to estimate the value of $\alpha$ as follows.

$$\alpha = \frac{\sum_{i=1}^{N} df_i}{N}$$

where $df_i$ is the number of faces that contains node $i$. Let $de_i$ be the number of edges that connect to node $i$. Since the number of edges is greater than the number of faces for a sensor node, we can represent such relationship node $i$ as

$$de_i \geq df_i.$$  

On the other hand, the summation of the edges of each faces is equal to

$$\sum_{i=1}^{N} de_i = 2 \times E.$$
From the above two equations, we obtain the following formula

\[ 2 \times E \geq \sum_{i=1}^{N} d_{f_i}. \]

Applying the above formula to the equation of \( \alpha \), we can obtain the maximal value of \( \alpha \) as follows.

\[ \alpha \leq \frac{2 \times E}{N}. \]

By using the above cost models, once the characteristic of the sensor network (e.g., details of deployment) and the movement profile of objects are given, we can obtain the total cost of POOT and DOT. Recall that the cost models were derived in the conservative manner mentioned above. We will also compare the difference of the cost models and the simulation results in Section 6.5.

6. Performance evaluation

In this section, we describe the series of experiments we conducted to study the performance of the proposed method. We developed a trace-based simulation system that tracks the target in a wireless sensor network by using Java [31,32]. We deployed 225 sensor nodes in a space of 400 × 400 m\(^2\). The communication range of a sensor node is 50 m. Fig. 11 illustrates an example of a face-structure network based on the RNG algorithm in our simulation.

The settings of the simulation system are described as follows. The power consumption for transmitting a packet is 700 mW, and the power consumption for receiving a packet is 360 mW. The movement of the target is modeled by using the Random Way Point model. The velocity of the target was randomly set to [0, 30] m/s. We will compare the performance of our proposed method to DOT [12], because the underlying networks of the two methods are the same, i.e., the face-structure networks. Moreover, under the above experimental settings, the experiments simulated the tracking of objects in large-scale fields. Hence, we adopted the linear mobility as a short-term prediction mechanism. In the experiments, the target was moved around 1000 steps to ensure the movement data was representative.

6.1. Energy comparison between two methods

In the first experiment, we studied the comparison of the two methods, i.e., POOT and DOT, and the results are shown in Fig. 12. The horizontal axis shows the movement speed of between [5, 30] m/s, and the vertical axis represents average power consumption. From the figure, we can see that our proposed POOT method consumes less power consumption than DOT in a range of 23%–2%. This is because POOT is an optimistic method, i.e., a lesser number of faces are activated in tracking tasks. We will show a comparison of active faces and discuss accuracy in the next two experiments. This significant improvement in power consumption can greatly prolong network lifetime.
However, while increasing movement speed, the optimistic method may make more inappropriate decisions on waking up faces. Hence, recovering the lost target increases energy cost during target tracking, and the power consumption of POOT is close to that of DOT when the movement speed is 30 m/s. For real-world applications, movement speeds of most kinds of targets are less than 30 m/s (i.e., around 100 km/h). Therefore, POOT is suitable for most application demands.

6.2. Effect of number of wake-up faces

In this experiment, we further studied the effect of speed versus number of wake-up faces, and the results are shown in Fig. 13 in which we list the number of activated faces of the two methods and the reduction ratio of POOT to DOT. From the results, we can see that POOT activates much fewer faces than DOT. The difference comes from the design principle, i.e., POOT optimistically wakes up fewer faces and thinks the faces it has woken can efficiently track the target. Note that an active node needs to transmit and receive a number of packets. Hence, increasing one active node could increase power consumption by a great deal. In addition, the simulation can be treated as fair and valid, because the results of the figure are consistent with the results shown in Fig. 12.

6.3. Does optimistic design affect accuracy?

It might be thought that linear mobility is straightforward and that therefore a possible concern in this paper is that optimistic prediction could affect tracking accuracy. This experiment studied an interesting issue regarding whether the optimistic design of the proposed method would greatly reduce tracking accuracy. Fig. 14 shows a comparison of tracking accuracy between the two methods. As we know, too optimistic a design could incur frequent target losses, leading to the waste of large amounts of energy. However from the diagram we can see that the accuracy of POOT is still higher than 97.5% in all cases, thus, the design is not overly optimistic and tracking accuracy is not sacrificed. We think the accuracy achieved
is acceptable for most applications. The relationship between accuracy and energy conservation under different degrees of optimism may still give rise to concerns in some. We leave this issue for our future studies.

6.4. Effect of object recovery schemes

In this experiment, we compared object recovery schemes, including flooding schemes, TORS, and CORS, and the results are shown in Fig. 15. The flooding scheme was originally used for general object tracking schemes, and thus, we implemented this scheme for both POOT and DOT. In the figure, TORS and CORS performed better than the flooding scheme, as was our expectation. From the results, we can see that both TORS and CORS possess the advantage of restricting the sending of messages to find the lost object. In addition, CORS schedules the search order of possible faces according to the priority values mentioned in Section 4.2. Therefore, CORS can reduce the amount of communication even more than TORS can, that is, CORS consumes less energy than TORS as shown in the figure. Notice that the frequency of the object becoming lost increases with increasing velocity. Hence, more energy is consumed in order to find the lost object, and each curve sharply rises as velocity increases. Also, frequent loss of the object means the cost of object recovery forms a major portion of the total cost. Therefore, the performance superiority of POOT reduces with increasing velocity, as shown in Fig. 15.

