Voronoi Tracking: Location Estimation Using Sparse and Noisy Sensor Data

Lin Liao, Dieter Fox, Jeffrey Hightower, Henry Kautz, and Dirk Schulz

Deptartment of Computer Science & Engineering
University of Washington
Seattle, WA

Abstract

Tracking the activity of people in indoor environments has gained considerable attention in the robotics community over the last years. Most of the existing approaches are based on sensors which allow to accurately determine the locations of people but do not provide means to distinguish between different persons. In this paper we propose a novel approach to tracking moving objects and their identity using noisy, sparse information collected by id-sensors such as infrared and ultrasound badge systems. The key idea of our approach is to use particle filters to estimate the location of people on the Voronoi graph of the environment. By restricting particles to a graph, we make use of the inherent structure of indoor environments. The approach has two key advantages. First, it is by far more efficient and robust than unconstrained particle filters. Second, the Voronoi graph provides a natural discretization of human motion, which allows us to apply unsupervised learning techniques to derive typical motion patterns of the people in the environment. Experiments using a robot to collect ground-truth data indicate the superior performance of Voronoi tracking. Furthermore, we demonstrate that EM-based learning of behavior patterns can indeed increase the tracking performance of the approach.

1 Introduction

Over the last years, the estimation of the location of people in indoor environments has gained increased attention in the robotics community [7; 14; 10]. This is mainly due to the fact that knowledge about the position and the motion patterns of people can help mobile robots to better interact with people, as stated by [2]. Most existing approaches to people tracking rely on laser range-finders [7; 2; 14] or cameras [10]. A key advantage of these sensors is their location accuracy. Unfortunately, they do not provide information about the identity of people. Recently, especially the ubiquitous computing community has started to equip indoor environments with networks of sensors that are capable of providing information about a person's location and identity [9]. Such sensors, as infrared and ultrasound badge systems [17; 12], have the disadvantage that they provide only relatively coarse location information. In addition to being corrupted by noise, such sensors provide measurements at relatively low frame rates only [9].

Even though id-sensors do not allow accurate position estimation, they can be used to keep track of a person's location at a more abstract level such as which room or hallway she is in. Such discrete, abstract location information additionally provides an ideal representation for learning patterns in a person's long term behavior. Note that pattern discovery in continuous space trajectories often requires supervised learning methods [2].

Based on these observations, we introduce a novel approach to estimating the locations of people using sparse and noisy sensor data collected by id-sensors. The key idea of our approach is to track the locations of people on Voronoi graphs [5], which allow us to naturally represent typical human motion along the main axes of the free space. The estimated trajectories on the Voronoi graph help us to bridge the gap between continuous sensor data and discrete, abstract models of human motion behavior. In contrast to existing localization approaches using discrete, abstract representations [15; 5], our method provides accurate estimates along the continuous edges of the graph. We introduce a two-level model in which a particle filter uses a switching state space model for human motion, and the Voronoi graph guides the particles through the high-level transition model of the graph structure. We additionally show how to apply EM to learn typical motion patterns of humans in a completely unsupervised manner. The transition probabilities learned from real data significantly increase the tracking quality of the approach. Our experiments show that tracking on the Voronoi graph significantly outperforms particle filters in the complete state space (arbitrary position and orientation).

This paper is organized as follows. In the next section, we will derive Voronoi tracking starting from the general Bayes filter. Then, in Section 3, we show how to learn the parameters of the tracking model using expectation maximization. Before concluding in Section 5, we show experimental results supporting the superior performance of Voronoi tracking.

2 Voronoi Tracking

We define a Voronoi graph $G = (V, E)$ by a set V of vertices v_i and a set E of directed edges e_j. Figure 1(a) shows the Voronoi graph representing the Intel Research Lab Seattle, our indoor testing environment. Note that this graph results from manual pruning of the original Voronoi graph [5]and that for clarity only the undirected version of the graph is shown.

