Management of Target-Tracking Sensor Networks

Khaled Hadi
Department of Computer Engineering,
College of Engineering and Petroleum
Kuwait University,
P.O. Box 5969, 13060,
Safat, Kuwait
Fax: 965-2481-7451
E-mail: khaled.hadi@ku.edu.kw

C.M. Krishna
Department of Electrical and Computer Engineering,
University of Massachusetts,
Amherst, MA 01003
Fax: 001-413-545-1993
E-mail: krishna@ecs.umass.edu

Abstract: Binary proximity sensors are an inexpensive approach to target tracking in cyber-physical and other applications. In this paper, we use analysis and simulation to study tradeoffs in the management of such networks. The use of simple averaging is compared against a more complex convex hull calculation for position estimation. We model the impact of sensor density on the accuracy of the position estimate and present performance results associated with the choice of sampling period and sensor wakeup area. We provide an iterative filtering approach to protect against false positives. We also model the effect of a non-uniform sensor scatter.

Keywords: Target Tracking, Binary Sensing, Sensor Networks, Probabilistic Waking.


Biographical notes: Khaled Hadi received his Ph.D. from the University of Massachusetts, Amherst, in 2009, in Electrical and Computer Engineering and his MS Degree from the School of Information and Computer Sciences, University of California, Irvine, in 2004. He received the BS in Computer Engineering in 1999. He is currently an assistant professor of computer engineering at Kuwait University. His research interests include sensor networks, computer networks, distributed systems, and operating systems.

C.M. Krishna received his Ph.D. from the University of Michigan in 1984, his M.S. from Rensselaer Polytechnic Institute in 1980, and his B.Tech. from the Indian Institute of Technology, Delhi, in 1979. Since 1984, he has been on the faculty of the Department of Electrical and Computer Engineering at the University of Massachusetts, Amherst. His current research interests include real-time systems, fault-tolerance, sensor and computer networks, and performance modelling. He has coauthored textbooks in the areas of real-time systems and fault-tolerant computing. He is a Fellow of the IEEE.

1 Introduction

Advances in micro-electro-mechanical systems (MEMS) technology [3] have made possible low power, small size, and low cost wireless sensor networks (WSNs). WSNs need to gather information from the physical world and fuse it to achieve application goals (9; 15; 18; 21). In this paper, we set out to study the performance of very simple approaches to target tracking using binary proximity sensors, and the tradeoffs relevant to such approaches. We model the performance of a target location approach that simply averages the positions of the detecting sensor nodes, showing that just a handful of detecting nodes is sufficient to sharply reduce the expected position estimate error. We show that a more sophisticated approach, based on the finding the
centroid of the convex hull of the detecting nodes does no better. We provide an analytical model to determine how to identify the search area over which the sensor field should look for the target. This model allows one to manage the sensor resources effectively. We also present an algorithm to filter out outlier messages and thereby obtain some protection against faulty sensor nodes.

The rest of the paper is organized as follows: In Section 2, we discuss related prior work. In Section 3, we present some technical background, including the algorithm that we will use as a vehicle to study basic tradeoffs in target tracking with binary proximity sensors. It is important to point out here that most of these tradeoffs are not limited to this algorithm, but rather, are general to the target-tracking problem. In Section 4, we model the impact of node density on the target location estimation error. In Section 5, we explore how best to pick an appropriate search area for the target, based on general assumptions about its mobility. Section 6 discusses how to filter out noise in this algorithm. Such noise can arise from poorly calibrated or otherwise malfunctioning sensors. Section 7 concludes the paper.

