

Technical Report No: WUCS-2002-37

A Multiple Hypothesis Markov Localization Approach to Tracking Moving Targets with Distributed Sensors

Weihong Zhang and Weixiong Zhang
{wzhang,zhang}@cs.wustl.edu
Department of Computer Science
Washington University
Saint Louis, MO 63130

Date: 10/28/2002

Abstract

Tracking or localizing a moving target is a difficult task in a distributed sensor network, due to the lack of knowledge of the target's motion and signal noises. Most existing approaches to the problem use only sensory information and may require accurate target's motion models. In this paper, we present a Markovian approach that combines dynamically estimated target's motion models with received sensory information. Without a given motion model, this approach localizes a target in two steps, a location prediction step using dynamically generated motion models and a location correction step integrating sensory readings. Our experimental analysis shows that our approach leads to substantially more accurate and robust location estimations than the previous approaches using only sensory information. In addition, we characterize probabilistic conditions under which the estimation accuracy increases if more sensors are used, and the estimations converge to the target's real position asymptotically. We show an interesting relationship between the steps of location prediction and location correction, i.e., as more sensors are used, belief correction guarantees the estimations to converge to the target's real position at each step, and belief prediction accelerates the convergence.

1. Introduction

Localization, i.e. determining the physical position of certain objects, is a fundamental function for many autonomous systems. The general localization problem has attracted much interest in diverse areas such as automatic control (Bar-Shalom, 1989; Blackman, 1989), robot navigation (Smith, Self, & Cheeseman, 1990; Moutarlier & Chatila, 1989; Fox, Burgard, & Thrun, 1999), military surveillance, context-aware systems (Chen & Kotz, 2000; Klemmer, Whitehouse, & Waterson, 2000; Bahl & Padmanabhan, 2000; Hightower & Borriello, 2001; Schmitt, Beetz, Hanek, & Buck, 2002; Want, Hopper, Falcao, & Gibbons, 1992), mobile computation and networking (Reichl, 2001; Takeda, 2001), and MEMS-related applications (Bulusu, Estrin, Girod, & Heidemann, 2001; Perkins, 2001; Toh, 2001; Deng & Zhang, 2002). Different application domains set forth different assumptions on sensing or perceptual models for receiving feedback from the world and motion models of the objects defining their behaviors.

We are interested in estimating the locations of constantly moving targets in a wireless sensor network (Zhang, Deng, Wang, Wittenburg, & Xing, 2002; Deng & Zhang, 2002). Deployed in such a network are small, low-powered micro-sensors with limited information processing capability. A set of sensors are grouped to detect a moving target. Localization refers to the procedure of estimating the position of a target with received sensor readings (e.g. signal strengths). Successful localization in this context has to overcome several technical difficulties. The system has restricted computational resources. For instance, the CPU speed and memory size are limited for micro-sensors. Another difficulty is the unreliability of the communication. Although wireless communication provides flexibility and mobility and extends significantly the applicability of sensor networks, the radio signals are typically unreliable and erroneous. Finally, maybe most importantly, the knowledge of a target's motion is unavailable or limited. Its frequent changes in velocity and direction make the localization task very difficult.

Several localization approaches have been proposed to estimate the locations of targets in such an environment (Klemmer et al., 2000; Bahl & Padmanabhan, 2000). Almost all approaches consider a discretization of the environment, called a *grid* hereafter. Broadly speaking, these approaches can be divided into two classes – approaches using only the sensory information and approaches using information from both sensor readings and a target's motion. In approaches using only sensory information, a sensing model provides a deterministic correspondence from a distance (between a sensor and a target) to a signal reading. These approaches combine sensor readings with sensing models. Thanks to the simplicity of the sensing models, the main advantage of these approaches is their computational efficiency. However, ignoring the target's behavior and the inadequate assumptions made in a sensing model lead to poor tracking results (Bahl & Padmanabhan, 2000; Klemmer et al., 2000).

The Kalman Filter(KF) approach has been successfully applied to estimating locations in a wide-range problem domains (Maybeck, 1979). KFs combine information from both a target's motion and sensor readings. KFs represent a sensing model as a Gaussian (normal) probability distribution for a distance between a sensor and a target. KFs also use a Gaussian to represent the estimations of a target's positions. Generally, linearity assumptions are made in the target's motion models and sensing models, although variants

and extensions on these assumptions exist. KF approaches are efficient and yield highly accurate estimations if motion models are adequate.

Unfortunately, the assumptions of KFs are not met in the domain of a distributed sensor network (Klemmer et al., 2000). First, no motion model is available for a target’s behavior. Targets move at various speeds and arbitrary directions. The alteration in speed and arbitrariness in direction add difficulties to the localization task. Second, because of the volatility of sensor readings, a sensing model cannot be described as a deterministic correspondence from distances to signal readings. It may not be necessarily described as a normal distribution either.

In this paper, we propose a new approach, which we call *multiple hypothesis Markov localization* (MHML) approach, to the problem of localizing moving targets using distributed sensors. One of the prominent features of this approach is that it does not require a motion model of a target. When a motion model is provided, the new method is able to incorporate it to improve the quality of localization. When the knowledge of a target’s motion is limited, as in most practical cases, our approach is also able to provide accurate estimations. Therefore, the new approach significantly extends a popular Markov localization approach in robot research (Fox et al., 1999) to deal with the important issue facing almost all localization methods, i.e., the lack of sufficient knowledge of a target.

MHML uses a probability distribution over a grid, which is referred to as a *belief*, to represent its estimation of a target’s position. MHML computes the next belief from the current one in two steps: *belief prediction* predicts an intermediate belief based on the target’s historical behavior, and then *belief correction* computes a new belief by using sensor readings to correct the intermediate belief. This new belief is the estimation of the target’s next position. In other words, belief prediction addresses the problem caused by the lack of knowledge about the target’s motion. At each step, MHML dynamically generates a set of hypotheses, each of which is an estimated motion model. MHML uses a maximum likelihood technique to select a hypothesis to apply. The selected hypothesis is used to predict a belief about the target. The predicted belief is then provided to the belief correction step.

MHML has several advantages over the approaches using only sensory information and KF-based approaches. First, the estimation of a target’s position is represented as a belief. This representation schema takes other schemas, for example those using only sensory information and KFs, as special cases. Second, the new approach has no restriction on the number and positions of sensors that can be used to localize a target. Third, the approach can apply to a problem domain where the knowledge of a target’s motion is unavailable or limited. If *a priori* knowledge of a target’s behavior is provided, MHML is able to exploit such knowledge to improve its efficiency and accuracy. Fourth, MHML can *globally* localize a target because it is not sensitive to the knowledge of a target’s initial position (Fox et al., 1999). Finally, the approach provides more accurate tracking results than approaches using only sensory information.

MHML also has some features that may be of theoretical interest. The belief prediction and belief correction steps in MHML enhance each other. Under some gentle statistical conditions to be detailed later, the belief correction step guarantees that the accuracy of position estimation monotonously increases with the number of sensors used, and the estimation asymptotically approaches to the target’s real position. Meanwhile, the belief

prediction step, which creates hypotheses on the target’s motion, is able to significantly accelerate the convergence of correct estimation.

However, MHML also has some drawbacks. The main weakness is its high computational cost. MHML is computationally expensive in our current straightforward implementation. Fortunately, many techniques have been proposed to improve the efficiency of probabilistic localization algorithms similar to the one in this paper (Fox et al., 1999; Zhou & Hansen, 2001). In addition, a rich set of particle filters techniques have been proven to be valuable tools to strike a balance between the solution quality and computational cost (Thrun, Fox, Burgard, & Dellaert, 2001; Arulampalam, Maskell, Gordon, & Clapp, 2002).

The rest of the paper is organized as follows. In the next section, we discuss two representational schemas of sensing models. In Section 3, we give an overview of the existing, related localization approaches. We then describe in detail our multiple hypotheses Markov localization approach in Section 4, and empirically compare it directly with the approaches using only sensory information in Section 5. We also investigate the robustness of the approach and empirically show that it is not sensitive to the knowledge of a target’s initial location. In Section 6, we theoretically analyze some features of the new approach. We show how its performance improves when sensor readings are integrated and when more sensors are used. Based on the theoretical results, we further examine how the belief prediction and belief correction steps interact and enhance each other in each step. We provide experimental evidence supporting our analysis in Section 7. In this section, we also show that a degenerate version of our algorithm is able to localize a target without using any knowledge of its motion. We discuss related work in Section 8, and finally conclude with a discussion of possible future directions in Section 9.

2. Representations of Sensing Models

A sensing model is an essential ingredient of a localization algorithm. Most sensing models are represented either in a sample-mean form or in a probabilistic form. These representations are the topics of this section.

2.1 Example sensor

We will describe these two representations based on our own experience on a radio sensor, which has a 4MHz 8-bit CPU with 512 bytes of flash memory (Deng & Zhang, 2002). We will also use this simple radio sensor throughout the paper, especially in our experimental analysis sections, to evaluate our new approach and compare it against many existing methods. Note that although the experiments are carried out on this particular sensor, our new localization algorithm is general and applicable to other types of sensors such as temperature, barometric and magnetic sensors.

Sensing models are usually determined by experiments. For example, to determine how radio signal strengths vary with distances, we carried out experiments of collecting signal strengths at different distances between sensors. We begin with 0mm in which two sensors are completely overlapped, and then sample the distances with the increment of 25mm. For each sampled distance, we repetitively collect the tested sensor’s readings 10 times. The results are plotted in Figure 1. The vertical axis denotes signal strengths and the horizontal axis denotes distances in millimeters. The middle curve plots the mean of

sampled signal strengths. For reference, we also plot the *upper bound curve* and *lower bound curve*. The strength in the upper (lower) bound curve is the maximum (minimum) among those collected at the same distance.

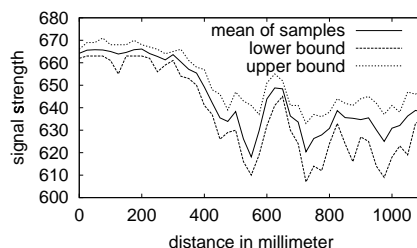


Figure 1: Signal strengths of a micro-radio-sensor vary with distances

The main curve in Figure 1 is typical as reported in the literature (e.g. see (Bahl & Padmanabhan, 2000)). It also provides some clue to the difficulty inherent in the localization problem using such sensors. The main source of difficulty is that signal strengths do not reflect the true distance between a sensor and a target. For this reason, it is not a surprise that distance estimation is not accurate using such a sensor. This is sometimes referred to as the *distance inference* problem (Klemmer et al., 2000). It can be seen from Figure 1 that the signals are almost indistinguishable for distances in the interval of [0mm,300mm]. For a signal reading of 665, for example, there are potentially infinite number of possible distances. Another example is signal reading of 630, to which at least 6 possible distances can be inferred.

2.2 Sample-mean Representation

Many localization approaches represent a sensing model by the mean of sampled signal strengths with respect to distances. Despite that the signal strength is the average of sampled readings, this representation actually assumes that given one distance the sensor produces only one signal strength. It ignores other possible readings between the lower bound and upper bound curves.

A sensing model can be represented as a piecewise linear function. The mean values are kept only for the end points corresponding to sampled distances. However, since distance is continuous, we need a procedure to generate signal strengths for every possible distance. Linear interpolation can serve this need well. If a distance d lies in the middle of two sampled distances d_1 and d_2 and their signal strengths are μ_i , the mean signal strength μ for distance d can be generated as $(1 - \lambda)\mu_1 + \lambda\mu_2$, where $\lambda (= (d - d_1)/(d_2 - d_1))$ denotes the closeness of d to d_1 .

2.3 Probabilistic Representation

The sample-mean representation is deterministic, in that implicitly for each distance there is only one signal strength corresponding to it. This is obviously insufficient in reality. For

example, Figure 1 shows that there may be multiple signal strengths for one distance, which lie between the two dashed curves.

A probabilistic representation takes possible signal strengths into account. It assumes that for a distance the signal strengths can be depicted by a probability distribution. Such a representation discriminates different signal strengths by assigning them with different probabilities measuring the likelihoods by which they are produced.