In the following we phrase the problem of location estimation on Voronoi graphs as a special case of Bayesian filtering, on which virtually all probabilistic location esti-

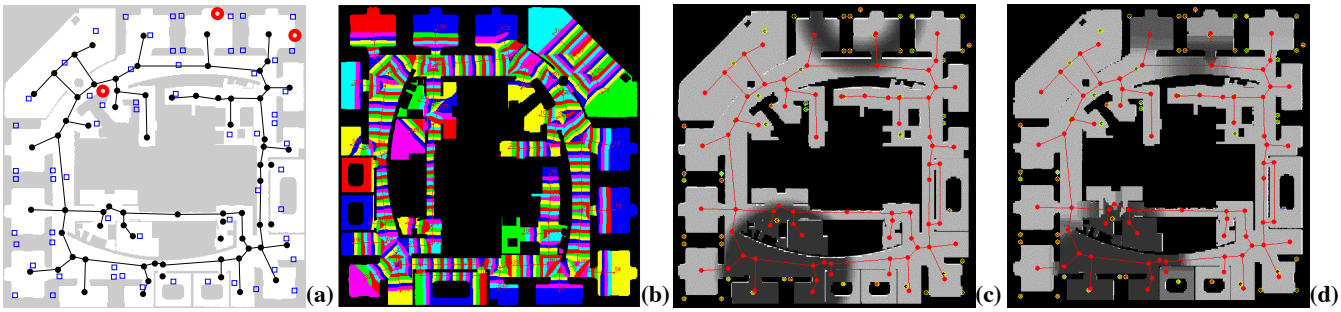

Figure 1: *Voronoi graphs for location estimation: (a) Indoor environment along with manually pruned Voronoi graph. Shown are also the positions of ultrasound Crickets (red) and infrared sensors (blue). (b) Patches used to compute likelihoods of sensor measurements. Likelihood of an ultra-sound cricket reading (c) in the full state space and (d) as projected onto the Voronoi graph patches.*

mation approaches are based. We will start by describing general Bayes filters for the full continuous state space, and we will then show how to project the different quantities of the Bayes filter onto the topological structure represented in a Voronoi graph.

2.1 Bayesian Filtering on a Voronoi Graph

Bayes filters estimate posterior distributions over the state x_t of a dynamical system conditioned on all sensor information collected so far:

$$p(x_t \,|\, z_{1:t}) \propto p(z_t \,|\, x_t) \int p(x_t \,|\, x_{t-1})\; p(x_{t-1} \,|\, z_{1:t-1}) dx_{t-1} \quad (1)$$

Here $z_{1:t}$ is the history of all sensor measurements obtained up to time t. In the context of location estimation, the state typically describes the position and velocity of the object in $\langle x, y, \theta \rangle$-space. The term $p(x_t \mid x_{t-1})$ is a probabilistic model of the object dynamics, and $p(z_t \mid x_t)$ describes the likelihood of making observation z_t given the location x_t and a map of the environment.

Let us now describe how to implement the recursive Bayes filter for Voronoi graphs. We represent the state x_t of an object by a triple $x = \langle e, d, m \rangle$, where e denotes the current edge on the graph, d indicates the distance of the object from the start vertex of edge e, and $m \in \{\text{stopped}, \text{moving}\}$ indicates the current motion state of the object. A straightforward implementation of the observation model for Voronoi graphs would be to simply compute the likelihood of observations for positions on the graph. However, this is not a valid approach since it does not consider the fact that the Voronoi graph projects the 3d state space onto the one-dimensional graph G. Hence, to compute the likelihood of an observation z given a position x on the graph, we have to integrate over all 3d positions projected onto x:

$$p(z|x) = \int_{\nu \in \mathcal{S}(x)} p(z \mid \nu)\; p(\nu \mid x)\; d\nu, \quad (2)$$

Here, $\mathcal{S}(x)$ denotes the set of all positions ν projected onto x. In the limit, $\mathcal{S}(x)$ are the states on the line from x to the closest objects defining the Voronoi graph. In our implementation of the sensor model, we discretize positions on the graph, which results in location *patches* $\mathcal{S}(x)$, as illustrated in Figure 1(b)–(d).

The motion model $p(x_t \mid x_{t-1})$ has to take into account that the objects are constrained to moving on the graph. Here, we adopt a switching state model approach based on the assumption that objects either do not move at all ($m_t = \text{stopped}$), or move ($m_t = \text{moving}$) [1] with a velocity v governed by a Gaussian process with mean v variance σ_v^2. More specifically, we have to consider motion along an edge of the Voronoi graph as well as motion from one edge to another. Let us first describe the more simple case of motion on one edge, which we denote $\hat{p}(x_t \mid x_{t-1})$:

$$\begin{aligned} \hat{p}(x_t \,|\, x_{t-1}) \;=\; & p(m_t = \text{move} \,|\, m_{t-1}) \cdot \mathcal{N}\left(\frac{x_t - x_{t-1}}{\Delta t}; v, \sigma_v\right) \\ & + p(m_t = \text{stopped} \,|\, m_{t-1}) \cdot \delta(x_t, x_{t-1}) \end{aligned} \quad (3)$$

