Prediction-based protocol for mobile target tracking in wireless sensor networks

Liang Xue¹,², Zhixin Liu¹, and Xinping Guan¹,²

¹. Institute of Electrical Engineering, Yanshan University, Qinhuangdao 066004, P. R. China;
². School of Electronic, Information and Electrical Engineering, Shanghai Jiaotong University, Shanghai 200240, P. R. China

Abstract: Remote tracking for mobile targets is one of the most important applications in wireless sensor networks (WSNs). A target tracking protocol--exponential distributed predictive tracking (EDPT) is proposed. To reduce energy waste and response time, an improved predictive algorithm--exponential smoothing predictive algorithm (ESPA) is presented. With the aid of an additive proportion and differential (PD) controller, ESPA decreases the system predictive delay effectively. As a recovery mechanism, an optimal searching radius (OSR) algorithm is applied to calculate the optimal radius of the recovery zone. The simulation results validate that the proposed EDPT protocol performs better in terms of track failed ratio, energy waste ratio and enlarged sensing nodes ratio, respectively.

Keywords: wireless sensor network, target tracking protocol, predictive algorithm, recovery mechanism.

DOI: 10.3969/j.issn.1004-4132.2011.02.024

1. Introduction

Mobile target tracking in wireless sensor networks (WSNs) is an active research area. Generally, tracking algorithms in WSNs are classified into two classes: non-cluster-based and cluster-based, which depends on the network topology and structure. In non-cluster-based algorithms [1–3], there is not any centralized node for control. When a sensor detects an object, it records the information in local memory. On the other hand, for cluster-based algorithms [4–6], when a non-cluster-head node detects the target, it forwards the information to its cluster head. Then the cluster head propagates target information to the sink. Particularly, for saving energy, some predictive approaches, e.g., dual prediction-based reporting (DPR) [7], have been applied to target tracking. In [7], if predictive results are consistent with target actual state, transmission of sensor readings is avoided. As mentioned above, hierarchical network topology is benefit to both energy-saving and timescheduling for tracking in large-scale sensor networks.

In this paper, to enhance the tracking precision and save energy, a prediction-based protocol for mobile target tracking is proposed, which consists of four sub-systems as shown in Fig. 1. The proposed protocol has lower computation complexity and better performances in terms of energy consumption and tracking precision.

![Fig. 1 The components of EDPT tracking protocol](image)

2. System model and analysis

The task of tracking is carried out by sequentially active nodes around mobile target trajectory. Usually an unknown point can be located through at least three related distances. The nodes, which can locate the target and communicate with each other, are defined as a set called sensor-triplet.

2.1 Location model

It is well known that energy consumption is related to the distance for radio transmutation. The degree of attenuation can be modeled as the ratio of transmitting power and receiving power.

In our in-door test, the measured value of power, \( P(d_t) \), deviates from the theoretical value \( P(d_e) \). In Fig. 2, it is clear that the shorter the distance between target and inspecting node is, the closer the measurement value will be to \( P(d_t) \).
triplet. The weights \( \mathbf{\xi} \) are given in (3) and (4).

The target’s location can be approximated by minimizing the following objective function.

\[
\hat{\varphi}^* = \arg\min_{i=U,V,W} \sum (\|\hat{\varphi} - O_i\| - \hat{d}_i)^2
\]

where \( \hat{\varphi} \) is the predicted location of the target, \( i \) is the index of node in triplet, \( O_i \) (\( i = U, V, W \)) denotes the geographical coordinate of node \( i \).

Inspired by the above observation, let nodes in a sensor-triplet, \( \nabla \), be \( U, V \) and \( W \). Let the rough coordinates of \( U, V \) and \( W \) be \( (x_U, y_U), (x_V, y_V), (x_W, y_W) \) respectively, as shown in Fig. 3.

While the error is considered, the actual relative distance can be described as follows

\[
\begin{align*}
\hat{d}_U &= d_U + \omega_U \\
\hat{d}_V &= d_V + \omega_V \\
\hat{d}_W &= d_W + \omega_W
\end{align*}
\]

where \( \omega_U, \omega_V, \omega_W \) are independent and identically distributed variables, which can be regarded as the noise interferences to triplet-nodes’ measurements.