Figure 2 presents such a probabilistic sensing model. In the left chart, the x-axis is for distance. For each distance, it is possible to receive any signal strengths, but each of them is produced with a specific probability. A thick solid curve in the left chart thus represents a probability density function (p.d.f.) of signal strength for a given distance. The right chart, for instance, further demonstrates a p.d.f. for distance d_1 .

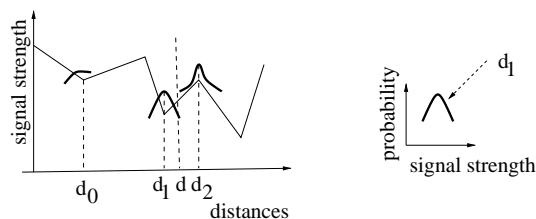


Figure 2: An example probabilistic sensing model

Determining the p.d.f. of signal strength for a distance is a matter of experimentation. In our empirical studies, we assume that the p.d.f. for a distance is Gaussian. Its mean and variance are determined as follows. For a distance d , if the measured signal strengths are r_1, r_2, \dots and r_n after n experiments, the Gaussian $N_d(\mu, \sigma)$ conditional on d is defined by its mean μ ($= \sum_{i=1}^n r_i/n$) and variance σ ($= \sqrt{\sum_{i=1}^n (r_i - \mu)^2}$). A linear interpolation procedure of generating a p.d.f. $N_d(\mu, \sigma)$ for any distance d follows. Let d_1 and d_2 be two sampled distances most closely to d and their p.d.f. be $N_{d_i}(\mu_i, \sigma_i)$. The p.d.f. $N_d(\mu, \sigma)$ can be defined by $\mu = (1 - \lambda)\mu_1 + \lambda\mu_2$ and $\sigma = \sqrt{((1 - \lambda)\sigma_1)^2 + (\lambda\sigma_2)^2}$ where λ ($= (d - d_1)/(d_2 - d_1)$) is the coefficient denoting the closeness of d to d_1 . As such, by the p.d.f.s for sampled distances, one can produce a p.d.f. for any distance.

In the rest of the paper, a probabilistic sensing model is denoted by $p_d(r)$, which is the probability of a sensor producing r if its distance from a target is d^1 . To explicitly indicate a target's position and the identity of a sensor, we will use $p_{il}(r)$ to denote the sensing model of sensor i , which gives the probability that sensor i collects strength r if the target is at location l . The two notations p_d and p_{il} are equivalent. One form can be converted to the other because sensor i is stationary and the distance d can be determined by the position of sensor i and the target's location l .

1. Mathematically, this is correct if the range of signal strengths is a discrete space. However, if the range is continuous, it is known that for any r the probability of sensor i producing r is 0.0. In this paper, we use the value $p_d(r)$ to mean the probability (likelihood) as in many papers because these values serve similar purposes as the discrete case.

3. Previous Approaches

We will compare our new method to three existing localization approaches in sensor networks. The triangulation approach and nearest neighbor in signal space use the sample-mean representation of sensing models². Kalman filter approach uses a probabilistic model and exploits a target’s motion information. To put our comparison in perspective, we describe these existing methods in turn in this section.

3.1 Triangulation Approach

The triangulation approach uses the geometric properties of a triangle to determine the location of a target. In the localization phase, a set of readings from detecting sensors are received. By the sample-mean representation, each reading is used to infer a distance between the target and a sensor. This results in a set of distances. Since sensors are stationary and their locations are known, the distances can be used to determine a location. This location is the estimation of the target’s position.

Figure 3 shows how this approach works in a 2D case. A target is detected by three sensors S_1 , S_2 and S_3 . At one step, three sensor readings r_i are received. By reading r_i and the model of sensor i , a distance d_i between S_i and the target is inferred. The target should lie on the circle centering at S_1 with a radius of d_1 . Similarly, the reading r_2 can be used to constrain the target’s position to one of the two intersections A and B . Finally the reading r_3 can be used to further constrain the candidate location to B , which is the estimation of the target’s position.

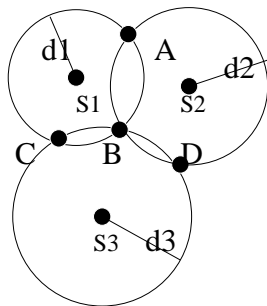


Figure 3: An illustration of the triangulation approach

The triangulation approach imposes requirements on both the number and the locations of detecting sensors (Harter, Hopper, Steggles, Ward, & Webster, 1999). In 2D case three sensors are needed and they must be non-collinear. In 3D case four sensors are needed and they must be non-coplanar. This is one of the main drawbacks of this approach. Additionally, the approach has to confront the distance inference problem. Due to the lack of effective inference techniques, the triangulation approach typically provides very inaccurate position estimations. The major merit of this approach is its efficiency. The

2. We choose these two approaches because they are representatives of two principal techniques – triangulation and proximity – for automatic location-sensing (Hightower & Borriello, 2001).

computation simply includes inferring distances and determining the intersections of circles in the 2D case.

3.2 Nearest Neighbor in Signal Space

Another approach only using sensing models is *nearest neighbor in signal space* (NNSS) (Bahl & Padmanabhan, 2000). This approach typically uses sensing models in sample-mean form. The main idea of this method is to differentiate, for each location l , the received sensor reading and the possible reading that can be generated if a target is at location l . The location with the minimum signal differentiating value is then the *nearest neighbor* of a target's position, which is the estimation of the target's position.

Given a sensing reading r_i , the signal differentiating value at a location l can be computed as follows. Denote d_i the distance between l and the position of sensor i . NNSS uses d_i to determine the mean signal strength μ_i based on the sensing model (cf. Figure 1) if the target is assumed to be at location l . The total differentiating value is then the overall difference of r_i and μ_i for all i , which can be computed using a metric function such as the Euclid distance, i.e. $\sqrt{\sum_i (r_i - \mu_i)^2}$, where the sum is over all detecting sensors. The location l with the minimum value is the estimation of the target's position.

NNSS has no restriction on the number and the locations of detecting sensors. In addition, NNSS avoids the distance inference problem by using a metric function. When the received sensor readings center around the sampled mean in the model, NNSS yields more accurate estimations than the triangulation approach. However, since NNSS needs to measure every location at each step, it is computationally more expensive than the triangulation approach on problems with large location spaces.

3.3 Kalman Filtering

Kalman Filter (KF) based approaches have been used for localization in sensor network (Klemmer et al., 2000). KF uses probabilistic sensing models and also requires a target's motion model. It takes two steps in localization. The location projection (or time update) step projects an estimation of the target's position at next step. The location correction (or measurement update) step corrects the estimation with sensor readings.

The generic KFs assume that both motion and perceptual models are linear, and the noise of perceptions is Gaussian. It is not surprising that existing work such as (Klemmer et al., 2000) devoted much effort to linearly approximating the sensing model and determining a target's motion model. The main advantage of KFs is that if the assumptions are appropriate for the problems at hand, they are able to efficiently provide accurate solutions. However, KFs have a few disadvantages. First, KFs use a uni-modal Gaussian to represent the estimation of a target's position. This representing schema is insufficient in some cases. For example, it is unable to represent a target's position if one has no idea about the position. Second, KF algorithms can barely recover from an estimation error, as noted by some authors (e.g., (Fox et al., 1999)).

4. Multiple Hypothesis Markov Localization

In this section, we give a detailed description of our multiple hypothesis Markov localization (MHML) approach and compare it with the algorithms in the preceding section. We also discuss efficient implementations of MHML at the end of this section.

4.1 The Algorithm

MHML combines the received sensing readings and the motion information of moving targets, which can be given or estimated dynamically. Therefore, this approach does not require a motion model. In the following, we first outline the algorithm and then describe its steps in detail. For simplicity, we focus on the problem of localizing one target, but the algorithm can be extended to multiple targets straightforwardly.

4.1.1 OUTLINE OF THE ALGORITHM

MHML uses a grid to denote the discretization of the area within which a target moves. At any step, the estimation of the target’s position is a probability distribution over the grid. Such a distribution is referred to as a *belief* about the target. Localization is a procedure to compute the next belief about the target from the current belief using a set of (estimated) motion models and perceived sensor signal.

MHML computes the new belief in two steps. The belief prediction step predicts an intermediate belief according to the target’s historical behavior. The belief correction step computes a new belief using the sensor readings to correct the intermediate belief. The resulting belief is the estimation of the target’s position at the next step. The belief also serves as the current belief in the next localization step, and the algorithm repeats.

More specifically, the belief prediction step addresses the problem of the lack of knowledge about the target’s movement. Based on the beliefs of the target’s current and previous positions, MHML dynamically generates a set of hypotheses about the target’s possible movements at the next step, each of which is then used to project a possible next belief. To choose one from these projected beliefs, MHML evaluates them and selects the best one according to certain criteria. The selected belief is then the input to the belief correction step. The corrected belief is the estimation of the target’s next position.

To facilitate our discussion, we introduce the following notations. The discretized grid is denoted by G . A belief about the target is denoted by b . For any location l in G , $b(l)$ is the probability of the target being at location l . Note that $\sum_{l \in G} b(l) = 1.0$. The notation b and b_+ represent the beliefs about the target respectively at the current and next step. The central part of MHML is to compute b_+ from b .

Table 1 outlines one-step localization *Localize* of MHML to compute an updated belief b_+ from b . It makes use of sensing models $\{p_{il} | i = 1, \dots, m; l \in G\}$ and sensor readings $\{r_i | i = 1, \dots, m\}$, where m is the number of detecting sensors and r_i the reading of sensor i . In the table, the procedure *HypothesisGeneration* generates a set \mathcal{H} of hypotheses of the target’s possible movements. The procedure *BeliefProjection* projects next beliefs from the current belief b by using the generated hypotheses. Its output is a set $\{b_+^h\}$ of beliefs, each of which corresponds to a hypothesis h . The beliefs are then evaluated in the procedure *BeliefScoring*. According to the evaluation results, *BeliefSelection* selects the

winning belief. The procedure *BeliefCorrection* corrects the selected belief. The corrected belief is then the estimation of the target’s position.

Localize

input : $b, \{p_{il}\}$ and $\{r_i\}$
output : b_+
1. $\mathcal{H} \leftarrow \text{HypothesisGeneration}()$
2. $\{b_+^h\} \leftarrow \text{BeliefProjection}(b, \mathcal{H})$
3. $\{b_+^h\} \leftarrow \text{BeliefScoring}(\{b_+^h\}, \{p_{il}\}, \{r_i\})$
4. $b_+^h \leftarrow \text{BeliefSelection}(\{b_+^h\})$
5. $b_+ \leftarrow \text{BeliefCorrection}(b_+^h, \{p_{il}\}, \{r_i\})$
6. **Return** b_+

BeliefProjection

input : b, \mathcal{H}
output : $\{b_+^h | h \in \mathcal{H}\}$
1. **For** each h in \mathcal{H}
2. compute b_+^h by Equation (1) where $p(l|l')$ is specified by h
3. **Return** $\{b_+^h\}$

BeliefScoring

input : $\{b_+^h\}, \{p_{il}\}, \{r_i\}$
output : $\{b_+^h\}$
1. **For** each h
2. compute $p(\{r_i\} | b_+^h)$ by Equation (2)
3. **Return** $\{b_+^h\}$

BeliefSelection

input : $\{b_+^h\}$
output : b_+^h
Return b_+^h with largest $p(\{r_i\} | b_+^h)$

BeliefCorrection

input : $b_+^h, \{p_{il}\}, \{r_i\}$
output : b_+
1. compute b_+ by Equation (3)
2. **Return** b_+

Table 1: Multiple hypothesis Markov localization

It is important to note that MHML can start with a known initial belief (initial target position), if it is given, or no initial belief at all, which can be represented by a uniform distribution.

4.1.2 HYPOTHESIS GENERATION

This step generates a set of hypotheses characterizing the target’s movements at one point. Since we do not assume *a priori* motion model, generating perfect hypotheses is infeasible. We propose a simple heuristic approach.