Here, $\delta(x_t, x_{t-1})$ is the Dirac delta function which is 1, if $x_t = x_{t-1}$ and 0 otherwise. We assume that $p(m_t | m_{t-1})$, the switching between motion states, is learned from data (see Section 3). $x_t - x_{t-1}$ denotes the distance function on the Voronoi graph. It can be defined as

$$x_j - x_i \;=\; \begin{cases} d_j - d_i & \text{if } e_i = e_j \\ |e_i| - d_i + d_j & \text{otherwise} \end{cases} \quad (4)$$

The term $|e_i| - d_i + d_j$ computes the distance to the end of edge e_i plus the distance d_j on the next edge. For clarity, we restrict the definition to the case of motion between neighboring edges e_i and e_j, but it can easily be extended to the general case of multi-edge motion.

To compute the probability of motion from one edge to another, we assume that the Voronoi graph is annotated with transition probabilities $p(e_j|e_i)$, which describe the probability that the object transits to node e_j given that the previous node was e_i and an edge transition took place. Without prior knowledge, for example, this probability is distributed uniformly over all neighboring edges of e_i. Combining this model with the single edge motion model given in (3), we get the following motion model:

$$p(x_t \,|\, x_{t-1}) = \begin{cases} \hat{p}(x_t \mid x_{t-1}) & \text{if } e_t = e_{t-1} \\ p(e_t \,|\, e_{t-1})\; \hat{p}(x_t \mid x_{t-1}) & \text{if } e_t \neq e_{t-1} \end{cases} \quad (5)$$

[1] While this model might seem too simplistic, our experiments indicate that it yields good results and the extension to more complex motion models is straightforward.

This finalizes our description of how general state space Bayes filters can be projected onto Voronoi graphs. In the next section we will describe how to implement this graph-based model using particle filters.

2.2 Particle Filter Based Implementation

Particle filters provide a sample-based implementation of general Bayes filters [8; 6]. The key idea of particle filters is to represent posteriors over the state x_t by sets S_t of n weighted samples:

$$S_t = \{\langle x_t^{(i)}, w_t^{(i)}\rangle \mid i = 1, \ldots, n\}$$

Here each $x_t^{(i)}$ is a sample (or state), and the $w_t^{(i)}$ are non-negative numerical factors called *importance weights*, which sum up to one. Just like Kalman filters, particle filters apply the recursive Bayes filter update to estimate posteriors over the state space. The basic form of the particle filter updates the posterior according to the following sampling procedure, often referred to as sequential importance sampling with re-sampling (SISR, see also [6]):

Re-sampling: Draw with replacement a random sample $x_{t-1}^{(i)}$ from the sample set S_{t-1} according to the (discrete) distribution defined through the importance weights $w_{t-1}^{(i)}$.

Sampling: Use $x_{t-1}^{(i)}$ to sample $x_t^{(j)}$ from the distribution $p(x_t \mid x_{t-1})$. $x_t^{(j)}$ now represents the density given by the product $p(x_t \mid x_{t-1})p(x_{t-1} \mid z_{1:t-1})$. This density is the so-called *proposal distribution* used in the next step.

Importance sampling: Weight the sample $x_t^{(j)}$ by the importance weight $p(z_t \mid x_t^{(j)})$, the likelihood of the measurement z_t given the state $x_t^{(j)}$.

Each iteration of these three steps generates a sample drawn from the posterior density. After n iterations, the importance weights of the samples are normalized so that they sum up to 1. It can be shown that this procedure in fact approximates the Bayes filter update (1), using a sample-based representation [6].

The application of particle filters to location estimation on a Voronoi graph is rather straightforward. The resampling step does not have to be changed at all. Sampling the motion update has to be done according to (3)–(5). To do so, recall that the state x_{t-1} contains the position on the graph along with the motion state m_{t-1}. Let us begin with (3). We first sample the motion state m_t with probability proportional to $p(m_t \mid m_{t-1})$. If $m_t =$ moving, then we randomly draw the moved distance d from the Gaussian distribution given in (3). For this distance d, we have to determine whether the motion along the edge results in a transition to another edge. If not, then $d_t = d_{t-1} + d$ and $e_t = e_{t-1}$. Otherwise, $d_t = d - |e_{t-1}| + d_{t-1}$ and the next edge e_t is drawn with probability $p(e_t \mid e_{t-1})$. Otherwise, if the motion state $m_t =$ stopped, the position x_t is set to be x_{t-1}. After these sampling steps, the resulting states are distributed according to $p(x_t \mid x_{t-1})$. The importance sampling step of the particle filter is implemented by weighting each sample proportional to the projected observation likelihood as given in (2).