2 Related Work and Technical Background

2.1 Tracking Schemes

Object tracking is an active research problem in sensor networks and has been addressed in previous work, e.g., (6; 20; 10; 12; 13; 16; 27; 23). Most of these studies focus on achieving high track quality without an excessive expenditure of energy. Energy savings are obtained through a variety of waking strategies for sensor nodes in order to track a moving target. In (13), there is only one node, called the leader node, which is awake at any given time, while the rest of the network is asleep. The leader node applies a sophisticated algorithm to estimate the target position. The leader then passes this updated belief to a node close to this estimated position. This node then becomes the new leader, and the original leader goes back to sleep. In (20), the authors show that combining a Selective Activation framework with prediction (SA), where a small group of nodes in a circular monitoring region is in tracking mode, and Duty Cycle Activation (DA), where the entire network is turned on and off at the same time, is effective in terms of tracking quality and energy efficiency. (27) proposes a dynamic convoy tree for data collection and fusion for target tracking to achieve energy savings. A convoy tree is a moving tree of nodes that reconfigure in an attempt to track a target. Collected and fused data are collected at the root of the tree. The authors provide two schemes to track the target: 1) Conservative Scheme, where the tree expands omnidirectionally with a given threshold that depends on target speed and current radius of the monitoring region, and 2) Prediction Scheme, where the target expands along the predicted path of the target. In this scheme, new nodes are added in the tree and some nodes are pruned. The results show that the conservative scheme achieves better tracking quality than the prediction scheme, at the price of greater energy consumption. In (6), the authors propose a proactive waking algorithm, which involves adaptive wake-period lengthening when a target is sensed nearby. This approach can be built atop any sleep-awake algorithm (e.g., Geographic Adaptive Fidelity (GAF)) or Probing Environment with Adaptive Sleeping (PEAS) (5)). The algorithm has multiple wake states of sensor nodes layered around the current target position. Results show that using the proactive waking algorithm in conjunction with sleep-awake algorithm improves track quality, but has some energy cost due to extending the waking period.

2.2 Mobility Models

There are several mobility models used to study the performance of target tracking networks. The simplest of all these models has the target move along a straight track at a constant speed (6; 13). (27) makes use of a Random Walk model, where at the end of each given time interval, the target changes its speed and direction according to some probability distribution. Where there are roads that the target has to follow, one can use the Pathway Mobility Model (13), where the target tracker makes use of the fact that the target is restricted to traveling on given roads or paths. Finally, (20) uses a sinusoidal trajectory as a mobility model to exploit the oscillatory behavior of the sinusoidal function.

2.3 Position Estimation Algorithms

There are different ways for a sensor network to estimate the location of a target. One simple approach is to average the position of nodes sensing the target (20). (17) estimates a weighting scheme for the sensor position that exploits the fact that if the sensor lies near the path of the object, the detection period will be longer: more details on this will be provided later in this paper. Another, more sophisticated, approach, uses cooperative signal processing and Bayes’s Law (13). In particular, Bayes’s law is used to quantify the probability of the target position, based on its recent history.

2.4 Track Quality

One rather obvious metric to measure the quality of a track is to measure the average error between the actual position of the track and the estimated one (20). Another metric uses the fraction of successful sensing points, where the target is sensed once every $T$ seconds (27). Another metric uses path exposure to express the quality of the tracking. Path exposure is the integral of the sensing intensity function over the track path, where the sensing intensity is inversely proportional to the Euclidian distance between the sensor and the target positions (6).
3 Technical Background: Basic Tracking Algorithm

The tradeoffs that we study in this paper are mostly general to tracking algorithms. However, we do require a concrete algorithm to use as a framework within which to carry out simulation studies. In this section, we describe such an algorithm.

The following are the underlying assumptions for this algorithm:

- The sensor nodes are identical and distributed uniformly and randomly on a given region. The sensors are denser within the border area than internal to the region (see below).
- The sensor nodes are divided into border nodes, located within a given distance of the border, and interior nodes. The border nodes, between them, keep the border area under surveillance all the time. Until an intruder is discovered, only their sensor part is on: the radio communications of the border nodes are off. The border area is more densely populated with nodes to ensure that it is continuously monitored over the designated period of operation without running out of energy. The radio communications of interior nodes use a low-energy paging channel (7; 22; 24). In this setup, a very-low-power radio (see (7)) is used to monitor the channel all the time. The monitoring circuit is responsible for waking up the node when appropriate. Other than this radio, the sensor devices and other elements of these interior nodes are turned off except when needed.
- Targets always enter the region through the border region: no targets are spontaneously created within the region under surveillance. In this paper, we assume no more than a single target at any time. Extending this paper to cover multiple simultaneous targets is the topic of future research.
- A single sink (base station) is also assumed in our implementation. This is where the sensor system reports the tracking data. The base station is computationally more powerful and we assume that it is not limited in its functioning by energy constraints (e.g., it may be connected to a power outlet).