A target’s motion can be defined by two key elements, moving direction and velocity. If they are known, a good hypothesis can be generated provided that one has an accurate estimation of its current position. However, neither the direction nor the velocity is directly available in most problem domains, including our distributed sensor networks. We thus propose to enumerate the possible directions and to update the velocity by a small increment for the purpose of generating motion hypotheses.

- Generally speaking, there is an infinite number of moving directions. A rough approximation is to consider 360 directions, each of which corresponds to 1 degree in angle. To further simplify the approximation, we can consider only four nominal directions (east, south, west and north).
- Velocity is a more complicated element to estimate. Fortunately, in general, targets move smoothly. When changing speed, they seldom make the change abruptly. This allows us to estimate the velocity at the next step as an additive form of the current velocity and a *base increment*, reflecting small changes. Note that this speed estimation assumes that a target’s initial speed is known. In our experiments, the base increment is set to the difference between a target’s estimated positions at the previous two steps. To deal with more complex behavior of a target, we may add multiple increments. Some of them may be larger or smaller than the base increment.

Each combination of enumerated directions and estimated velocities forms a hypothesis. Combined with the estimation of a target’s current position, a hypothesis determines a unique location for the target at the next step. This location is referred to as *the projected location* generated from the hypothesis.

To reduce the potential error of the projected location, we use a probabilistic transition model. The idea is that locations nearby the projected one take more weight, if the directions from the target’s current estimated location to them are consistent with that in the hypothesis. As a result, a hypothesis determines a probabilistic motion model. We denote such a model by $p(l|l')$, where l' and l are any locations in the grid, and $p(l|l')$ is the transition probability from location l' to location l . This model is Markovian since the target’s next location depends only on its current location. Due to this dependent relationship between an estimated motion model and a hypothesis, from now on we use these two terms interchangeably.

All possible hypotheses thus generated constitute a hypothesis space. Its size is the multiplication of the number of enumerated directions and the number of considered velocity increments.

We note that on-line learning techniques can be used to construct the hypothesis space (e.g. (Sutton, 1984; Simmons & Koenig, 1995; Thrun, Burgard, & Fox, 1998)). We also note that *a priori* knowledge on a target’s motion, if any, can be used to reduce the size of hypothesis space and therefore to effectively accelerate the procedure of hypothesis generation. An important case is that if a (probabilistic) motion model is given, a space of

only one hypothesis determined by the given model is generated. In another case, if we know that a target moves in a constant speed, a hypothesis space of size 360, in which only directions are considered, could serve as a good approximation to the target’s behavior.

4.1.3 BELIEF PROJECTION

Given a belief state and a hypothesis space, the belief projection step calculates a set of beliefs. The set is obtained by considering each hypothesis in the space. We now consider how to compute the projected belief for one hypothesis.

Let the motion model determined by a hypothesis h be $p(l|l')$ and the current belief be b . The projected belief, denoted by b_+^h , is computed by the total probability theorem. For each l in G ,

$$b_+^h(l) = \frac{\sum_{l'} p(l|l')b(l')}{\sum_l \sum_{l'} p(l|l')b(l')} \quad (1)$$

In the above, the denominator is a normalizer ensuring $\sum_l b_+^h(l) = 1.0$.

The pseudocode for belief projection is shown as the procedure *BeliefProjection* in Table 1. Line 1 considers each hypothesis and line 2 calculates a projected belief state for the hypothesis. Line 3 returns the union of the projected beliefs.

4.1.4 BELIEF SCORING

The belief scoring step evaluates each projected belief and assigns to it the evaluation result as a belief score. Belief scoring uses sensing models $\{p_{il}\}$ and sensor readings $\{r_i\}$. More specifically, for a belief b_+^h , it computes $p(\{r_i\}|b_+^h)$, i.e. the probability of the sensors receiving readings $\{r_i\}$ conditional on the belief b_+^h . In the derivations below, we assume that individual sensors receive their readings independently.

$$\begin{aligned} p(\{r_i\}|b_+^h) &= \sum_l p(\{r_i\}|\text{target is at location } l)b_+^h(l) \\ &= \sum_l \Pi_i p(r_i|\text{target is at location } l)b_+^h(l) \end{aligned}$$

Here the first step uses the total probability theorem and the second step uses the independence assumption. Therefore,

$$p(\{r_i\}|b_+^h) = \sum_l \Pi_i p_{il}(r_i)b_+^h(l) \quad (2)$$

The pseudocode for belief scoring is included in *BeliefScoring* in Table 1. Its input includes a set $\{b_+^h\}$ of predicted beliefs, sensing models $\{p_{il}\}$, and a set $\{r_i\}$ of sensor readings. The output is the same set of beliefs, each of them associated with a probability.

4.1.5 BELIEF SELECTION

This step selects the belief b_+^h with the largest $p(\{r_i\}|b_+^h)$ or based on some other criterion. In our study, we use the largest probability criterion for simplicity. If the probability $p(\cdot|b_+^h)$ is viewed as a likelihood function, the selection criterion implies that the selected belief is the maximum likelihood estimation of the belief about a target among all the projected beliefs. Since every projected belief corresponds to a hypothesis, we can also talk about the maximum likelihood hypothesis.

The pseudocode for belief selection is given in *BeliefSelection* in Table 1.

4.1.6 BELIEF CORRECTION

This step makes a correction to the selected belief b_+^h in belief selection using sensing models $\{p_{il}\}$ and sensor readings $\{r_i\}$. The probability of the target being at location l conditional on b_+^h and $\{r_i\}$ can be obtained from basic probability theory as follows.

$$\begin{aligned} p(l|\{r_i\}, b_+^h) &= \frac{p(\{r_i\}|l, b_+^h)p(l|b_+^h)}{p(\{r_i\}|b_+^h)} \\ &= \frac{p(\{r_i\}|l)b_+^h(l)}{p(\{r_i\}|b_+^h)} \\ &= \frac{\Pi_i p_{il}(r_i)b_+^h(l)}{p(\{r_i\}|b_+^h)} \end{aligned}$$

The probability $p(l|\{r_i\}, b_+^h)$ is the corrected belief at location l . We use b_+ to denote the corrected belief $p(l|\{r_i\}, b_+^h)$. The above equation can be written as follows.

$$b_+(l) = \frac{1}{c} \Pi_i p_{il}(r_i) b_+^h(l) \quad (3)$$

where c is the normalizer $p(\{r_i\}|b_+^h)$. The formula indicates that the corrected belief at a location is proportional to the input belief at that location and the probability of receiving the set of readings.

The step is implemented as *BeliefCorrection* in the table. It computes a new belief using the formula, which is the estimation of the target's real position.

4.2 Characteristics and Comparison

MHML provides a framework to localize a target by integrating information from two resources, the estimated motion models and sensor readings. To tackle the problem of lack of motion model, MHML dynamically generates a set of hypothetical motion models. It then uses a maximum likelihood method to choose the best among all generated motion models. The chosen motion model is used to predict a belief. The predicted belief is then corrected so as to be consistent with the perceived signal readings. The corrected belief is the estimation to the target's position. The algorithm has several advantages when compared with the triangulation, NNSS and KF approaches.

MHML uses a belief over a grid to represent the estimation of a target's location. This representational scheme is general and takes other schemas as special cases. A belief can be specialized to be a unique location when probability mass concentrates on the location in the triangulation and NNSS approaches. It can also be specialized to a Gaussian in the KF approach.

MHML has no restriction on the number and the positions of the detecting sensors. Also, it does not directly address the distance inference problem. This is similar to NNSS but different from the triangulation approach. In a sensor network, it is important for a localization algorithm to use whatever available sensors to detect a target. When resources are scarce, MHML is able to detect a target with even one sensor. On the other hand, when resources are plentiful, the algorithm is able to make use of all the available sensors in order to improve tracking quality. Following this, one important associated question is

whether more sensors will indeed give rise to more accurate localization results. We will address this question later in the paper. We will show that this is true for MHML under some circumstances.

MHML is applicable regardless whether the knowledge of a target’s initial position is available. It is able to use the knowledge of the target’s initial position, if available, to improve the accuracy of position estimations. If a target’s initial position is known, MHML sets the initial belief to that position. Otherwise, it sets the initial belief to a uniform distribution. This differs from the triangulation and NNS approaches, which do not make use of the target’s initial position, even if it is available.

Furthermore, MHML is applicable regardless whether a target’s motion model is given. If there is no knowledge of a target’s motion at all, later on in this paper, we will show that a degenerate version of MHML can be used to localize it. If the knowledge of a target’s motion is limited, MHML will generate a set of hypothetical motion models at each step. If a motion model is given, MHML generates a singleton hypothesis space and the algorithm degenerates to the usual Markov localization. Again, the triangulation and NNS approaches do not make use of a target’s motion knowledge.

However, there are also disadvantages of MHML. The main problem is its high computational cost. The first expensive step is belief projection. It takes $(|G|)^2$ operations since for each location MHML needs to consider the reachability from any other locations. The second expensive step is belief correction. It takes $|G|$ operations since MHML has to update the belief at each location. Finally and more prominently, the hypothesis space can be huge, and MHML needs to deal with every hypothesis in belief projection.

4.3 Efficient Implementation

To curtail its high computational cost, we outline some of the measures we can take to develop an efficient implementation of MHML.

The first one is concerned with motion models. For one hypothesis, storing a model $p(l|l')$ takes $|G|^2$ memory units. Since the hypothesis space could be huge and the grid size is exponential in the dimension of location space, it is infeasible to keep motion models in memory especially for micro devices. One method is to generate them dynamically. For a location l , computing $b_+^h(l)$ in Equation (1) needs only those locations with non-zero transition probabilities $p(l|l')$. In our current implementation, we never generate and keep those locations l' if $p(l|l')$ are zero or negligible. This greatly saves space for the motion model. Meanwhile, this representation makes belief projection efficient. For a location l , we do not need to consider the entire grid in belief projection as in Equation (1). Instead, we can consider the locations l' with nonzero $p(l|l')$.

The second measure is concerned with belief projection. For one hypothesis, belief projection computes a belief by updating the belief at each location. Techniques have been proposed to accelerate the procedure. One method is to restrict the updates to a selective set of locations in regular grid approach. In (Fox et al., 1999) for example, the grid is divided into regions, each of which is a subset of the grid. Belief projection is then updated over locations in selected regions. The authors of (Fox et al., 1999) show that this subdivision technique not only produces approximately good results but also makes belief update feasible for real time applications. Another applicable technique uses a variable-

resolution grid instead of a fixed-resolution grid. (Zhou & Hansen, 2001) develops techniques to optimize the tradeoff between the approximation quality and computational time. These techniques can be combined to speed up belief projection.

5. Empirical Studies

We experimentally analyzed the performance of MHML. We now report our results from three experiments. We show in the first experiment that in terms of accuracy, MHML performs significantly better than both the triangulation approach and the NNS approach. We excluded the Kalman filter approach from our comparison here because no target’s motion models are assumed. In the second experiment, we show MHML’s robustness when no target’s initial position is provided. Finally, we show in the third experiment how a noisy sensing model affects the performance of MHML.

5.1 Experiment Setups

Our experiments were carried out in a 1000mm×800mm simulation environment, as shown in Figure 4. 50 stationary sensors are scattered around the environment. Sensors are evenly distributed in an array of 5 rows by 10 columns. In the figure, the upper (lower) dotted line is the 1st (10th) row and the left (right) dotted line is the 1st (5th) column. The positions of the left-down and right-top sensors are respectively (175, 100) and (900, 520). When detecting targets, we randomly picked as many sensors as needed.