To summarize, we described how to perform recursive Bayesian filtering by projecting the general Bayes filter onto Voronoi graphs, followed by a sample-based implementation. In the next section, we will describe how to learn the motion patterns of individual people.

3 Parameter Learning

One of the key applications of graph-based location estimation is the collection of long-term data so as to learn behavior models of individual people. Learning parameters for a specific person not only increases the accuracy of tracking, but also allows us to understand different motion patterns for different people. As a first step in this direction, we will now describe how to learn the parameters of our Voronoi model using data collected by an object moving through the environment. An obvious difficulty here is that we have to learn the motion model even without knowledge of the people's real trajectories. One general approach of solving learning problems with missing features is the well known EM (Expectation-Maximization) algorithm [3; 13]. EM is an iterative algorithm which has an E-step and a M-step at each iteration.

3.1 E-Step:

In the E-step, we update the posterior distribution over the trajectories of the person and compute the expectation of the log-likelihood. We define:

$$\begin{aligned} & Q(\Theta, \Theta^{(i-1)}) \qquad (6) \\ &= E[\log p(z_{1:t}, x_{1:t} \mid \Theta) \mid z_{1:t}, \Theta^{(i-1)}] \\ &= \int_{x_{1:t}} \log p(z_{1:t}, x_{1:t} \mid \Theta) p(x_{1:t} \mid z_{1:t}, \Theta^{(i-1)}) dx_{1:t} \qquad (7) \end{aligned}$$

Here $x_{1:t}$ and $z_{1:t}$ are the states and observations, respectively. Θ are the parameters of the Voronoi graph-based model we want to estimate and $\Theta^{(i-1)}$ are the estimation thereof at the $i-1$-th iteration of the EM algorithm. The difficulty here is to estimate the posterior distribution over state trajectories $x_{1:t}$ given $z_{1:t}$ and $\Theta^{(i-1)}$. In our scenario, it is not possible to find a closed form solution to this estimation problem. Therefore we resort to approximate approaches. Observe that when we do particle filtering using the motion model with parameter $\Theta^{(i-1)}$, the particle distribution at each time t along with the history of particles is an approximation for $p(x_{1:t} \mid z_{1:t}, \Theta^{(i-1)})$. Therefore, (7) can be rewritten as

$$Q(\Theta, \Theta^{(i-1)}) \approx \frac{1}{m} \sum_{j=1}^{m} \log p(z_{1:t}, x_{1:t}^{(j)} | \Theta), \qquad (8)$$

where m is the number of particles and $x_{1:t}^{(j)}$ is the state history of the j-th particle. For simplicity, we assume that all the particles have equal weight, *i.e.* after they are resampled. It is straightforward to extend our derivation to the case of different weights.

Our approach is in fact a direct extension of the Monte Carlo EM algorithm [18]. The only difference is that we

allow particles to evolve with time. It has been shown that when m is large enough, Monte Carlo EM estimation converges to the theoretical EM estimation [11].

3.2 M-step:

The goal of the M-step is to maximize the expectation we computed in the E-step by updating the parameter estimations. From (8), we have:

$$\begin{aligned}\Theta^{(i)} &= \operatorname*{argmax}_{\Theta} Q(\Theta, \Theta^{(i-1)}) \\ &= \operatorname*{argmax}_{\Theta} \sum_{j=1}^{m} \log p(z_{1:t}, x_{1:t}^{(j)} \mid \Theta) \\ &= \operatorname*{argmax}_{\Theta} \sum_{j=1}^{m} (\log p(z_{1:t} | x_{1:t}^{(j)}) + \log p(x_{1:t}^{(j)} | \Theta)) \quad (9) \\ &= \operatorname*{argmax}_{\Theta} \sum_{j=1}^{m} \log p(x_{1:t}^{(j)} \mid \Theta) \quad (10)\end{aligned}$$