The target’s location can be approximated by minimizing the following objective function.

\[
\hat{\varphi}^* = \arg\min_{i=U,V,W} \sum (\|\hat{\varphi} - O_i\| - \hat{d}_i)^2
\]

where \( \hat{\varphi} \) is the predicted location of the target, \( i \) is the index of node in triplet, \( O_i \) (\( i = U, V, W \)) denotes the geographical coordinate of node \( i \).

\[
\delta = \sum_{i=U,V,W} \left( O_i \times \left( \frac{\xi_i}{\sum_{i=U,V,W} \xi_i} \right) \right)
\]

where \( \delta \) can be used to estimate the initial point for (2).

### 2.2 ESPA for target tracking

For a given target trajectory, we adopt historical trajectory points to deduce the target’s next possible locations. It is assumed that the sample period is a constant. Considering the \( x \)-coordinates at time \( t \), the required data are \( \hat{\varphi}_{x,t-4T}, \hat{\varphi}_{x,t-3T}, \hat{\varphi}_{x,t-2T}, \hat{\varphi}_{x,t-T}, \hat{\varphi}_{x,t} \) respectively. The next location, \( \hat{\varphi}_{x,t+T} \), can be approximated by

\[
\hat{\varphi}_{x,t+T} = \theta + \theta'T + \frac{1}{2} \theta''T^2
\]

where \( \theta, \theta', \) and \( \theta'' \) can be calculated with repeating operations using (6) and (7).

\[
\begin{align*}
S^{(1)}_1 &= \alpha_j \hat{\varphi}_{x,t-4T} + (1 - \alpha_j)S^{(1)}_{i-1} \\
S^{(2)}_i &= \alpha_j S^{(1)}_i + (1 - \alpha_j)S^{(2)}_{i-1} \\
S^{(3)}_i &= \alpha_j S^{(2)}_i + (1 - \alpha_j)S^{(3)}_{i-1}
\end{align*}
\]

For the initialization, \( i = j = 1, \alpha_1 = 0.1, S^{(1)}_0 = S^{(2)}_0 = S^{(3)}_0 = \hat{\varphi}_{x,t-4T} \). For the first round, the values of \( S^{(1)}_1, S^{(1)}_2, S^{(1)}_3 \) are obtained by (6), then increase \( i \) by one and \( \hat{\varphi}_{x,t-4T} \) is replaced by \( \hat{\varphi}_{x,t-3T} \). The rest can be done in the same manner. Until \( i \) reaches 5, we increase \( j \) by one and obtain \( \theta, \theta' \) and \( \theta'' \) by (7).

\[
\begin{align*}
\theta &= 3S^{(1)}_5 - 3S^{(2)}_5 + S^{(3)}_5 \\
\theta' &= \frac{\alpha_j[(6 - 5\alpha_j)S^{(1)}_5 - 2(5 - 4\alpha_j)S^{(2)}_5]}{2(1 - \alpha_j)^2} + \\
\theta'' &= \frac{[4(4 - 3\alpha_j)S^{(3)}_5]}{2(1 - \alpha_j)^2}
\end{align*}
\]

In order to reduce track failed scenarios caused by predictive error and intrinsic delay, we design a proportion and differential (PD) controller further as follows

\[
\begin{align*}
e_c &= \hat{\varphi}_{x,t+eT} - \hat{\varphi}_{t+eT} \\
\dot{e}_c &= (e_c - e_{c-1})/T \\
\alpha_{e+1} &= \alpha_e + (-k_1e_c + k_2\dot{e}_c)
\end{align*}
\]

where \( \alpha_j (j = 1) \) shown in (6) should be accordingly incremental with the increment of target’s speed.
3. Distributed tasks for member nodes

Step 1 For node \(v_k\) \((v_k \in S_{c_i})\), it checks that whether it received the wake-up messages from \(c_i\). If \(v_k\) is awakened by its cluster head, it will begin to sense the target. Otherwise, \(v_k\) keeps sleeping without state altering.

Step 2 \(v_k\) changes its hibernation mode to \(\bar{v}_k\) (\(\bar{v}_k\) denotes that \(v_k\) has been awakened). \(\bar{v}_k\) senses the target and judge that whether target, which is denoted as \(\lambda\), has been in its effective sensing range \(\ell\). Otherwise, \(\bar{v}_k\) will try again with the enlarged sensing range \(\mathcal{R}\).

Step 3 \(\bar{v}_k\) sends the detected energy level to \(c_i\), then \(\bar{v}_k \rightarrow v_k\).

3.2 Distributed tasks for cluster heads

This subsection features the distributed processing tasks for cluster heads.

Step 1’ If target \(\lambda\) is in triplet \(\nabla_{c_i}(\nabla_{c_i}\) denotes current triplet-nodes e.g., \(U, V, W\) all locate in \(\Phi_{S_{c_i}}\), \(c_i\) notifies its downstream cluster head \(c_k\) about the approaching target \(\lambda\).