To test the performances of algorithms for different (unknown) motion behaviors, we design two classes of targets: *directed targets* and *random targets*. A directed target follows a designated path that is characterized by a set of anchoring locations. If a directed target reaches the end of the designated path, it is reset to its initial position. Examples of directed paths are shown as solid lines in Figure 4. A random target starts at a location and randomly moves along one of four nominal directions with a predefined probability. The direction along which a target moves with the largest probability is the *dominant* direction. Dominant directions are marked by E(ast), S(outh), W(est), N(orth) or R(andom) in the figure, where R(andom) means that a target moves along four directions with equal probability. All targets move in the environment with a constant or variable speed to be specified later. In addition, no target is allowed to move out of the environment, i.e., a movement that may push a target out of the environment will have no effect.

We used two sensing models in our experiments, drawn in Figure 5, and assume that these models are accessible to every sensor. Unless explicitly specified, the default model of most of our experiments is the one in the left chart of the figure. In both charts, the middle curves plot the mean of sampled signal readings along distances. We represent the p.d.f. at any distance as a Gaussian. We use methods in Subsection 2.3 to determine p.d.f.s at sampled distances and extrapolate p.d.f.s at any other distances. Obviously, the right chart corresponds to a noisier model than the left. First, we note that the set of sampled distances (end points) in the left is a subset of that in the right. Second, due to the piecewise representation of a model, the “pieces” in the right chart split the “pieces” in the left chart. For instance, the piece in distance interval [450,725] of the left chart are broken into 2 pieces in the right chart, i.e. pieces in [450,600] and [600,725]. In terms of the mean of p.d.f.s, the left model is linear in [450,725] but the right model is non-linear in the same interval.

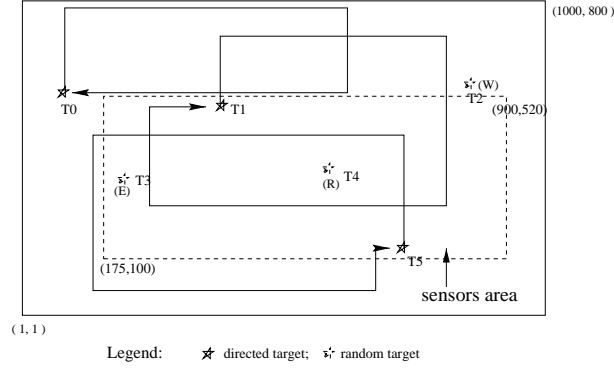


Figure 4: Simulation environment

For simplicity, we refer to these two models as *5-piece* and *10-piece* models throughout the discussions below.

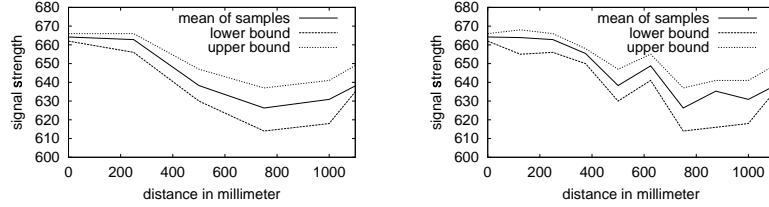


Figure 5: Sensoring models used in experiments

A target can move in a constant or variable speed. A random target is free to choose one of the four nominal directions at any step. If a target moves with a constant speed, its speed is set to the base speed, i.e. 5mm per step. If a target moves with a variable speed, its speed at any step is generated from the base speed and an increment that is uniformly distributed in $[0\text{mm}, 5\text{mm}]$. Specifically, if the increment is δ , a target's speed is generated as $(1.0 - \lambda) \cdot 5 + \lambda \cdot \delta$, where λ is set to 0.1. Thus, a target's speed will not grow to infinity. To facilitate hypothesis generation, an estimated speed is maintained for each target at any step. For a constant-speed target, the estimation of its speed at the first step is set to the base speed, and the estimation at any other step is set to be the distance of the estimated locations at the previous two steps. For a variable-speed target, the estimation of its speed at the first step is set to the base speed, and the estimation at any other step is set as: $(1.0 - \lambda) \cdot (\text{the current estimated speed}) + \lambda \cdot (\text{the distance of the estimated locations at the previous two steps})$. As such, the speed estimation of a directed target does not assume any knowledge about its motion except at the first step, and the estimation of a random target uses limited knowledge, i.e., λ . (For the case that there is no *a priori* knowledge, we will discuss a degenerate version of MHML later in this paper.) For a target, at any

step hypothesis generation generates a space of 4 hypotheses, each of which is formed by a nominal direction and the estimated speed.

The performances of a localization algorithm are evaluated by two measures. The primary measure is trajectory accuracy, representing the closeness of an overall estimated trajectory to an overall real trajectory. The secondary measure is distance error, measuring the distance between the estimated and the actual locations. To compare average performances, each algorithm takes 50 trials, each of which runs 300 steps. The reported results are the average estimations over all trials across individual steps.

5.2 Comparison of Algorithms

In this experiment, we compare MHML with triangulation and NNSS algorithms for tracking a moving target using only 3 sensors. We report results on a directed target with constant speed and a random target with variable speed.

We present the results on a directed target with a constant speed in Figure 6. It plots the target’s true trajectory and the trajectories tracked by the three algorithms. It is evident that MHML is able to trace the shape of the real path, while neither triangulation nor NNSS is able to do so. The triangulation approach gives a shape far from the real trajectory. The trajectory by the NNSS approach does move out of the cloud near the target’s initial location. However, NNSS is still not able to find a path roughly reflecting the shape of the real trajectory.

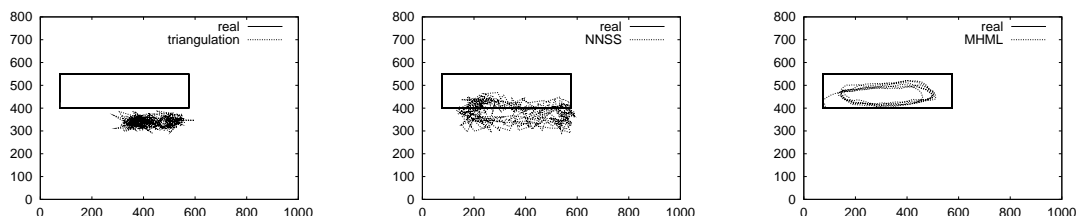


Figure 6: Tracked trajectories by triangulation, NNSS and MHML on a directed target

We conduct similar comparisons on a random target with variable speed. The real and tracked trajectories are drawn in Figure 7. The results on the random target are similar to those from directed targets. Again, the triangulation approach yields the worst accuracy. NNSS performs better than the triangulation method in that the tracked trajectory is able to move out of the vicinity of the target’s initial position. MHML is clearly the best one for this target. Its tracked trajectory almost traces the real one.

From these results, we see that MHML performs significantly better than triangulation and NNSS approaches. The reason is that MHML exploits information from both sensor readings and estimated target’s behavior, and the other two approaches only make use of sensory information. Even if the motion information is generated dynamically, the estimation accuracy of MHML implies that the multiple-hypothesis technique is effective in estimating targets’ motions.

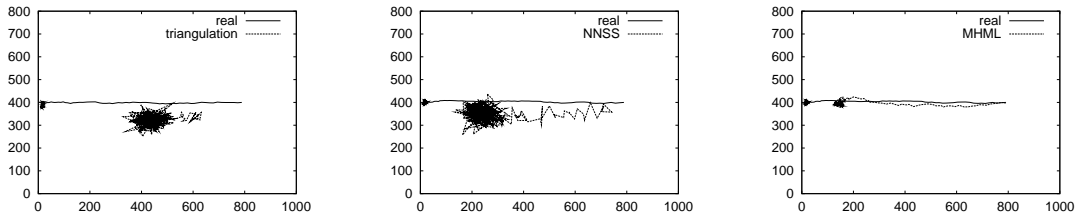


Figure 7: Tracked trajectories by triangulation, NNSS and MHML on a random target

5.3 Robustness of MHML

This experiment is designed to test the robustness of MHML, i.e. how a noisy sensing model degrades the performance of MHML. We tested MHML by two sensing models, shown in Figure 5. We plot the results in Figure 8. The figure presents a directed target’s real trajectory and trajectories tracked by MHML using the 5-piece and 10-piece sensing models. The results are obtained using 5 sensors. As shown in the figure, the trajectory by the noisier 10-piece model is farther away from the true trajectory. Hence, a noisier model degenerates the performance of MHML. Fortunately, the trajectory tracked by MHML using the noisier model has almost the same shape of the real trajectory.

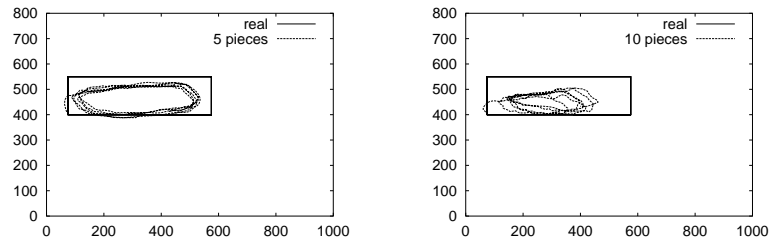


Figure 8: Tracked trajectories by MHML using a noisier sensing model

Late in this paper, we will show that the performance degradation of MHML with noisy sensing models can be compensated by increasing the number of detecting sensors.

5.4 Knowledge of Initial Position

This experiment is designed to analyze the sensitiveness of MHML to a target’s initial position. To this end, we run the algorithm in two versions. In one version the initial belief about a target in MHML is set to the target’s real location, while in the other version it is set to a uniform distribution over the entire grid.

The tracked trajectories by MHML using 5 sensors are shown in Figure 9. The left and right charts are respectively for MHML starting with the target’s real position and a uniform distribution. As the result shows, the overall performance of MHML without

knowing the initial target position is comparable to that of MHML with an initial position. This means that in a long run, MHML is almost insensitive to the initial belief of target's position.

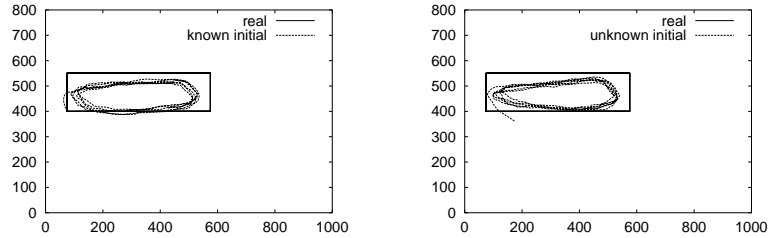


Figure 9: Tracked trajectories by MHML with and without initial position

However, it is expected that the knowledge of a target's initial position can help improve the estimation accuracy at least at the first steps. To shed light on this, we plot the distance errors along steps in Figure 10. The left chart is for the tested 300 steps, and the right chart is especially for the first 7 steps. The left chart shows that the overall distance errors are almost the same for MHML with and without the initial position. The right chart shows that the initial position is very useful in improving the estimation accuracy at the first 3 steps.

6. Analysis

Despite its superior performance over that of the two existing methods, the properties of MHML remain to be analyzed, which are the subject of this section. We start with a summary of our results.

6.1 Summary of Results

To put our results in perspective, let us re-examine the algorithmic structure of MHML. To reiterate, MHML computes an estimation (a belief) about a target's position in two

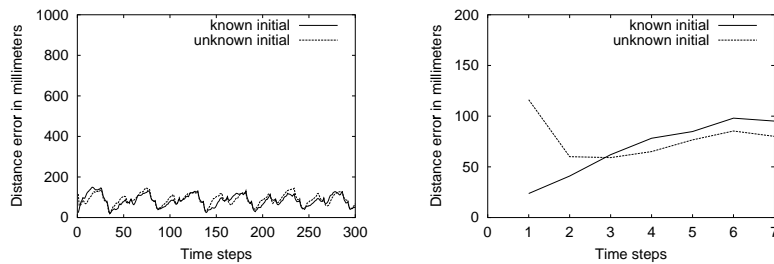


Figure 10: Distance errors by MHML with and without initial location

steps. First, the *belief prediction* step predicts the best possible belief from the current belief through a sequence of hypotheses generation, belief projection, belief scoring and belief selection. Second, belief correction step corrects the predicted belief using perceived sensor readings. The corrected belief is then the estimation of the target’s position at the next step. We specifically show that these two steps interact with each other in a supportive way. Specifically, under some conditions, as the number of detecting sensors increases, the belief correction step guarantees the convergence of estimations to a target’s real position and the belief prediction step accelerates this convergence.