Here, (9) follows from the independence condition $p(z_{1:t} \mid x_{1:t}^{(j)}, \Theta) = p(z_{1:t} \mid x_{1:t}^{(j)})$, *i.e.* observations are independent of model parameters if the state trajectory is known. Directly evaluating (10) requires a particle smoothing operation which usually requires storing the history of each particle. Therefore it is often too expensive. In practice, we avoid direct smoothing by performing a forward particle filtering and backward particle filtering step. Then we multiply the two distributions at corresponding time slices, which corresponds to the Baum-Welch algorithm widely used for Hidden Markov Models [13]. This turns out to be very efficient since at each time we only need to save the aggregate information for each edge, not for each particle.

3.3 Implementation

Applied to our Voronoi tracking, the EM algorithm is implemented as follows. As described in Section 2, the parameters Θ of the model consist of the transition probabilities $p(e_i \mid e_j)$ on the Voronoi graph, the switching parameters $p(m_t \mid m_{t-1})$ of the motion model, and the Gaussian motion parameters (v, σ_v^2).[2] For the first E-step, we initialize $\theta^{(0)}$ with some reasonable values using background knowledge of typical human motion and a uniform distribution for the outgoing edges at each vertex of the Voronoi graph. Then we collect data in the environment and estimate the position of the person using the initial model. Time is discretized in intervals of equal size Δt. After each time step, we determine $e_{ij}(t)$, the number of particles that move from one edge e_i to another edge e_j. We also generate counts for $m_{ijk}(t)$, the number of particles on edge e_i that switch from one motion state m_j into another other motion state m_k. The reason for learning different switching models for each edge is that we want to be able to determine where a person typically stops for extended periods of time. To estimate the parameters of the Gaussian motion model, we simply have to store the velocities of all samples that are in the "moving" state. In the backward pass, the same values are computed when localizing the person backward in time, *i.e.* we proceed backwards through the data log. The final values of the E-step are then given by multiplication of the estimates of the forward and backward pass at each time slice (see [13] for details). To maximize the parameter set $\Theta^{(1)}$, the M-step essentially converts the frequency counts obtained in the E-step into probabilities. The parameters of the Gaussian motion model are the mean and variance of the velocity values. In the next iteration of EM, these new parameters are used to re-estimate the expectations using forward backward localization. The algorithm stops if $\Theta^{(i)}$ and $\Theta^{(i-1)}$ are close enough.

[2] We do not learn the observation likelihood (2), since it can be extracted directly from the general model of the sensors.

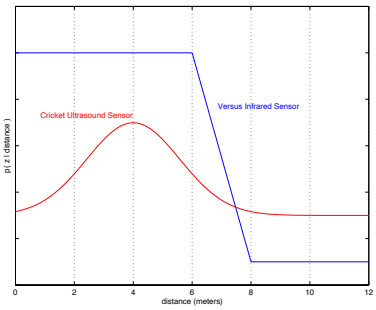

Figure 2: *Sensor model of ultrasound crickets and infrared badge system. The x-axis represents the distance from the detecting infrared sensor and ultrasound sensor (4m ultrasound sensor reading), respectively. The y-axis gives the likelihood for the different distances from the sensor.*

4 Experiments

We evaluated the performance of the Voronoi tracking (VT) approach based on data recorded at the Intel Research Lab Seattle. This office environment is equipped with two different kinds of id sensors: 73 Versus infrared receivers which solely provide id information, and three Cricket ultrasound receivers, which provide identity and distance estimates [17; 12]. Figure 2 shows a model of the uncertainty of the two sensor types. Note that both sensors additionally suffer from a high rate of false-negative readings, *i.e.* they fail to detect the presence of a person.

To generate data for which ground truth is available, we equipped a Pioneer IIDX robot with two Versus badges, a Cricket beacon, and additionally with a Sick laser range-finder. Our experiment is based on a log of sensor measurements received while driving the robot through the environment, at an average velocity of 30 cm/s. During the experiment, we only stopped the robot at designated resting places. The laser range-finder allowed us to accurately estimate the path of the robot using the map shown in Figure 1(a). The complete data log was split into a training set of 25 minutes and a test set of 10 minutes of robot motion. Figure 3(a) shows the robot trajectory used for training.