Step 2’ Since \(c_k\) can also obtain target predictive location \(\bar{\varphi}\) from \(c_i, c_k\) checks whether there exists some nodes \(v (v \in S_{c_k})\). If such nodes \(v\) exist, \(c_k\) wakes up the node that satisfies the condition \(v = \arg\min d(v, \bar{\varphi})\).

Step 3’ Repeat Step 2’ until enough triplet-nodes are awakened. Otherwise, go back to Step 1’.

3.3 Two peculiar situations

For the area with sparse nodes, it is possible that there are not enough nodes for target detection, i.e. the number of nodes is too few to form triplet for on-going target. Therefore, an alternative approach is designed.

Step 1” Downstream cluster head \(c_k\) checks whether there are some member nodes \(v\), i.e. \(\ell \leq d(v, \bar{\varphi}) \leq \mathcal{R}\). \(c_k\) selects node \(v (v \in S_{c_k})\), \(\ell \leq d(v, \bar{\varphi}) \leq \mathcal{R}, \ell \leq v \leq \mathcal{R}, v = \arg\min d(v, \bar{\varphi})\) as the triplet-node for target sensing.

Step 2” If there are not enough nodes to form triplet after Step 2’–Step 1”, \(c_k\) polls all \(c_i\) \((1 \leq i \leq m)\) with condition \(\langle c_i, c_k \rangle \in E\) for cooperation. Cluster head \(c_i\) will repeat the triplet-nodes searching process i.e. Step 2’–Step 1”.

Step 3” If Step 2” is still failed, a recovery mechanism should be activated, which will be introduced in subsection 3.4.

3.4 Recovery mechanism

Optimal searching radius (OSR) is used to re-capture the lost target.

Assumed that the maximum speed \(p_{\text{max}}\) of a certain target, \(\lambda\) is known in advance. Actually, \(\lambda\) rarely moves at its maximum speed \(p_{\text{max}}\). Thus the minimum radius \(\mathcal{S}R\) that can cover the lost target is given by (11)–(13).

\[ \mathcal{S}R = \frac{(p_{\text{max}})^2 - (p_t)^2}{2a} \quad (11) \]

\[ p_t = \frac{\bar{\varphi}_{t+T} - \bar{\varphi}_{t}}{T} \quad (12) \]

\[ a \approx \frac{p_{\text{max}} - p_t}{T} \quad (13) \]

Cluster head \(c_k\) sends wake-up messages to all heads \(c_j\) that satisfy \(d(c_j, c_k) \leq \mathcal{S}R\). Then each qualified \(c_j\) wakes
up their member nodes \(v (v \in S_{c_k})\) to recapture the lost target.

The whole state flow chart is shown in Fig. 4.

![Fig. 4 State flow chart](image)

**Remark 3** With steps stated above, the state flow will finally terminate.

According to Fig. 4, any tracking state, which reaches Step 1 in the state flow chart, will eventually reach OSR or terminate by itself. To demonstrate this, we illustrate the philosophy by the state transfer diagram as shown in Fig. 5.

![Fig. 5 State transfer diagram](image)

If a specific node \(v\) changes to \(\bar{v}\) with probability \(p_1(t) > 0\), Step 1 will definitely transfer to Step 2. Otherwise, node \(v\) will keep sleeping. On the other hand, since \(p_2(t) = p_3(t) = p_{1'}(t) = 1\), the state flow migrates to Step 2′ directly. Furthermore, the transfer probability, \(p_2(t)\), which usually depends on some elements, such as node density \(\rho\), \(\|\bar{v} - \bar{v}\|\), \(\|\bar{v} - c_k\|\), can not be achieved in an explicit manner. If distances from eligible nodes to \(\bar{v}\) are all less than \(\ell\) and these eligible nodes belong to the same downstream cluster head \(c_k\), the state flow will terminate itself. Similar to \(p_2(t)\), \(p_{1'}(t)\) just enlarges triplet searching range from \(\ell\) to \(\bar{\ell}\). Both searching processes are the same in essence. Therefore, Step 2′ will be another possible point for flow termination. OSR, as an efficient mechanism, can recover the state flow graph from Step 1. At all events, even Step 2′ and Step 2″ are terminated or the target is beyond of tracking for some reasons, the state flow graph can also be recycled.