For belief correction, we will examine the following issues: how sensor reading information will affect the beliefs about a target, how the beliefs will change when more sensors are used, and how the beliefs will evolve when a large number of sensors are used. Our answers are the following. Under some conditions, to be specified later, integrating a sensor reading increases the belief at the target’s real position, more sensors lead to more accurate estimations, and asymptotically as an infinite number of sensors are used the estimations converge to the target’s real position. We shall present the conditions under which these results are true for any p.d.f.s representing sensing models. Then we illustrate the results in the Gaussian case.

The belief prediction step plays the role of accelerating the convergence of estimations to a target’s real position as more sensors are used to detect the target. It does so by predicting an estimation that is close to the target’s real position. However, our theoretical results also indicate that it is also possible that belief prediction may decelerate the convergence if the estimation provided by belief prediction is too far from the target’s real position.

We note that our results can be directly applied to the original Markov location approach where a target’s motion model is given since MHML is a generalization of it. In the rest of this section we present these results in detail. We examine the belief correction step first and then move on to the belief prediction step.

6.2 Belief Correction

We are interested in how the belief at a target’s real position will change after the belief correction step. Our results rest on an average-case analysis.

6.2.1 AVERAGE-CASE ANALYSIS

Given a set of sensor readings $\{r_i\}$ and sensing models $\{p_{il}\}$, the belief correction step computes the next belief b_+ about a target from the current belief b using Equation (3). Due to the probabilistic nature of sensor readings, even if both detecting sensors’ locations and the target’s location l^* are fixed, the set of received sensor readings varies from time to time.

To facilitate our analysis, we consider how beliefs evolve in the average sense. Specifically, we consider how b_+ evolves from b as sensor readings $\{\mu_{il^*}\}$ are collected where μ_{il^*} is the mean of the p.d.f. determined by the location of sensor i and the target’s location l^* . To support this average case analysis, we use \bar{b}_+ and \bar{b} to respectively denote beliefs b_+ and b . With these notations, we rewrite Equation (3) as

$$\bar{b}_+(l) = \frac{\prod_i p_{il}(\mu_{il^*}) \bar{b}(l)}{\sum_l \prod_i p_{il}(\mu_{il^*}) \bar{b}(l)} \quad (4)$$

where $p_{il}(\mu_{il^*})$, specified by the sensing model of sensor i , is the probability of sensor i collecting reading μ_{il^*} if the target's position is l . In the rest of the paper, when we talk about belief change (increase or decrease), we mean it in the average sense.

For obvious reasons, we represent beliefs by vectors, all of which have the same dimension as the size of the grid. The beliefs \bar{b}_+ and \bar{b} are related by a vector transformation. To simplify our analysis, we first define the transformation and study its properties.

6.2.2 k -TRANSFORMATION

Definition 1 *Let u and k be nonnegative vectors. The k -transformation of u is a vector v whose component is defined as follows.*

$$v(i) = \frac{k(i)u(i)}{\sum_i k(i)u(i)}. \quad (5)$$

By definition, each component $v(i)$ of the k -transformation of a vector u is proportional to vector $u(i)$ and $k(i)$, or u is rescaled to v according to k . The following lemma shows how vector u changes its components after a k -transformation.

Lemma 1 *Let u and k be nonnegative vectors and $\sum_i u(i) = 1.0$. If v is the k -transformation of u ,*

- (1) *The ratio $v(i)/u(i)$ is constant for each i if and only if $k(i)$ is constant.*
- (2) *If $k(i) \neq k(j)$ for a pair (i, j) , then there exists an l such that $v(l) > u(l)$.*
- (3) *If $k(i) \neq k(j)$ for a pair (i, j) and $u(l^*) \neq 0$ for $l^* = \arg \max_i k(i)$, then $v(l^*) > u(l^*)$.*

Proof: See Appendix A. □

This lemma is useful in comparing the k -transformed vector v and the original vector u if the sum of $u(i)$ is 1.0. The first conclusion means that none of the components of vector u increases if and only if $k(i)$ is a constant for each i . If k is not a constant vector, the second conclusion means that some components increase after the transformation, which also implies that some other components must decrease, and the third means that such an increasing component is the one with the largest $k(i)$.

6.2.3 EFFECT OF INTEGRATING SENSOR READINGS

In belief correction, the belief \bar{b}_+ is a k -transformation of belief \bar{b} , where $k(l)$ is a multiplication of a series of probabilities, i.e. $\prod_i p_{il}(\mu_{il^*})$. The following theorem presents a condition under which the belief at the target's real position will be raised after belief correction. It is a direct application of Lemma 1.

Theorem 1 *If (1) $\bar{b}(l^*) > 0$, and (2) $\prod_i p_{il^*}(\mu_{il^*}) \geq \prod_i p_{il}(\mu_{il^*})$ for each l and the strict inequality holds for at least one l , then $\bar{b}_+(l^*) > \bar{b}(l^*)$.*

Proof: See Appendix A. □

This theorem specifies a sufficient condition ensuring belief increase at a target's real position. It can be paraphrased as follows. If (1) belief \bar{b} has a nonzero belief at the target's real position, and (2) the probability that sensors collect readings $\{\mu_{il^*}\}$ at l^* is greater than that at any other location, then the belief at l^* will increase after belief correction. We note that the first condition is necessary to ensure belief increase. If belief \bar{b} has a zero belief at the target's real position, so does the corrected belief \bar{b}_+ . This follows from the belief correction equation. Verbally, belief correction can never increase (change) the belief at a location if the belief at that location is zero.

The second condition of Theorem 1 is collective in the sense that it is given with respect to all detecting sensors. A stronger condition with respect to individual sensors is that $p_{il^*}(\mu_{il^*}) \geq p_{il}(\mu_{il^*})$ for each sensor i and any location l , i.e. the probability of sensor i collecting μ_{il^*} at l^* is greater than that at any other locations. For convenience, if sensor i has this property for any l^* , it is called a *legitimate* sensor. Therefore, if all detecting sensors are legitimate, belief increase at l^* is guaranteed after belief correction.

Corollary 1 *If (1) $\bar{b}(l^*) > 0$, and (2) $p_{il^*}(\mu_{il^*}) \geq p_{il}(\mu_{il^*})$ for each (i, l) pair and the strict inequality holds for at least one pair, then $\bar{b}_+(l^*) > \bar{b}(l^*)$.*

We give a few remarks on belief change after belief correction. First, as a special case of the above corollary, we note that if a legitimate sensor is used to detect a target, on average the belief at the target's position will increase. Second, it is also interesting to know when the belief at the target's position will decrease. We can give the following corollary, which is parallel to Corollary 1. It specifies a condition ensuring belief decrease at a target's real position. If (1) belief \bar{b} has a nonzero belief at the target's real position, and (2) the probability that each sensor i collects reading μ_{il^*} at l^* is smaller than that at any other location, then the belief at l^* will decrease after belief correction.

Corollary 2 *If (1) $\bar{b}(l^*) > 0$, and (2) $p_{il^*}(\mu_{il^*}) \leq p_{il}(\mu_{il^*})$ for each (i, l) pair and the strict inequality holds for at least one pair, then $\bar{b}_+(l^*) < \bar{b}(l^*)$.*

6.2.4 MONOTONICITY AND CONVERGENCE

We now turn to the monotonicity and convergence of MHML. The monotonicity is concerned with whether the estimations become more accurate when the number of detecting sensors increases. The convergence is concerned with whether the estimations converge to a target's real position as the number of sensors goes to infinity.

For monotonicity, we consider the case in which new sensors are added to the current group of detecting sensors. We prove that if a sensor is legitimate, i.e. the mean of all its possible readings has the largest probability to appear at the target's real position among all locations, adding it will increase the belief at the target's real position. We use $\{\mu_{1l^*}, \dots, \mu_{ml^*}\}$ and $\{\mu_{1l^*}, \dots, \mu_{m+1,l^*}\}$ to denote two sets of sensor readings. Note that the first set is produced by sensors from 1 to m and the second is by the same m sensors plus sensor $m+1$. Therefore the first group of sensors is a subset of the second.

Theorem 2 *Let \bar{b}_+^m and \bar{b}_+^{m+1} be updated beliefs from \bar{b} by Equation (4), respectively, using the readings $\{\mu_{1l^*}, \dots, \mu_{ml^*}\}$ and $\{\mu_{1l^*}, \dots, \mu_{m+1,l^*}\}$. If (1) $\bar{b}(l^*) > 0$, and (2)*

$p_{m+1,l^*}(\mu_{m+1,l^*}) \geq p_{m+1,l}(\mu_{m+1,l^*})$ for each location l and the strict inequality holds for at least one l , then $\bar{b}_+^{m+1}(l^*) > \bar{b}_+^m(l^*)$.

Proof: See Appendix A. □

For convergence, our conclusion is that the estimation converge to a target's real position if the difference between $p_{il}(\mu_{il^*})$ and $p_{il^*}(\mu_{il^*})$ for each sensor i and location l is greater than a predetermined nonnegative number ϵ .

Theorem 3 Let \bar{b}_+^m be updated belief from \bar{b} by using Equation (4) for the set of sensor readings $\{\mu_{1l^*}, \dots, \mu_{ml^*}\}$. If (1) $\bar{b}(l^*) > 0$, (2) there exists a positive number ϵ such that $p_{il}(\mu_{il^*}) \leq p_{il^*}(\mu_{il^*}) - \epsilon$ for any i and any location l other than l^* , then $\bar{b}_+^m(l^*)$ approaches 1.0 as m goes to infinity.

Proof: See Appendix A. □

It is worthwhile to mention that Condition (2) of the above theorem can be relaxed as follows. If there is a finite number of sensors such that $p_{il}(\mu_{il^*}) \geq p_{il^*}(\mu_{il^*}) - \epsilon$, the conclusion is still true. By limit theory, altering finite items in a sequence does not change its convergence. After the relaxation, the conclusion becomes more general.

6.2.5 DISCUSSIONS WITH GAUSSIAN CASE

We have proven some theoretical results for belief correction, which hold for p.d.f.s in any form. In this subsection, we illustrate the results in the Gaussian case. That is, we assume that sensing models for sensors and locations in the grid are represented by a Gaussian. We choose normal distribution in our discussion for several reasons. First of all, normal distribution is popular in approximating noise. Second, the property that a normal distribution is characterized by its mean and variance eases our discussions. Third, our experimental results are based on the normal distribution case.

For sensor i and location l , we denote the sensing model by p_{il} . In the Gaussian case, it is defined by the mean μ_{il} and the variance σ_{il} . The probability that sensor i receives a reading r , is $p_{il}(r)$, i.e., $\frac{1}{\sqrt{2\pi}\sigma_{il}}e^{-(r-\mu_{il})^2/(2\sigma_{il}^2)}$.

Theorem 1 presents the following collective condition ensuring that the belief at a target's actual position increases after belief correction.

$$\prod_i p_{il^*}(\mu_{il^*}) \geq \prod_i p_{il}(\mu_{il^*}) \text{ for any } l \in G.$$

If the p.d.f.s are replaced by Gaussians, the condition is instantiated as

$$\prod_{i=1}^m \frac{1}{\sqrt{2\pi}\sigma_{il^*}} \geq \prod_{i=1}^m \frac{1}{\sqrt{2\pi}\sigma_{il}} e^{-\frac{(\mu_{il^*}-\mu_{il})^2}{2\sigma_{il}^2}} \quad (6)$$

We are interested in finding when the above inequality is true. We consider two cases. First, if all p.d.f.s have the same variance, obviously the inequality is true for any sensor i and location l . Consequently, if the p.d.f.s for all sensors and all locations are Gaussian and they have the same variance, belief correction will never reduce the belief about the target at its real position. Moreover, if the variance is non-zero, the belief at the real position will strictly increase. Second, if the p.d.f.s have different variances, we need to compute both

sides of the above inequality. If the inequality is true, the belief at the real position will increase.