The tracking algorithm is summarized in the flowchart depicted in Figure 1. It starts when a border node senses a target. It then switches on its radio communication channel and sends data information to interior nodes, which forward this message to the base station. In our implementation, we use geographic routing (14), but any other routing mechanism can be substituted. For data messages initiated from border nodes only and sent directly to the base station, the base station collects these messages and computes the predicted location of the next target point using linear prediction, i.e. a linear extrapolation of the last two sensing points. The base station then sends a new data message to a sensor closest to the predicted location of the target at the next sample point. We call this sensor a leader node. The leader node then wakes up sensors that are within $R$ meters away from it. $R$ is a wakeup range parameter that can take one of several allowed values. The system starts with a small value of $R$ and increases it as necessary to locate the target. The final $R$ value usually involves waking up all sensor nodes in the field (although the user may opt to set the algorithm to give up the search earlier than that). The waking sensors then switch on their sensor devices, and try to detect the target. Every sensor that detects the target sends a report to its leader node. If the leader node receives reports from sensors, it computes the next prediction location based on linear extrapolation of the previous and current sensing points and sends a data message to the sensors within the new prediction location. It also sends a report to the base station to update the tracking position. If the target is not detected after a given time for the widest possible search area, however, the search is abandoned.

It is worth emphasizing here that the leader node is appointed for the sole purpose of obtaining one particular location fix on the target; the basis for such an appointment is that it is the closest node to the predicted position of the target. Once that particular target fix has been obtained and the result reported (or the search is abandoned), the leader node relinquishes its leadership role.

As mentioned previously, we assume a wakeup mechanism to alert sensors within the sensing area to wake up (see (7) for one such mechanism). Sensors
Table 1  Simulation Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of nodes</td>
<td>2601, 3601</td>
</tr>
<tr>
<td>Border Width</td>
<td>20 m</td>
</tr>
<tr>
<td>Communication Range</td>
<td>80 m</td>
</tr>
<tr>
<td>Sensing Range</td>
<td>30 m</td>
</tr>
<tr>
<td>Power Consumption in Sensing</td>
<td>10 mW</td>
</tr>
<tr>
<td>Sensor Duty Cycle</td>
<td>10%</td>
</tr>
<tr>
<td>Power Consumption in Sleep Mode</td>
<td>1 µW</td>
</tr>
<tr>
<td>Transmission Rate</td>
<td>250 kbps</td>
</tr>
<tr>
<td>Data packet size</td>
<td>64 bytes</td>
</tr>
<tr>
<td>Wakeup packet size</td>
<td>16 bytes</td>
</tr>
<tr>
<td>ACK packet size</td>
<td>32 bytes</td>
</tr>
<tr>
<td>Target speeds</td>
<td>5 m/s; [2-8] m/s</td>
</tr>
<tr>
<td>$\epsilon_{amp}$</td>
<td>50 nJ/bit</td>
</tr>
<tr>
<td>$\mu$</td>
<td>100 pJ/bit/m²</td>
</tr>
</tbody>
</table>

(Other than in the border area) are therefore woken up as required; they do not have regular sleep-wake cycles.

The system attempts to obtain a location fix on the target every $t_s$ seconds, where $t_s$ is called the sampling time step (or just the time step).

We will use this algorithm as a framework for our simulation studies of the key tracking tradeoffs. Unless otherwise stated, we will assume the parameter values listed in Table 1.

We have used a random walk model (27) to describe the target mobility. The target follows a piecewise linear track, changing its speed and direction every $T$ seconds. The change in angle is a random variable, uniformly distributed over a given range.

Included in our simulation experiments are results pertaining to energy consumption. We have adopted the energy model presented in (8). When transmitting, the radio expends energy according to the following expression:

$$E_{TX}(d) = E_{elec} + \epsilon_{amp}d^2$$

(1)

and when receiving, it expends

$$E_{RX} = E_{elec}$$

(2)

where $E_{elec}$ is the energy required to run the radio circuitry, $\epsilon_{amp}$ is the energy consumed by the transmitter amplifier, and $d$ is the transmission range.

4 Position Estimation

In this section, we study the problem of estimating target position. We focus on simple ways of doing so.

4.1 Unweighted Averaging with Uniformly Distributed Sensors

Perhaps the simplest approach to estimating a target position is to just average the position of the nodes that detect the target. If the detecting nodes are sufficiently numerous and are scattered reasonably uniformly over the sensing area, we can expect that the estimate will be quite accurate. In this section, we model the accuracy of such a simple estimation approach; in particular, we study the impact of the number of detecting nodes on the error in the target position estimate.

Intuitively, the greater the number of detecting nodes, the more accurate will be the position estimate. Here, we quantify this intuition.