As an initial test, we determined the average localization error when using un-trained VT *vs.* a particle filter (PF) representing locations in the complete free space of the en-

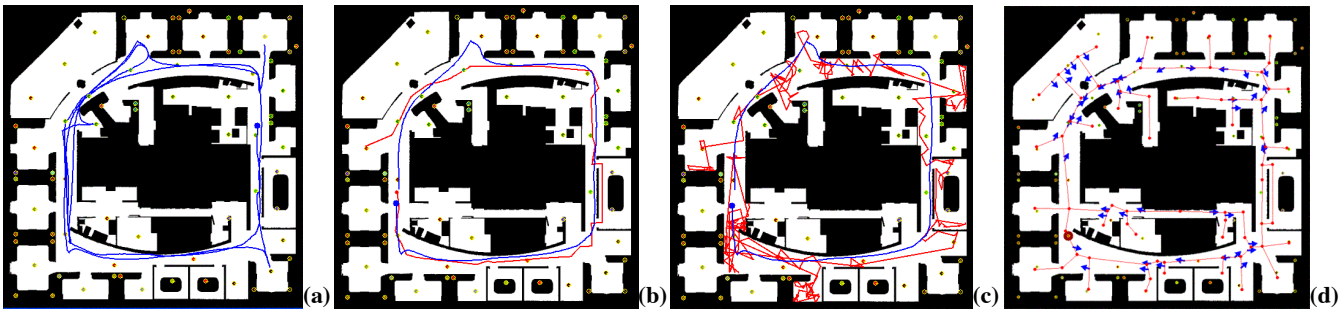

Figure 3: *(a) Trajectory of the robot during a 25 minute period of training data collection. True path and most likely path as estimated using (b) Voronoi tracking and (c) original particle filters. (d) Transition model learned using EM. Shown are only those transitions for which the probability is above 0.7*

vironment. The resulting error on the test data was 2.586m for VT and 3.230m for PF. This result is extremely encouraging since it indicates that the projection onto the Voronoi graph does not only result in a good approximation of the PF, but yields *better* performance than the PF. Figure 3(b) and (c) show the most likely trajectories using VT and PF, respectively [3]. The graphs clearly demonstrate the superior performance of the VT technique.

Next, we evaluated the learning performance of VT. To do so, we trained the VT model using the EM algorithm described in Section 3 on the training set. The results are summarized in the table below.

EM Iteration	Voronoi Accuracy	Avg. change
before learning	2.586	
1	1.888	0.235
2	1.787	0.083
3	1.530	0.075
4	1.538	0.054

The middle row gives the average error over the training set for the different iterations of EM. As can be seen, the algorithm converges after only 3 iterations. Furthermore, learning reduces the error on the test set by 60%. This shows that the Voronoi graph is able to extract the correct motion patterns from the training data. This fact is supported by Figure 3(d), which shows the transition model of the Voronoi graph after learning (shown are only transitions with probability above 70%). The plot demonstrates that the VT model learned transitions along the main path of the robot.

Using the trained VT model, we additionally evaluated the tracking performance for different sample set sizes. The results are given in Figure 4. For comparison, we added the average error for the original PF using the same number of samples. It becomes clear that VT makes very efficient use of particles and its performance degrades only below 125 particles.

5 Conclusions and Future Work

We have presented a novel approach to tracking the location of people in indoor environments. The technique is

[3]These paths were generated from the trajectories of the most likely particle at the end of the run.

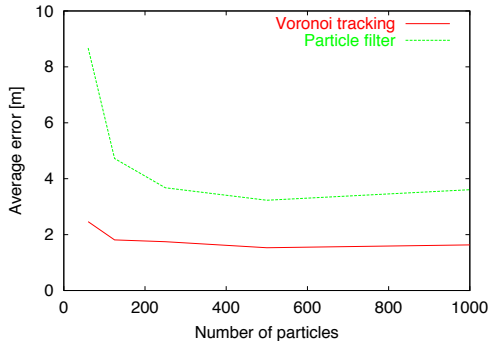

Figure 4: *Average localization error for different numbers of samples.*

able to robustly estimate a person's location even when using only sparse, noisy information provided by id-sensors. The key idea of our approach is to use a particle filter that is projected onto a Voronoi graph of the environment. The resulting model is a two-level technique, where, on the lower level, the particle filter estimates the location of people along the continuous edges of the graph. At the next level, the particles generate transitions on the discrete Voronoi graph. These two levels are highly connected, since transitions on the Voronoi graph help to predict the motion of particles through the graph structure. On the other hand, particle motion helps to estimate and learn the discrete transition model of the graph. Learning is achieved by EM, which concurrently estimates the model parameters at both levels.