### 4. Performance evaluation

The simulation scenario is set that a target moves randomly within a 2-dimensional sensing region. The system parameters for network initialization are shown in Table 1. Each cluster head is located at the center of a grid whose monitoring area is 20 m × 20 m.

<table>
<thead>
<tr>
<th>System parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network area/m²</td>
<td>60 × 60</td>
</tr>
<tr>
<td>Node placement</td>
<td>Random</td>
</tr>
<tr>
<td>Cluster head placement</td>
<td>Grid</td>
</tr>
<tr>
<td>Target moving speed/(m/s)</td>
<td>(\ell : \mathbb{R} = 10 : 15)</td>
</tr>
<tr>
<td>Additive noise</td>
<td>(\mu = 0, \sigma = 0.1)</td>
</tr>
<tr>
<td>Time interval T/s</td>
<td>1</td>
</tr>
<tr>
<td>Energy model</td>
<td>(E_{\text{normal}} : E_{\text{enlarged}} = l^2 : \mathbb{R}^2 = 1 : 2.25)</td>
</tr>
<tr>
<td>Moving pattern</td>
<td>Fold line</td>
</tr>
</tbody>
</table>

#### 4.1 The evaluation metrics

(i) Tracking failed ratio \(R_f\)

\[
R_f = \frac{N_f}{N_t} \times 100 \% \tag{14}
\]

where \(N_f\) denotes the number of failed points for tracking, \(N_t\) is the total number of the considered points.

(ii) Energy waste ratio \(R_{ew}\)

\(N_t\) is finite when tracking within a given period. When triplet-nodes simultaneously sense the target using normal range \(\ell\), energy consumption amounts to 3 units, e.g., if \(N_t = 200\), the ideal energy consumption is 600 units. We define the energy waste ratio as follows

\[
R_{ew} = \frac{(E_a - E_i)}{E_i} \times 100 \% \tag{15}
\]

where \(E_a\) represents the actual energy consumption and \(E_i\) represents ideal energy consumption, which is 3 times of \(N_t\).

(iii) Enlarged sensing nodes ratio \(R_e\)

If there is no triplet-node detects the presence of targets, the failed node tries again with enlarged sensing range \(\bar{\ell}\). \(R_e\) is defined as

\[
R_e = \frac{N_{en}}{N_{tn}} \times 100 \% \tag{16}
\]

\(N_{en}\) is the number of times enlarged sensing range triggered, and \(N_{tn}\) denotes the total number of nodes awakened.

#### 4.2 Simulation results

In the fold line tracking scenario (as shown in Fig. 6), \(R_f\) of two protocols keep increasing with the increment of target’s speed, which is revealed \(R_f\) in Fig. 7. From Fig. 8, one can get that \(R_{ew}\) increases slowly and a better performance is gained compared with DPT. \(R_e\) is approximatively proportional to the increment of moving speed as...
shown in Fig. 9. It is obvious that ESPA is difficult to achieve high precision for the fast moving target. One solution is to reduce the sample period and increase the sample points. Thus, triplet-nodes will continuously resort to triggering enlarged sensing range or communicating with surrounding cluster heads for cooperation. From simulation results, one can get that the EDPT protocol has better performances than DPT with appropriate parameters.

true, since to the same trajectory, the faster moving target is more difficult to trace.

Fig. 11 shows the comparison of energy consumption when the moving speed varies from 1 m/s to 20 m/s. Due to enlarged sensing range triggered and track failed scenarios, actual energy consumption is usually more than ideal energy consumption. Although the energy waste ratios of both protocols increase gradually, EDPT performs a little better.
5. Conclusions

Compared with many filtering approaches, the EDPT protocol is suitable to track target with lower computation complexity. Accordingly, it can also estimate the target position without any prior noise state matrix or target state matrix, which are difficult to achieve. To recapture a lost target, an OSR mechanism is proposed. The EDPT protocol can serve as a tracking protocol with relatively higher precision and lower computation complexity.

References


Biographies

**Liang Xue** was born in 1982. He is now a Ph.D. candidate in Yanshan University. His research interests include localization and target tracking in wireless sensor networks.
E-mail: lxue@ysu.edu.cn

**Zhixin Liu** was born in 1976. He is now an associate professor. His research interests include network congestion control, network rate control for wireless sensor networks
E-mail: lzxauto@ysu.edu.cn

**Xinping Guan** was born in 1963. He is a professor with Institute of Electrical Engineering of Yanshan University. His research interests include wireless sensor networks, coordination of multi-agent systems.
E-mail: xpguan@ysu.edu.cn