The left chart of Figure 11 illustrates the case in which only one sensor is used to detect a target. The thick curve is the p.d.f. determined by the sensor and the target's real position l^* , whereas other curves correspond to other locations. We assume that all p.d.f.s in the chart have the same variance. Under this assumption, it can be proven that Inequality (6) is true. Therefore, belief correction increases the belief at location l^* .

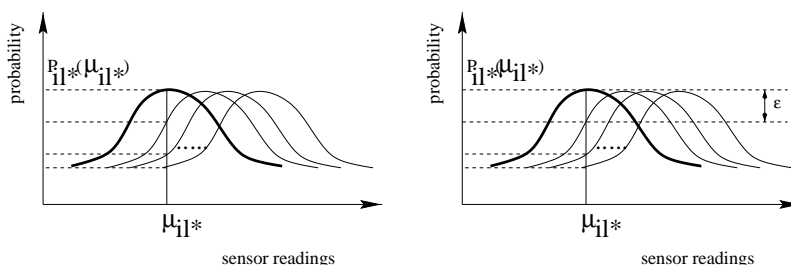


Figure 11: Illustrations for monotonicity and convergence theorems in Gaussian case

By the above analysis, if the p.d.f.s determined by a sensor and any locations are Gaussians and have the same variance for any l^* , the sensor is legitimate. In the multiple sensors case, if each sensor is legitimate, Corollary 1 means that belief correction will never reduce the belief about a target at its real position. Theorem 2 claims that adding a legitimate sensor leads to more accurate estimations on average.

Theorem 3 sets a more restrictive condition for convergence. It shows that the estimated belief converges to the real position when a large number of sensors are used. For sensor i , the theorem requires not only that it is legitimate, but also that probability $p_{il^*}(\mu_{il^*})$ differs from the probabilities at other locations by at least a nonnegative quantity ϵ , i.e., $p_{il^*}(\mu_{il^*}) \geq p_{il}(\mu_{il^*}) + \epsilon$. This is shown in the right chart of Figure 11, which is similar to the left one. To ensure the convergence, a large number of such sensors are needed.

6.3 Belief Prediction

We now consider the role of belief prediction, the other important step in MHML. This step is to predict the next belief of a target based on the current belief using a sequence of hypothesis generation, belief projection, belief scoring and belief selection. Its role is in fact complementary to that of belief correction under the conditions given in the preceding subsection. As the number of sensors increases, belief prediction accelerates the convergence of the estimations to a target's real position, and meanwhile belief correction guarantees the convergence.

One important question related to convergence is the convergent rate. In essence, if belief correction starts with a belief closer to the target's real position, the corrected belief

approaches to the real position faster as the number of sensors increases³. On the other hand, if the same number of legitimate sensors are used, the estimation from a starting location closer to the real position will be more accurate than that from a starting location farther away from the real position. In MHML, the starting location is provided by belief prediction. Therefore, the quality of belief prediction directly affects the performance of MHML. If the predicted belief is closer to the real position, belief correction should be able to compute a belief closer to the real position.

Figure 12 shows how belief prediction affects the performance of MHML. Suppose that the estimation of the target’s current position is l' . We compare two cases. First, if there is no belief prediction, belief correction takes l' as input. Second, if belief prediction predicts a belief and its induced location is l_2 , belief correction takes l_2 as input. Since l_2 is closer to the real l^* than l' , the corrected belief based on l_2 should be more accurate than that based on l' if the same set of sensors are used. On the other hand, if the predicted belief is l_1 , a location farther from l^* , the corrected belief based on l_1 should be less accurate than that based on l' .

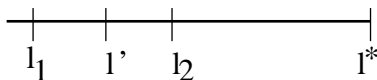


Figure 12: Role of belief prediction

In summary, the role of belief prediction can be twofold. A more accurate prediction will accelerate the convergence of estimated beliefs to a target’s real position when more sensors are used, whereas a less accurate prediction would decelerate the convergence. In this regard, a known motion model of a target can help improve the performance of MHML by providing belief correction with accurate predicted beliefs. In the case that there is no motion model, using more legitimate sensors can help improve the performance of MHML by calculating more accurate corrected beliefs. Thus, more sensors can be used to compensate for the lack of *a priori* knowledge, as we discussed in the previous subsection. Therefore, there exists a tradeoff between the knowledge of a target’s behavior and the resources of a tracking system.

7. Role of Motion Information

In this section, we provide empirical evidence to support our analysis on the relationship between belief prediction and belief correction in MHML. To this end, we will describe a degenerate MHML by removing its belief prediction part. For convenience, we refer to the degenerate MHML as *MHML0*. We will use MHML0 for three purposes. The first purpose is to demonstrate that as a degenerate version of MHML, MHML0 is able to localize a target without any knowledge of its motion. The second purpose is to demonstrate the monotonicity and convergence of belief correction in number of sensors used. This purpose

3. A belief b can be used to induce a location l in a few ways. One popular way is to take the average, i.e. $l = \sum_{l'} b(l') \cdot l'$. Another way is to take the location with the largest belief, i.e., $l = \arg \max_{l'} b(l')$. When we say “a belief is closer to or farther away from ...”, we mean the location induced by the belief.

is fulfilled by showing these properties in MHML0 and using the fact that MHML0 and MHML have the belief correction steps in common. The third purpose is to demonstrate the critical importance of the dynamically generated motion information in MHML. This purpose is fulfilled by showing the better performance of MHML than that of MHML0 and using the fact that only MHML has belief prediction.

7.1 MHML without Belief Prediction

MHML0 is a very simple algorithm of its own for localization. Its input and output are the same as in MHML. MHML0 corrects the input belief using only Equation (3). The corrected belief is used as the estimation of a target’s position. The pseudocode of one step localization *Localize0* for MHML0 is given in Table 2. At any step, the algorithm takes a belief b , a set $\{p_{ii}\}$ of sensing models, and a set $\{r_i\}$ of sensor readings. It computes a new belief b_+ by belief correction. Belief b_+ is the estimated belief of a target position.

Localize0

input : $b, \{p_{ii}\}$ and $\{r_i\}$

output : b_+

1. $b_+ \leftarrow \text{BeliefCorrection}(b, \{p_{ii}\}, \{r_i\})$
2. **Return** b_+

Table 2: MHML without belief prediction

A key question to MHML0 is how to choose the input belief, i.e., belief b in Table 2, at a step. Following MHML, a simple way for MHML0 is to choose the estimated belief from the previous step. In the following, we show that this is not appropriate for MHML0. We specifically show that very often this choosing method renders the estimated belief to be zero at a target’s real position. Let l^* and l_+^* be the target’s real positions at the current and next steps, which may be different since the target is moving. Let the estimated belief at the current step be b . If $b(l_+^*) = 0.0$, which is true if the previous estimation is accurate, then $b_+(l_+^*) = 0.0$ by belief correction equation⁴. The belief b_+ cannot be a good estimation of l_+^* since its belief is zero at the actual real position l_+^* . To avoid this problem, in our experiments, we set the input belief in MHML0 to be a uniform distribution at any step.

Before presenting empirical results, we note that as a degenerate version of MHML, MHML0 itself is able to localize a target. The prominent feature of MHML0 is that it never uses any motion information, just like the triangulation and NNS algorithms. This also makes MHML0 able to localize a target even without any knowledge of the target’s motion.

4. $b(l_+^*) = 0.0$ implies $b_+(l_+^*) = 0.0$. Note that this has nothing to do with the number of sensors. Therefore, using more sensors cannot improve the accuracy of the estimations. There is no convergence guarantee even if sensors are legitimate. In fact, that $b(l_+^*) = 0.0$ violates the first condition of theorems and corollaries in Subsection 6.2.

7.2 Performances of MHML without Belief Prediction

To demonstrate the performance of MHML0, we run it by adding more sensors. We report the results from a directed target with a variable speed and sensors using a 10-piece model (the right chart in Figure 5). The other experimental conditions remain the same as before.

Figure 13 shows the tracked and the real trajectories when 3, 5, 10 and 20 sensors are used. When 3 sensors are used, the tracked trajectory is completely out of the area of the real trajectory. When 5 sensors are used, the tracked trajectory partially intersects with the real, but with a very low estimation accuracy. The tracked trajectory forms a sketch of the real when 10 sensors are used. As the number of sensors increases to 20, the tracked trajectory is very close to the real, i.e., the estimated trajectory converges to the real.

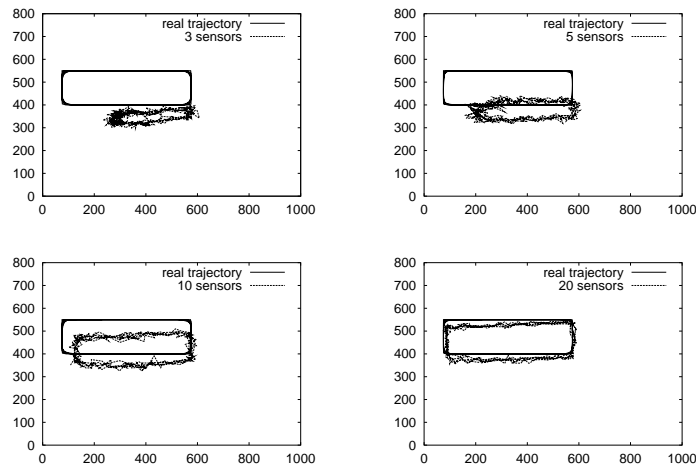


Figure 13: Tracked trajectories by MHML without belief prediction

From these results, it is evident that as more sensors are used, the tracked trajectory becomes very close to the real trajectory. Combined with the fact that MHML0 begins with a uniform belief at each step, it is clear that belief correction has the monotonicity and convergence properties. Since MHML and MHML0 have belief correction in common, this implies that belief correction in HMML has these two properties.

Due to the monotonous and convergent property, MHML0 has a value of interest of its own. As a degenerate version of MHML, MHML0 is able to localize a target without using any knowledge of its motion. Moreover, as more legitimate sensors are used, the estimation accuracy increases. Compared to MHML, MHML0 is computationally efficient since there is no belief prediction. If a tracking system has plentiful resources and is required to respond in real time at each step, MHML0 is an attractive choice.

7.3 Role of Motion Information

To understand the role of a target’s motion information, we further compare MHML and MHML0 using the same target and the same setups as in previous subsection. We run MHML with 3, 5, 10 and 20 sensors. The tracked trajectories are drawn in Figure 14.

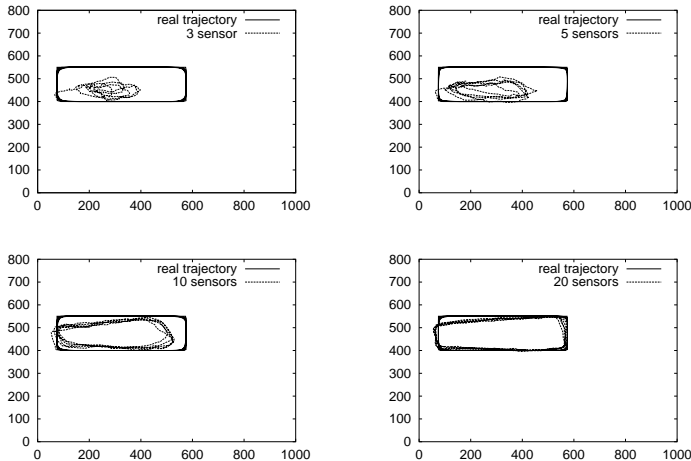


Figure 14: Tracked trajectories by MHML with belief prediction

Comparing the tracked trajectories in Figure 14 with those in Figure 13, we find that the trajectory tracked by MHML is closer to the real trajectory than that by MHML0 when the same number of sensors are used. This is especially true when the number of sensors is less than 10. This shows that (estimated) motion information in belief prediction is able to improve the estimation accuracy of MHML. As discussed earlier, belief prediction achieves this improvement by predicting a belief closer to the target’s real position and using the improved belief to drive the belief correction step. Additionally, we find that the improvements by belief prediction become insignificant when the number of sensors is greater than 10. The explanation follows. When the number of sensors is sufficiently large, MHML0 itself is able to accurately localize a target. There is not much room for improvements from belief prediction.