Using data collected by a mobile robot, we demonstrate that our approach has two key advantages. First, it is by far more efficient and robust than particle filters which estimate the location of people in the complete free space of an environment. Second, the Voronoi graph provides a natural discretization of human motion, which allows us to learn typical motion patterns using expectation maximization. We show that the EM-based learning of behavior patterns indeed increases the tracking performance of the approach.

At this point, it should be noted that our work is related to tracking a person's or vehicle's location using outdoor GPS data. Most state of the art GPS tracking systems solely project GPS readings onto graph-based maps describing

streets, walkways *etc*. Recently, several researchers suggested using Kalman filters to integrate GPS sensor readings over time [16; 4]. A disadvantage of these approaches is that they only represent unimodal distributions, which is clearly insufficient for noisy sensor information. Furthermore, at vertices of the Voronoi graph, belief distributions are clearly not Gaussian and we suspect that even mixtures of Gaussians are not well suited to represent very uncertain distributions on Voronoi graphs.

Despite these very encouraging results, there are still warrants for future research. First, we will collect long-term data from people working in the environment. This data will be used to estimate the parameters of the model for each person. Long-term data logs will allow us to determine whether the model is able to detect different motion patterns for the different people. So far, our Voronoi model only captures first-order transitions. This can only be the first step towards high-level pattern learning. For example, most navigation patterns depend on the time of day and the (potentially far away) goal of the person and not only on the previous position. We intend to use the model presented here as a tool to generate long sequences of data logs based on which we can determine high-level behavior patterns. Nevertheless, we strongly believe that this is a first step towards unsupervised learning of behavior patterns.

Acknowledgments

This work has partly been supported by the NSF under grant number IIS-0093406, by DARPA's SDR Programme (contract NBCHC020073), and the Intel corporation.

References

[1] D. Austin and P. Jensfelt. Using multiple Gaussian hypotheses to represent probability distributions for mobile robot localization. In *Proc. of the IEEE International Conference on Robotics & Automation*, 2000.

[2] M. Bennewitz, W. Burgard, and S. Thrun. Using EM to learn motion behaviors of persons with mobile robots. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002.

[3] J.A. Bilmes. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. Technical Report ICSI-TR-97-021, University of Berkeley, 1998.

[4] P. Bonnifait, P. Bouron, P. Crubillé, and D. Meizel. Data fusion of four ABS sensors and GPS for an enhanced localization of car-like vehicles. In *Proc. of the IEEE International Conference on Robotics & Automation*, 2001.

[5] H. Choset. *Sensor Based Motion Planning: The Hierarchical Generalized Voronoi Graph*. PhD thesis, Caltech, 1996.

[6] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo in Practice*. Springer-Verlag, New York, 2001.

[7] A. Fod, A. Howard, and M.J. Mataric. Laser-based people tracking. In *Proc. of the IEEE International Conference on Robotics & Automation*, 2002.

[8] D. Fox, S. Thrun, F. Dellaert, and W. Burgard. Particle filters for mobile robot localization. In Doucet et al. [6].

[9] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *Computer*, 34(8), 2001. IEEE Computer Society Press.

[10] E. Kruse and F. Wahl. Camera-based monitoring system for mobile robot guidance. In *Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1998.

[11] R. Levine and G. Casella. Implementations of the Monte Carlo EM algorithm. *Journal of Computational and Graphical Statistics*, 10, 2001.

[12] N.B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. In *Proceedings of MOBICOM 2000*, Boston, MA, 2000. ACM Press.

[13] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*. IEEE, 1989. IEEE Log Number 8825949.

[14] D. Schulz, W. Burgard, and D. Fox. People tracking with a mobile robot using joint probabilistic data association filters. *International Journal of Robotics Research (IJRR)*, 2003. Accepted for publication.

[15] R. Simmons and S. Koenig. Probabilistic robot navigation in partially observable environments. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1995.

[16] R. Thrapp, C. Westbrook, and D. Subramanian. Robust localization algorithms for an autonomous campus tour guide. In *Proc. of the IEEE International Conference on Robotics & Automation*, 2001.

[17] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1), 1992.

[18] G. Wei and M.A. Tanner. A Monte Carlo implementation of the EM algorithm and the poor mans data augmentation algorithms. *Journal of the American Statistical Association*, 85, 2001.