The target is at the center of a sensing circle whose radius is the sensing radius of each sensor, i.e., any awake sensor within that circle can detect the target. For convenience, define the origin of the coordinate system as the target position. We calculate the probability distribution function (PDF) of the $x$-coordinate of the one sensor, randomly placed within the sensing circle. We start by obtaining the PDF of the $x$-coordinate; the derivation for the $y$-coordinate is similar due to symmetry. The first step is to calculate the area, $W$, of the shaded region shown in Figure 2.

$$W = 2 \times (\text{Area of Sector} - \text{Area of Triangle})$$

$$= 2 \left( \frac{\theta}{2} S^2 - \frac{1}{2} \alpha S \sin(\theta) \right)$$

$$= S^2 \cos^{-1}\left(\frac{\alpha}{S}\right) - \alpha S \sin\left(\cos^{-1}\left(\frac{\alpha}{S}\right)\right)$$

(3)

The probability distribution function for the $x$-coordinate of the sensor, $F_X(\alpha)$, can now be derived as follows:

$$F_X(\alpha) = 1 - \text{Prob}(X > \alpha)$$

$$= 1 - \frac{W}{\text{Area of Circle}}$$

$$= 1 - \frac{S^2 \cos^{-1}\left(\frac{\alpha}{S}\right) - \alpha S \sin\left(\cos^{-1}\left(\frac{\alpha}{S}\right)\right)}{\pi S^2},$$

(4)

for $0 \leq \alpha \leq S$

(5)

By symmetry, we have

$$F_X(\alpha) = 1 - F_X(-\alpha), \quad -S \leq \alpha \leq 0$$

(6)

The probability density function (pdf) of the $x$-coordinate can be obtained either by differentiating this distribution function, or more directly, by observing that the length of the vertical chord in Figure 2 is given by
As a result, we can obtain the pdf of the average of the positions of the detecting sensors. The density of the x-coordinate of each of these sensors is obviously the same as in (7); furthermore, the sensor positions are assumed to be independent of one another. As a result, we can obtain the pdf of the average of the sensor positions by convolution. In particular, the pdf of the sum of the x-coordinates is

\[ f_{x_1 + \ldots + x_n}(\beta) = \sum_{\alpha=\min(\beta+S, n-1)S}^{\max(\beta-S, -(n-1)S)} f_{x_1}(\alpha) \cdot f_{x_2 + \ldots + x_n}(\beta - \alpha) d\alpha \quad 0 \leq \alpha \leq S \]  

The density function of the average x-coordinate can now be written as:

\[ f_{\bar{x}}(\beta) = n f_{x_1 + \ldots + x_n}(n\beta) \]  

An identical argument applies to the y-coordinate of the average.

Figure 3 shows the density functions for a given number of sensors detecting the target. There is a marked improvement in the quality of the estimate for two, as opposed to just one, detecting node. Further gains in accuracy are more limited. To quantify this, the variance of the estimate is plotted in Figure 4.

The mean position error as a function of the number of detecting nodes is shown in Figure 5. Our results indicate that as long as four or more nodes can detect a target, its position estimate will be quite accurate; further marginal improvements will be small. We do not therefore require a very large number of sensors to be awake within the sensing circle of the target.

In some instances, the sensor sensitivity may be a controllable parameter. Then, the sensing range is tunable within certain limits. Let us now consider the impact of sensing range on MPE.

Let \( \lambda \) be the average node density per unit area. If the total number of nodes is large, the distribution of nodes in any given area can be treated, to a good approximation, as a Poisson random variable with parameter \( \lambda \). This leads to the following expression for the mean position error as a function of the average node density:

\[ MPED(\lambda, S) \approx \sum_{i=1}^{\infty} e^{-\lambda\pi S^2} \frac{(\lambda\pi S^2)^i}{i!} MPE(i, S) \]  

In practice, we can obviously truncate the sum quite quickly.

Figure 6 shows the impact of the sensing range, \( S \), on the mean estimation error for a number of values of nodes. However, this comes at a cost: the variance of the median estimate is somewhat greater than that of the mean, as shown in Figure 4.
sensor density. As $S$ increases, the error at first drops a little owing to the increased number of sensors that can pick up the target. Beyond a certain point, however, the mean error increases since the increased number of detecting nodes can no longer compensate for the increased error due to many detecting sensors being far away from the target.