It is also of interest to observe the interactions between belief prediction and belief correction. If the multiple-hypotheses method is effective in predicting a target’s position, the improved estimation accuracy is “propagated” to the next localization step through belief correction. Consequently, it is expected that the overall estimation error of MHML should become smaller over steps. This can be seen from the plotted distance errors of MHML in Figure 15. In comparison, MHML0 does not have this property since at any step the input belief is set to a uniform distribution. Even if the estimation at the current step is very accurate, it cannot affect the estimation at the next step.

Finally, as mentioned in Subsection 5.3, noisy sensing models degrade the performance of MHML. Here we observe that the performance degradation of MHML using a noisy 10

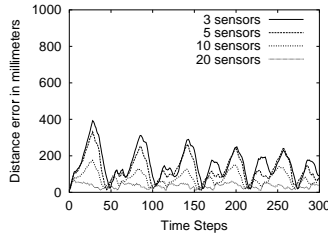


Figure 15: Distance errors of MHML when varying the sensor numbers

pieces model can be compensated by using more sensors, and that there exists a tradeoff between estimation quality and the amount of available resources.

8. Related Work

Many previous works are related to and have influenced this research. We now discuss them under the categories of localization problems, localization approaches, localization and motion models, and sensory information integration.

8.1 Localization Problems

The general localization problem has attracted a great deal of interest in diverse areas such as automatic control, robot navigation, military surveillance, context-aware systems, mobile computation, networking and MEMS-related applications. Different application domains make different assumptions on sensing or perceptual models as well as motion models of targets.

Most papers in automatic control consider state estimations in multi-sensor/multi-target environments (Bar-Shalom, 1978; Reid, 1979), where motion models, which are often referred to as *process models*, and sensing models are stochastic. However, the correspondence between collected observations and targets is unknown, which has been noticeably the main focus of a rich body of papers (Bar-Shalom, 1978; Reid, 1979).

In mobile robotics domains, one fundamental goal is to make a robot be capable of self-localizing, i.e., determining its own location in an environment. In one version of the problem, the robot has a probabilistic action model and a probabilistic perceptual model (Fox et al., 1999). The action model describes how the odometry readings reflect the robot’s real behavior. The perceptual model describes how the robot receives the landmark-based information such as junctions and crosses. A more challenging version of the mobile robot localization problem is called *simultaneous localization and mapping* (SLAM), where the knowledge of maps such as the landmark information is unknown (Thrun et al., 1998; Dissanayake, Newman, Clark, Durrant-Whyte, & Csorba, 2001; Montemerlo, Thrun, Koller, & Wegbreit, 2002). The map need to be established as the robot navigates within an environment.

Our research is motivated by multi-sensor/multi-target application domains where the knowledge of targets’ behaviors is unknown or limited, and sensors are erroneous and unre-

liable (Hill, 2000; Klemmer et al., 2000). Our problem differs from problems in automatic control in that (1) the correspondence is known between sensor readings and targets, and (2) no explicit motion model is given or required. This latter difference is also where our problem differs from the problems in robot navigation domain and the SLAM problem, since the effects of a robot’s control action are known *a priori* and can be used to derive a (stochastic) motion model. Additionally, our problem assumes a sensing model as in the navigation problem.

8.2 Localization Approaches

The existing localization approaches can be roughly categorized into three classes. The first class includes estimation-theoretical approaches such as Kalman filtering (KF) and its extensions and variants (Maybeck, 1979). This class has been surveyed in Subsection 3.3.

The second class of localization approaches includes Markov localization (ML) and its variants (Fox et al., 1999; Simmons & Koenig, 1995; Kaelbling, Littman, & Cassandra, 1998). ML approaches use a probability distribution to represent a target’s position estimation. This representation also makes robots be able to recover from localization errors. ML approaches require both motion models and sensing models. A localization procedure consists of belief projection, which projects beliefs using given motion models, and belief correction, which corrects the projected beliefs using sensor readings and a sensing model. In principle, ML approaches do not impose the linearity requirements on motion and sensing models. This feature renders ML approaches applicable to some problems unsuitable for KFs. For these problems, if KFs are used, targets’ motional and sensors’ perceptual models need to be linearized.

Finally, the third class of localization approaches do not stick to the rigorous statistical formalism but instead focus on pursuing numerical and computational solutions. A typical one is sample-based approach, also called *particle filter* (Arulampalam & Ristic, 2000; Thrun et al., 2001; Arulampalam et al., 2002). This class of approaches has gained increasing attention lately, because they provide a valuable and powerful tool for making tradeoffs between overall system performance and required computational cost.

8.3 Localization and Motion Models

To tackle the problem of no motion model, we adopt a multiple-hypothesis (motion model) approach. We dynamically generate a set of estimated motion models, each of which is a “guess” of a target’s next position. A likelihood function is used to choose the winning motion model. In the current form of our algorithm, only one hypothesis is the final winner. From step to step, the algorithm may switch one motion model to another. This is called a *model-switch* approach. In contrast to this strategy, other multi-model approaches, called *model-mixture* approaches, maintain a filter for each motion model and combine the individual results into an estimation of a target’s position. Model-mixture approaches differ from one another in the way they maintain certain filters and how they combine the results from individual filters. In (Blom & Bar-Shalom, 1988; Gustafsson, 2000), the authors consider maneuvering targets, whose behavior is described by a set of motion models. It is required that transition probabilities between these motion models be specified. Methods have been proposed to update the weights of estimations from individual filters when they

are combined. The approach in (Andersson, 1985; Bergman, 1999) keeps N models for localization and M models for possible maneuvers. The number of filters is kept constant by splitting the most probable filter and terminating the $M-1$ least probable filters. The N results from filters are combined to form a final estimation (Karlsson, 2002). The range-parameterized-extended-KF approach considers the estimation of range and velocity using only passive sensors (Peach, 1995; Arulampalam & Ristic, 2000; Karlsson & Gustafsson, 2001). Each filter is associated with a probability. If the probability associated with a filter is below a predefined level, it is removed from further computation.

One problem that our approach has to address is a potentially huge hypothesis space, especially in the case where the target’s behavior is irregular and abrupt. This can be alleviated by cutting less likely hypotheses or constraining the size of hypothesis space. A number of techniques have been proposed to cut less likely hypotheses. (Reid, 1979) uses a clustering technique to group sets of correspondences between targets and measurements. In visual tracking research, (Cox & Hingorani, 1996) follows (Reid, 1979) and uses a polynomial-time algorithm to determine the best n hypotheses for a constant n . Recently, (Schmitt et al., 2002) proposes techniques to ignore unlikely hypotheses and successfully applies a multiple-hypothesis approach in a robot soccer game. Another way to restrict a hypothesis space is to generate the space as small as possible. For this purpose, *a priori* knowledge or on-line learning and navigation techniques may be used (Sutton, 1984; Simmons & Koenig, 1995; Thrun et al., 1998).

8.4 Localization and Sensor Integration

Belief correction in MHML is essentially a sensor integration step in the sense that all sensor readings are incorporated to update the belief about a target’s location. It has been recognized that sensor integration is a prerequisite for advanced and complex applications of multi-sensor systems (Mutambara, 1998; Durrant-Whyte, 1988; Mutambara, 1998) and probabilistic approaches have been widely used in sensor integration. (Durrant-Whyte, 1988) discusses how individual sensors and a sensor system dynamically find a consistent consensus estimation of the environment. If individual sensors have inconsistent estimations from the sensor system, they allow their estimations to be changed. (Mutambara, 1998) contains sensor integration approaches based on Kalman filters. (Gustafsson, 2000) shows how to handle biased errors from individual sensors when multiple sensor readings are combined. (Faugeras & Ayache, 1986) employs a probabilistic model to fuse noisy line segment descriptions extracted from stereo visual images. Similar statistical models have been employed to estimate objects’ locations in (Porril, Pollard, & Mayhew, 1987).

In spite of much effort on developing sensor integration approaches, relatively little work analyzed the performance of integration approaches in terms of the number of sensors. In this paper, we show that if a sensor is legitimate, its addition to the detecting group statistically increases the estimation accuracy. We also mention another theoretical possibility, although we do not give a running example. Given a target’s real location, if a sensor picks the mean reading (of the p.d.f. determined by the sensor and the target’s position) with the smallest probability among all locations, it can be proven that adding it never increases the belief at a target’s real location.

9. Conclusions and Directions

In this paper, we proposed and studied a probabilistic approach to tracking moving targets in a distributed sensor network. The approach combines two sources of information, estimated target's motion and perceived sensor readings. This approach does not assume a motion model for targets. Rather, a target's motion is dynamically estimated. Empirical studies show that it works significantly better than previous approaches using only sensory information. We examined at a one-step basis how integrating sensing information changes the belief about a target on its real position, how the algorithm performance varies with the number of detecting sensors, and asymptotic behavior of the algorithm in number of sensors. We presented conditions under which integrating sensors' readings increases the belief about a target at its real position, more detecting sensors lead to more accurate results and the estimations converge to a target's real location as more sensors are used. Based on these results, we demonstrated that belief prediction and belief correction play different but complementary roles in the algorithm. Belief correction guarantees the convergence of estimated beliefs to the target's real position as more sensors are used, and meanwhile belief prediction accelerates the convergence by providing belief correction with a predicted belief close to the target's real position.

Several directions deserve further investigation in the future. Firstly, in this paper we consider the monotonicity and convergence issues at the basis of one step localization. One important question is whether these properties preserve over multiple steps. The difficulty is again the unavailability of motion models. If the motion model of a target is known in some nondeterministic form, it is possible to study the monotonicity and convergence issues in a statistical sense. Secondly, we assume that the sensors are stationary. One interesting direction would be to analyze the performance of the algorithm if detecting sensors themselves can move or follow the target. Thirdly, how to reduce the size of a hypothesis space is an important issue. In its current form, the hypotheses were generated through an enumeration. How to adaptively generate hypotheses is of theoretical interest and practical importance. By generating hypotheses adaptively, we expect the size of the space to be reduced and computation to be more focused on the most relevant hypotheses. In this regard, online learning algorithms in navigation research are useful. Finally, in a distributed sensor network, the localization task often needs to be real-time. This requirement is contradictory to the fact that micro-sensors have only limited information processing capability. The probabilistic algorithm described in this paper is computationally intensive, though in the text we have suggested steps for efficient implementations. Fortunately, the algorithm can be readily implemented in a particle-filter version (Arulampalam et al., 2002). This implementation is expected to be computationally cheap and obtain good-quality tracking results, as it suggested in similar studies (Thrun et al., 2001).

Acknowledgments

This research was supported in part by NSF grants IIS-0196057 and ITR/EIA-0113618, and in part by DARPA Cooperative Agreements F30602-00-2-0531 and F33615-01-C-1897. Special thanks go to Zhidong Deng for experiments on the micro sensors and sensing models used in this paper and for discussions on Kalman filter approaches. Thanks also to Sharlee Climer for reading a draft.

Appendix A. Proofs

Lemma 1 *Let u and k be nonnegative vectors and $\sum_i u(i) = 1.0$. If v is the k -transformation of u ,*

- (1) *The ratio $v(i)/u(i)$ is constant for each i if and only if $k(i)$ is constant.*
- (2) *If $k(i) \neq k(j)$ for a pair (i, j) , then there exists an l such that $v(l) > u(l)$.*
- (3) *If $k(i) \neq k(j)$ for a pair (i, j) and $u(l^*) \neq 0$ for $l^* = \arg \max_i k(i)$, then $v(l^*) > u(l^*)$.*

Proof: (1) This follows from that $\frac{v(i)/u(i)}{v(j)/u(j)} = \frac{k(i)}{k(j)}$.

(2) If $k(i) \neq k(j)$, then $\frac{v(i)}{u(i)} \neq \frac{v(j)}{u(j)}$. Thus either $v(i) \neq u(i)$ or $v(j) \neq u(j)$. Without loss of generality, let $v(i) \neq u(i)$. There are two cases.