Fig. 16 shows the tracking accuracy of object recovery schemes under various speeds. The vertical axis stands for the recovery accuracy of object recovery schemes. DOT with flooding has almost perfect recovery accuracy, because it behaves most conservatively, that is activating nodes as possible on object recovery (refer to its performance in Fig. 15). In contrast to DOT with flooding, TORS and CORS perform aggressively. The proposed object recovery schemes aim at activating few nodes where the lost object most possibly resides. Observing the experimental results, we clearly know that TORS and CORS both successfully achieve the lost object recovery because their recovery accuracy is over 99.97% for all cases in experiments. Comparing CORS and TORS, CORS can save more power consumption than TORS by reducing communication (refer to Fig. 15), but such behavior of CORS reduces the recovery accuracy as well (refer to Fig. 16). This is because CORS prioritizes the
possible faces where the lost object resides, and then searches for them sequentially in order to reduce communication loads. On the contrary, TORS simultaneously searches the limited number of faces for the lost object according to the movement distance estimated based on velocity and lost duration. From the above two figures, we can see that the experimental results fully reflect the performance characteristics and advantages of TORS and CORS.

6.5. Accuracy of analytic cost models

Our last experiment studied the accuracy of the analytic cost model presented in Section 5. We compared the analysis results against the simulation results based on the experimental settings. The implemented schemes included POOT with TORS, POOT with flooding scheme, and DOT with flooding scheme, and the results are shown in Fig. 17. From the figure, we can observe the trends of the analytic cost models are basically consistent with those of the simulation results. This indicates that our cost models can be applied to more situations.

Notice that difference between the analytic cost models and the simulation results is small when velocity is low, but becomes large when velocity is high. This can be explained as follows. The recovery of lost object events was modeled in a conservative manner, and thus, the analytical models provide the upper bounds of the consumed energy, i.e., the energy in the analysis models is higher than in the simulation results. As the frequency of lost object events increases under high velocity, the differences are accumulated and become large when velocity increases.

We further study the analytic cost models to show the error of the analysis results between analytic cost models and the simulation is very small, and can be negligible in most cases. We define the error of the analysis result for a tracking scheme as

\[
\text{Error} = \frac{|\text{simulation value} - \text{analysis value}|}{\text{analysis value}}.
\]
Fig. 18 illustrates the error of the analysis results. From the figure, we can see that the errors of most cases for the three schemes are much less than 10%, except the case at 25 (m/s) of TORS. Even in the above exception case, the error is smaller than 15% and such range is acceptable for most applications. Hence, the analytic cost models perform very similarly to the simulation results, and indeed can be applied to other cases of different velocity.

7. Conclusions and future work

In this paper, we proposed a short-term Prediction-based Optimistic Object Tracking (POOT) strategy to efficiently track moving objects. In order to reduce communication loads, POOT is designed based on a face-routing structure which optimizes the communication layers in the sensor network, and optimistically tracks objects based on a short-term prediction module. Moreover, two schemes that included TORS and CORS were created to improve performance of the object recovery mechanism. We derived cost models for POOT and conducted a set of experiments to reveal that communication costs are indeed reduced, as compared to related schemes. In addition, we also showed that an optimistic tracking design still provides acceptable tracking accuracy.

Future work includes two extensions. The first direction is to integrate our previous work [3] and POOT in a hierarchical object-tracking framework over sensor networks. In addition, we plan to develop a new prediction mechanism to replace current object movement prediction modules, such as [33]. One of our potential research issues in replacing the prediction mechanism is to develop a dynamic movement detection algorithm that can determine an optimal prediction module among pre-installed ones. The chosen prediction module would then work with the object-tracking framework in the next period to achieve superior performance. The second direction is to further investigate feasibility that performance analysis of the POOT mechanism may be compared with some new strategies. In this paper, we also developed the cost models for both object tracking schemes (POOT and DOT) under the face-structure sensor networks. Hence, we conducted experiments that fairly compare POOT to DOT based on the face-structure sensor networks. Developing a generic performance framework for object tracking schemes could offer more scenarios of performance study.

Acknowledgments

Authors thank Mr. Da-Chung Mao for sharing his experience about applying the Kalman filter to location prediction in the small-scale sensor networks. This work was partially supported by National Science Council of Taiwan (R.O.C.) under Grants NSC 97-2219-E-415-001, 98-2219-E-415-001, 99-2219-E-415-001, 99-2221-E-218-035, 100-2219-E-415-002 and 100-2221-E-218-053-MY2.

References