4.2 Unweighted Averaging with Non-Uniformly Distributed Sensors

Rather than wake up every node within the wakeup circle with the same probability, we can wake up more nodes closer to the anticipated location of the target, while keeping the rest of the area covered less densely. The former allows the target to be located more precisely in the likely event that the prediction error is limited; the latter ensures a lower target miss probability if the predicted point is distant from the actual target location. Perhaps the simplest non-uniform strategy is to have the wakeup probability decline linearly as we move away from the predicted target location;

$$p(r) = \left(1 - \frac{r}{R}\right) q$$

where $q$ is a control parameter. Denoting the average node density by $\rho$, the expected number of waking nodes is given by

$$N(R, \rho, q) = \int_0^R 2\pi r p(r) \rho dr = \frac{1}{3} \pi R^2 \rho q$$

Figure 7 compares the behavior of the Non-Uniform (NU) against that of the Uniform (U) model we presented previously. To keep the comparison fair, the value of $q$ is set so that the expected number of waking nodes in the U and NU cases are the same.

For the same value of $R_1$, for example, the probability of sensing the target is roughly the same: this depends on $R_1$ and not on the node density (so long as there is at least one node in the sensing area of the target). Similarly, since the expected number of waking nodes is the same, the energy consumed is about the same. However, there is a significant effect on the accuracy with which the position error is estimated, for larger values of $R_1$. The NU approach allows an inner circle to be densely populated by awakened nodes: this is compensated for by making the outer area more sparsely covered. When $R_1$ is large enough, the inner, denser, circle is large enough to capture most target instances and the average position error is reduced. For a similar reason, the advantage of the NU over the U case is increased when the prediction accuracy is greater; indeed, the purpose of NU is to squeeze additional performance out of accurate prediction. Figure 8 shows the estimated position accuracy as a function of the variance in the target mobility model. The lower the variance the greater the probability that the target will be in the more densely populated area near the predicted position, and the better will be the NU performance. For a purely random track, NU is not recommended. An insufficient awakened node density is not the only reason that NU can behave poorly if there is substantial error in predicting the target position. Another is a biasing caused by the non-

Figure 6  Impact of Sensing Range on Mean Position Estimate

Figure 7  Impact of node density on position errors as a function of $R_1$

Figure 8  Impact of track uncertainty on N vs NU performance
uniform positioning of the awakened nodes. Figure 9 illustrates this. Because the prediction error in this case is considerable, there are more awakened nodes on one side of the actual target position than in others. Therefore, any averaging of the positions of all those nodes which detect the target will have an inherent bias. One can correct for this bias by weighting each node position by the inverse of the awakened node density in that location and then averaging over these weighted positions.

We can model the bias effect as follows. Define a coordinate system with the predicted position of the target at the origin and the actual position at \((d,0)\). The target will be sensed by any node within a circle of radius \(S\), centred at \((d,0)\). The bias will be entirely along the \(x\)-axis, because of the way we have defined the axes of our coordinate system and due to symmetry in the density function. The expected estimated position will be at the centroid of the sensing circle.

This centroid can be computed as follows. Consider a vertical chord of the sensing circle around \((d,0)\), whose \(x\)-coordinate is \(a\) (illustrated by the heavy line in Figure 10). The length of this chord is \(L(a) = 2\sqrt{S^2 - a^2}\). The “mass” of this chord is proportional to

\[
m(a) = 2 \int_0^{L(a)} p(\sqrt{a^2 + y^2}) dy
\]

where, as before, \(p(r)\) is the density of awakened sensor nodes \(r\) away from the predicted target point. The centroid of this chord is obviously at \((a,0)\). Hence, the

\[
x_c = \frac{\int_{d-S}^{d+S} am(a) da}{\int_{d-S}^{d+S} m(a) da}
\]

The error caused by the bias effect is then given by

\[
\text{Bias}(d, S, r) = d - x_c
\]

In Figure 11, we plot the bias as a percentage of, \(d\), the actual distance of the target from the predicted point. The effects of the bias are most pronounced when the wakeup area is small and when the target is closer to the edge of this area. When the wakeup area is small, the density of awake nodes varies more rapidly; when the target is closer to the edge of the area, edge effects (the fact that all the detecting nodes have to be within the wakeup area) take over.

The impact of inaccurate position estimation (and the bias effect) on the NU approach is quantified in Figure 12. The leader node is the one closest to the predicted target position. If this distance is small, the increased density of nodes around the actual target position renders NU more accurate than U; as this distance increases, however, NU behaves more poorly.

We should point out that the problem of bias also occurs in the uniform node waking case, near the perimeter of the wake-region. Since the only nodes with sensors switched on are within this wake-region, there is effectively a non-uniform distribution near the perimeter, with the density falling abruptly to zero outside this perimeter. The modeling approach in this section can be used to capture that bias effect as well.