(i) $v(i) > u(i)$. Let $l = i$. The conclusion is true.

(ii) $v(i) < u(i)$. By condition, $\sum_i u(i) = 1.0$. On the other hand, it can be verified that $\sum_i v(i) = 1.0$ by the definition of k -transformation. Therefore $\sum_i v(i) = \sum_i u(i)$. The decrease of v 's component i implies the increase of other components. Such an l exists.

(3) By condition, component l^* gains the largest increase. By (2), there exists at least one component l such that $v(l) > u(l)$. Therefore $v(l^*) > u(l^*)$. \square

Theorem 1 *If (1) $\bar{b}(l^*) > 0$, and (2) $\Pi_i p_{il^*}(\mu_{il^*}) \geq \Pi_i p_{il}(\mu_{il^*})$ for each l and the strict inequality holds for at least one l , then $\bar{b}_+(l^*) > \bar{b}(l^*)$.*

Proof: The belief \bar{b}_+ is a k -transformation of belief \bar{b} where $k(l)$ is substituted by $\Pi_i p_{il}(\mu_{il^*})$. By the condition $\Pi_i p_{il^*}(\mu_{il^*}) \geq \Pi_i p_{il}(\mu_{il^*})$ for each location l , and the inequality holds for at least one location l , the component l^* gets the largest increase. By (3) of Lemma 1, $\bar{b}_+(l^*) > \bar{b}(l^*)$. \square

Theorem 2 *Let \bar{b}_+^m and \bar{b}_+^{m+1} be updated beliefs from \bar{b} by Equation (4), respectively, using the readings $\{\mu_{1l^*}, \dots, \mu_{ml^*}\}$ and $\{\mu_{1l^*}, \dots, \mu_{m+1,l^*}\}$. If (1) $\bar{b}(l^*) > 0$, and (2) $p_{m+1,l^*}(\mu_{m+1,l^*}) \geq p_{m+1,l}(\mu_{m+1,l^*})$ for each location l and the strict inequality holds for at least one l , then $\bar{b}_+^{m+1}(l^*) > \bar{b}_+^m(l^*)$.*

Proof: Since $\bar{b}_+^m(l) = \frac{1}{c_1} \Pi_{i=1}^m p_{il}(\mu_{il^*}) \bar{b}(l)$ and $\bar{b}_+^{m+1}(l) = \frac{1}{c_2} \Pi_{i=1}^{m+1} p_{il}(\mu_{il^*}) \bar{b}(l)$ where $c_1 = \sum_l \Pi_{i=1}^m p_{il}(\mu_{il^*}) \bar{b}(l)$ and $c_2 = \sum_l \Pi_{i=1}^{m+1} p_{il}(\mu_{il^*}) \bar{b}(l)$,

$$\bar{b}_+^{m+1}(l) = \frac{c_1}{c_2} p_{m+1,l}(\mu_{m+1,l^*}) \bar{b}_+^m(l).$$

It can be verified that

$$c_2 = \sum_l c_1 p_{m+1,l}(\mu_{m+1,l^*}) \bar{b}_+^m(l).$$

Hence, \bar{b}_+^{m+1} is the k -transformation of \bar{b}_+^m where $k(l)$ is $c_1 p_{m+1,l}(\mu_{m+1,l^*})$. By the given condition and (3) of Lemma 1, $\bar{b}_+^{m+1}(l^*) > \bar{b}_+^m(l^*)$. \square

Theorem 3 Let \bar{b}_+^m be updated belief from \bar{b} by using Equation (4) for the set of sensor readings $\{\mu_{1l^*}, \dots, \mu_{ml^*}\}$. If (1) $\bar{b}(l^*) > 0$, and (2) there exists a positive number ϵ such that $p_{il}(\mu_{il^*}) \leq p_{il^*}(\mu_{il^*}) - \epsilon$ for any i and any location l other than l^* , then $\bar{b}_+^m(l^*)$ approaches 1.0 as m goes to infinity.

Proof: Since $\bar{b}_+^m(l) = \frac{1}{c} \Pi_i p_{il}(\mu_{il^*}) \bar{b}(l)$ for $l \neq l^*$ and $\bar{b}_+^m(l^*) = \frac{1}{c} \Pi_i p_{il^*}(\mu_{il^*}) \bar{b}(l^*)$ where c is a normalizer, the following steps are derived.

$$\frac{\bar{b}_+^m(l)}{\bar{b}_+^m(l^*)} = \Pi_{i=1}^m \frac{p_{il}(\mu_{il^*})}{p_{il^*}(\mu_{il^*})} \cdot \frac{\bar{b}(l)}{\bar{b}(l^*)} = \Pi_{i=1}^m \frac{p_{il}(\mu_{il^*})}{p_{il^*}(\mu_{il^*})} \leq (1.0 - \epsilon_1)^m$$

where

$$\epsilon_1 = \frac{\epsilon}{p_{il^*}(\mu_{il^*})}.$$

As m goes to infinity, $(1.0 - \epsilon_1)^m$ approaches zero. On the other hand, $\sum_l \bar{b}_+^m(l) = 1.0$. Consequently, $\bar{b}_+^m(l^*)$ approaches 1.0 as m increases. \square

References

- Andersson, P. (1985). Adaptive forgetting in recursive identification through multiple models. *International Journal of Control*, 42(5).
- Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2), 174–188.
- Arulampalam, S., & Ristic, B. (2000). Comparison of the particle filter with range-parameterized and modified polar EKF's for angle-only tracking. In *Proceedings of SPIE, Signal and Data Processing of Small Target*.
- Bahl, P., & Padmanabhan, V. N. (2000). RADAR: An in-building RF-based user location and tracking system. In *INFOCOM (2)*, pp. 775–784.
- Bar-Shalom, Y. (1978). Tracking methods in a multitarget environment. *IEEE transactions on Automatic Control*, 23(4), 618–626.
- Bar-Shalom, Y. (Ed.). (1989). *Multitarget-multisensor tracking: advanced applications*. Artech House.
- Bergman, N. (1999). *Recursive Bayesian Estimation: navigation and tracking applications*. Ph.D. thesis, Linköping University. Dissertations No. 579.
- Blackman, S. S. (1989). Association and fusion of multiple sensor data. In Bar-Shalom, Y. (Ed.), *Multitarget-Multisensor tracking: advanced applications*, chap. 5, pp. 187–218. Artech House.
- Blom, H., & Bar-Shalom, Y. (1988). The interacting multiple model algorithm for systems with Markovian switching coefficients. *IEEE Transactions on Automatic Control*, 33(8), 780–783.
- Bulusu, N., Estrin, D., Girod, L., & Heidemann, J. (2001). Scalable coordination for wireless sensor networks: self-configuring localization systems. In *Proceedings of the 6th IEEE International Symposium on Communication Theory and Application*.

- Chen, G., & Kotz, D. (2000). A survey of context-aware mobile computing research. Tech. rep. 2000-381, Department of Computer Science, Dartmouth College.
- Cox, I. J., & Hingorani, S. L. (1996). An efficient implementation of Reid's multiple hypothesis tracking and its evaluation for the purpose of visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(2).
- Deng, Z., & Zhang, W. (2002). Localization and dynamic tracking using wireless networked sensors and multi-agent technology: first steps. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E00-A(1).
- Dissanayake, M. W., Newman, P., Clark, S., Durrant-Whyte, H. F., & Csorba, M. (2001). A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3), 229–241.
- Durrant-Whyte, H. F. (1988). Sensor models and multisensor integration. *The International Journal of Robotics Research*, 7(6), 97–113.
- Faugeras, O., & Ayache, N. (1986). Building visual maps by combining noisy stereo measurements. In *IEEE conference on Robotics and Automation*.
- Fox, D., Burgard, W., & Thrun, S. (1999). Markov localization for mobile robots in dynamic environments. *Journal of Artificial Intelligence Research*, 11, 391–427.
- Gustafsson, F. (2000). *Adaptive filtering and change detection*. John Wiley & Sons Ltd.
- Harter, A., Hopper, A., Steggle, P., Ward, A., & Webster, W. (1999). The anatomy of a context-aware application. In *Proceedings of the 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM)*.
- Hightower, J., & Borriello, G. (2001). A survey and taxonomy of location sensing systems for ubiquitous computing. Tech. rep. 01-08-03, Department of Computer Science and Engineering, University of Washington, Seattle, WA.
- Hill, J. (2000). A software architecture supporting networked sensors. Master's thesis, Computer Science Department, University of California at Berkeley.
- Kaelbling, L. P., Littman, M. L., & Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2).
- Karlsson, R. (2002). *Simulation Based Methods for Target Tracking*. Ph.D. thesis, Linköping University. Dissertations No. 930.
- Karlsson, R., & Gustafsson, F. (2001). Range estimation using angle-only target tracking with particle filters. In *Proceedings of the American Control Conference*, Vol. 5.
- Klemmer, S., Whitehouse, K., & Waterson, S. (2000). An empirical evaluation of TinyOS RF networking (and beyond...)..
- Maybeck, P. S. (1979). *Stochastic models, estimation and control*, Vol. I. Academic Press.
- Montemerlo, M., Thrun, S., Koller, D., & Wegbreit, B. (2002). FastSLAM: a factored solution to the simultaneous localization and mapping problem. In *in Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*.

- Moutarlier, P., & Chatila, R. (1989). Stochastic multi-sensory data fusion for mobile robot location and environment modeling. In *5th International Symposium on Robotics Research*.
- Mutambara, A. G. (1998). *Decentralized estimation and control for multisensor systems*. CRC Press.
- Peach, N. (1995). Bearings-only tracking using a set of range-parameterized extended Kalman filters. In *IEE Proceedings of Control Theory and Applications*, Vol. 142.
- Perkins, C. E. (2001). *Ad hoc Networking*. Addison Wesley.
- Porril, J., Pollard, S. B., & Mayhew, J. E. (1987). Optimal combination of multiple sensors including stereo vision. *Image and vision computing*, 5, 174–180.
- Reichl, H. (2001). Overview and development trends in the field of MEMS packaging. In *14th International Conference on Micro Electro Mechanical Systems*. invited talk.
- Reid, D. B. (1979). An algorithm for tracking multiple targets. *IEEE Transactions on Automatic Control*, 24(6), 843–854.
- Schmitt, T., Beetz, M., Hanek, R., & Buck, S. (2002). Watch their moves: applying probabilistic multiple object to autonomous robot soccer. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI)*.
- Simmons, R., & Koenig, S. (1995). Probabilistic robot navigation in partially observable environments. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1080–1087.
- Smith, S., Self, M., & Cheeseman, P. (1990). Estimating uncertain spatial relationships in robotics. In Cox, I. J., & Wilfong, G. T. (Eds.), *Autonomous robot vehicles*, pp. 167–193. Springer Verlag.
- Sutton, R. (1984). *Temporal credit assignment in reinforcement learning*. Ph.D. thesis, University of Massachusetts, Amherst, MA.
- Takeda, M. (2001). Applications of MEMS to industrial inspection. In *the 14th International Conference on Micro Electro Mechanical Systems*. invited talk.
- Thrun, S., Burgard, W., & Fox, D. (1998). A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning and Autonomous Robots (joint issue)*, 31(5).
- Thrun, S., Fox, D., Burgard, W., & Dellaert, F. (2001). Robust Monte Carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2), 99–141.
- Toh, C. (2001). *Ad Hoc Mobile Wireless Networks: Protocols and Systems*. Prentice Hall.
- Want, R., Hopper, A., Falcao, V., & Gibbons, J. (1992). The active badge location system. *ACM Transactions on Information Systems*, 10(1).
- Zhang, W., Deng, Z., Wang, G., Wittenburg, L., & Xing, Z. (2002). Distributed problem solving in sensor networks. In *Proceedings of AAMAS*. post paper.
- Zhou, R., & Hansen, E. (2001). An improved grid-based approximation algorithm for POMDPs. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI)*.