### 4.3 Convex Hull Approach

The convex hull of a set of points is the smallest convex area containing all these points. We can calculate the convex hull of the detecting sensor nodes and estimate the target position as its centroid. Experiments were
S = 30, R = 120; target speed = 6 m/sec; angle ∈ [−30°, 30°]

Figure 12  Position error under U and NU approaches

Note: NU and U are unweighted averages over all awakened nodes
Target speed = 5 m/sec; angle ∈ [−30°, 30°]

Figure 13  Simple averaging vs convex hull approach

conducted to compare the convex hull approach to averaging. Sample simulation results are provided in Figure 13. These indicate that the convex hull approach to position estimation has about the same performance as simple averaging.

4.4 Linear and Logarithmic Weighting

Two interesting weighted averaging approaches have been proposed in (17). This scheme estimates the distance of the sensor from the target and applies a weighting function that is monotonically decreasing with this distance. The distance estimate is obtained in (17) under the assumption that the target is moving in a straight line and that the sensor is always awake. In such a case, the distance between the sensor and the target can be related to the duration over which the sensor can sense the target. Proportional and logarithmic weighting apply weights that are linearly proportional to, and a logarithm of, this duration, respectively. In particular, in the two schemes, the weight factor for sensor node $i$ is given by

$$w_i = \left\{ \begin{array}{ll} \ln(1 + t_i) & \text{Log Wt} \\ (r_i^2 - 0.25(v(t_i - 1/f)))^{-1/2} & \text{Proportional Wt} \end{array} \right\} (17)$$

where $r$ is the closest distance between the track and sensor node $i$, $t_i$ is the time spent by the target within sensor $i$’s sensing range, $v$ is the velocity of the target, and $f$ is the sensor sampling frequency (17).

This weight factor is then normalized and then multiplied by the sensor position to estimate the position of the target. Such a scheme cannot be exactly applied to our case, since we do not assume that sensors are always awake. As a result, a sensor cannot be certain to sense the target for the entire duration over which that target is within its sensing range (since it may have been asleep for part of that time). If we simply use the duration for which the sensor does sense the target for weighting purposes, we obtain a position error that is noticeably greater than under simple averaging. Even if we use the actual time over which the target is within the sensing circle (this is obviously not available to the sensor for reasons stated above) and set $f = \infty$, the quality of the position estimate (marked “Opt” in Figure 14) is not significantly better than that of simple averaging. NU, however, is somewhat better in terms of position error because of the increased number of waking nodes closer to the target.
5 Sampling Rate and the Area of Sensing

We turn now to studying the impact of the sampling rate and the area of sensing on the performance of the system. Recall that in the general approach we are studying, we use successively larger areas of sensing (around the predicted target position) until either the target is found or the search is abandoned.

5.1 Random Mobility Model

The purpose of an analytical model is to provide the user with a rapid and intuitive approximation to the system under study. We introduce an analytical model of the power consumption of a two-step system. We should stress that this model depends on the central limit theorem, and is therefore an asymptotic model.

We assume a two-region system, consisting of two circular areas of sensing, of radius, $R_1$ and $R_2$, with $R_1 < R_2$. $R_2$ is chosen so that the probability of the target being within that circle is acceptably high. An extension of is to a larger number of regions is trivial.

The system attempts to obtain a fix on the target position every sampling interval of $t_s$ seconds. Since the target can change its direction of motion as it goes, the greater the sampling interval, the more inaccurate is likely to be the prediction of where the target will be at the next sampling epoch. The actual level of prediction error will depend on the mobility model of the target. We analyze two mobility models: one random and the other consisting of piecewise linear steps.

Assume that in each small interval of time, the target can diverge from its projected direction by a random variable which has some distribution. These divergences add up over time. If the individual divergences are stochastically independent, one can use the Central Limit Theorem to model the total divergence from the projected track approximately as a normally distributed random variable with zero mean and variance $kt_s$ for some constant, $k$.

Denote by $p_f(r, t_s)$ the probability of the target being in a circle of radius $r$ around the projected position for sampling interval $t_s$, and by $p$ the average number of awakened sensors per unit area (we model a uniform waking strategy over the search area). Then, the expected number of nodes involved in the sensing operation per unit time of tracking a target is given by:

$$N(R_1, R_2, t_s) = \frac{1}{t_s} \left\{ p_f(R_1, t_s) \pi R_1^2 \rho + (1 - p_f(R_1, t_s)) \pi R_2^2 \rho \right\}$$ \hspace{1cm} (18)

If we assume that the sensor density is sufficiently high (i.e., until a lot of nodes within the sensing area have died), then waking up all sensors within a radius of $r$ corresponds to being able to detect targets within a radius of $r + S$, i.e., targets which have diverged from their predicted point by between $-(r + S)$ and $r + S$. Hence, we can write

$$p_f(r, t_s) = \Phi \left( \frac{r + S}{k \sqrt{t_s}} \right) - \Phi \left( \frac{-r - S}{k \sqrt{t_s}} \right)$$
$$= 2 \Phi \left( \frac{r + S}{k \sqrt{t_s}} \right) - 1$$ \hspace{1cm} (19)

where $\Phi(\cdot)$ is the standard normal distribution with mean 0 and variance 1. Figure 15 shows some numerical results.

We can use these expressions to appropriately tune the system parameters. For example, suppose that we decide that $R_2$ has to be large enough that the probability of missing the target during any one sampling epoch is upper-bounded by some given value, $\mu$. Then, from (19) and the constraint that $R_2 \geq 0$, we have

$$R_2 = \max \left( k \sqrt{t_s} \Phi^{-1} \left( \frac{1 + \mu}{2} \right) - S, 0 \right)$$ \hspace{1cm} (20)

We can now calculate the expected power consumption per track. Let $e_{sens}$ be the energy consumed by a sensor in seeking a target, $e_{local}$ the expected energy required to communicate from a sensing node to its leader, and $e_{base}$ the expected total energy required to communicate from the leader to the base station. The expected power consumption per track can now be written down as

$$Pwr(R_1, R_2, t_s) = N(R_1, R_2, t_s) e_{sens} + \frac{\pi S^2 \rho e_{local} + e_{base}}{t_s}$$ \hspace{1cm} (21)

The designer will seek to set values of $t_s$ and $R_1$ to minimize the average power consumed per track.

In Figure 16 we provide a numerical example. For low sampling times, the power consumption is large since the probability of missing in $R_1$ is high; as $t_s$ increases, this drops to a minimum. With an increase in $t_s$, however, comes a requirement to increase the sensing areas. The power-optimal value for $R_1$ also increases...
with $t_s$, reflecting the increased uncertainty of the track. We should stress that the value of $R_1$ that is plotted here is the power-optimal one: it does not account for the impact of $R_1$ on the accuracy of the position estimation. The smaller the sensing circle, the greater the chance that the target will be close to the edge of that circle, which causes a bias effect since none of the nodes outside this circle is awake. This bias effect can be modeled similarly as in Section 4.2.

### 5.2 Piecewise Linear Mobility Model

In this mobility model, the target changes its direction every $T$ seconds forming a piecewise linear segment as. We assume that the target speed $s$ is fixed here (extending this model by assuming randomly varying speed can be done by applying Bayes’s Law and integrating over the range of allowed speed). Each such linear segment is at a displacement angle $\theta$ with respect to its predecessor, and is uniformly distributed over a certain range, $[\phi_{lo}, \phi_{hi}]$. We estimate the probability of the target being within a certain distance of the next sensing point by using as follows. Suppose there are $n$ linear segments between every two sensing points. We know the displacement that has occurred, in both the $x$ and $y$ dimensions between the previous sensing point and the current sensing point. Denote these displacements by $\alpha_x$ and $\alpha_y$, respectively.

Now, condition on the displacement angles of these $n$ segments: $\theta_1, \theta_2, \ldots, \theta_n$. For convenience, define $A_i = \sum_{j=1}^{i} \theta_j$. These have to be such that

$$d \sum_{i=1}^{n} \cos(A_i) = \alpha_x$$

(22)

$$d \sum_{i=1}^{n} \sin(A_i) = \alpha_y$$

(23)

Denote the displacement of the target between the current and next sensing points in the $x$ and $y$ directions by $\beta_x$ and $\beta_y$, respectively. We clearly require that

$$d \sum_{i=n+1}^{2n} \cos(A_i) = \beta_x$$

(24)

$$d \sum_{i=n+1}^{2n} \cos(A_i) = \beta_y$$

(25)

The joint probability distribution of $\beta_x, \beta_y$ conditioned on $\alpha_x, \alpha_y$ can now be obtained by applying Bayes’s law and integrating over all appropriate values of $\theta_i, i = 1, \ldots, 2n$. In practice, this can be achieved (for small values of $n$) by numerical means.

We can use this distribution function to calculate the probability, $P_c(r)$ that the actual position of the target is within a given distance, away from the predicted...
The expected energy consumption $E$, can now be written down:

$$E = (P_c(R_1)R_1^2 + (1-P_c(R_1))R_2^2)\pi \rho E$$  \(26\)

where $\rho$ is the node density and $E$ is the expected energy consumption per node involved in the sensing operation.

The effect of the mobility model ($\phi_{lo}$, $\phi_{up}$) on the catch probability in $R_1$ and expected energy consumption is described in Figures 17 and 18, respectively for a target moving at 5 m/sec for a variety of $[\phi_{lo}, \phi_{up}]$ values. We can see that as the uncertainty of the track increases (as expressed through a wider range of possible angles), our ability to catch the target in $R_1$ is reduced. As $R_1$ becomes larger, the probability of catching also increases since the target becomes more likely to fall within it. As expected, the optimum $R_1$ (for minimum expected energy consumption) depends on the mobility model: the greater the intrinsic uncertainty of the track, the greater the optimum value of $R_1$. The reason is quite simple to explain. When $R_1$ is very small, the probability of having to widen the search to the next wakeup radius is quite large, and increases with the track uncertainty (which is represented in our model by the maximum angular deflections of each segment). In such a case, the number of nodes that need to be awakened is greater and the drain on node energy reserves correspondingly larger; the second term in Equation 26 becomes more important. As $R_1$ increases, the need to switch to $R_2$ reduces, with a corresponding reduction in energy demands on the nodes. As $R_1$ becomes very large, however, the number of nodes far away from the target position that are awakened every time also increases, and energy drain is high. In this case, the first term in Equation 26 dominates.

We now turn to the case where there are three wakeup ranges (the third, $R_3$, here involves waking up all the nodes in the system that still have sufficient energy to function). Figure 19 and Figure 20 provide some numerical results. Selecting the value of $R_1$ that offers the lowest energy in the two-level case results in constant value of expected energy regardless of the value of $R_2$ shown in Figure 19, this is because the catch ratio at this optimum $R_1$ almost equals one. When we set $R_1 = 70$ as shown in Figure 20, for example, we can have a lower energy cost for certain $R_2$ values compared when we have the optimum $R_1$ value. The point of Figure 19 is that selecting the optimum $R_1$ does not always provide the minimum expected energy for different values of $R_2$ as shown in Figure 20.

Figure 21 shows the catch probability in $R_1$ for constant speed ($s = 5m/s$) and randomly varying speed over the interval $[2, 8]$ m/s. As expected, the speed uncertainty causes a decrease in the catch probability. This change, for obvious reasons, becomes insignificant as $R_1$ increases.

6 Filtering Out False Alarms

Nodes are susceptible to noise and noise can give rise to false reports of target acquisition. This false reporting has the potential to degrade the accuracy of the tracking
if it is not filtered out. We present here a simple noise filtering algorithm (shown in Figure 22) that attempts to filter out noisy nodes from target position estimation.

This is an iterative process, executed by the leader node. The algorithm estimates the target position based on all the reports sent by the nodes. Then, it filters out reports from nodes whose distance from that estimated target position exceeds the sensing diameter \(2 \times \text{sensorRange}\). This results in a new estimate of target position, and the process of filtration can be iterated until no reports have to be filtered out.

Figure 23 shows the effect of noisy nodes on position error. It can be seen that this simple noise filtering algorithm improves position error significantly compared with the no-filtering case for both the U and NU waking schemes. We can see, in general, that the NU scheme performs better than the U scheme because of the higher waking density surrounding the current target position. So long as the estimated target position is not far away from the actual position, the NU scheme will ensure that few nodes far away from the target will be awake. As a result, there will simply be fewer awake nodes outside the sensing range of the target.

7 Conclusion

In this paper, we have studied tradeoffs in target tracking by binary proximity sensor networks. We model the performance of simple node position averaging and show that it does about as well as a more complex convex hull approach. Weighting approaches that attempt to glean information about target distance by using the time the target spends in any one sensing circle are shown to perform no better than simple averaging. We model uniform and non-uniform node
wakeup in a region around the predicted target position and obtain insights into which is better under what conditions. We model the impact of node sensing range as well as the sampling time on the quality of the position estimate. We present a simple approach to filter out false alarms from malfunctioning sensor nodes.

This work has many extensions. The wakeup range is currently fixed; we are considering ways by which the system can learn the parameters of the intruder’s mobility model and adapt the wakeup range. This paper has concentrated on single-target problems; it can be extended to handle multiple simultaneous targets. Such an extension must include methods to disambiguate between targets even if they are relatively close together.

Acknowledgement: This material is partly based on work supported by the National Science Foundation under Grant No. CNS-0931035. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the national Science Foundation.

References


