2003 Workshop on Location-Aware Computing (Part of UbiComp 2003), October 12, 2003, Seattle, WA, USA

Probabilistic Inferencing for Location

John Krumm
Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA USA
jckrumm@microsoft.com

Keywords
Ubiquitous computing, pervasive computing, location, tracking, probability

INTRODUCTION

Automatically measuring the location of a person or device for ubiquitous computing always involves the conversion of a raw measurement into a location measurement. While this process is a prepackaged part of some sensors (*e.g.* GPS and cell phones), researchers are often faced with making this conversion on their own as part of their efforts to deploy novel sensing technologies. This paper describes and discusses various general techniques that researchers have adopted for processing sensor readings into location measurements, emphasizing probabilistic approaches. Reasoning probabilistically is attractive because it naturally accounts for the uncertainty and ambiguity of sensor data, and a probabilistic representation is a good way to communicate uncertainty to higher level modules that exploit location. The paper concludes by advocating recursive filtering as the best general technique to use, with particle filtering having a slight advantage over the next best recursive technique, the hidden Markov model.

DETERMINISTIC FUNCTION INVERSION

Sometimes there is no probability involved. Suppose that at time t a sensor (or sensors) produces a vector of l measurements $\mathbf{z}_t$. This $l \times 1$ measurement vector could be l signal strength measurements from l wireless access points, or l ultrasound delays from l ultrasound detectors, or any l measures from one or more sensors. The location at time t is represented by the vector $\mathbf{x}_t$, which is what we want to estimate. $\mathbf{x}_t$ is an $m \times 1$ state vector whose elements might represent a position in space, orientation, velocity, or any combination of state variables that need to be inferred.

It is sometimes possible to model the output of the sensor as a deterministic function of the input, i.e.

$$\mathbf{z}_t = h(\mathbf{x}_t) \qquad (1)$$

If $h(\mathbf{x})$ can be inverted, then $\hat{\mathbf{x}}_t = h^{-1}(\mathbf{z}_t)$, where $\hat{\mathbf{x}}_t$ is the estimate of the state vector. If $h(\mathbf{x})$ cannot be inverted, then a common technique is to find the state vector that minimizes $\|\mathbf{z}_t - h(\mathbf{x}_t)\|^2$ using an iterative least squares algorithm like Levenberg-Marquardt[1].

As a special case, if the measurement and state vectors are related linearly as $\mathbf{z}_t = H\mathbf{x}_t$ by the $l \times m$ matrix H, then the least squares solution has the closed form

$$\hat{\mathbf{x}}_t = \left(H^T H\right)^{-1} H^T \mathbf{z}_t \qquad (2)$$

For most problems of interest, however, the relationship between the state vector and measurement vector is not deterministic, so probabilistic methods must be used.

MAXIMUM LIKELIHOOD ESTIMATE

Often the relationship between sensor readings and state variables is characterized by a state-conditional probability $p(\mathbf{z}_t|\mathbf{x}_t)$. This is merely a probabilistic sensor model giving the distribution of measurement vectors for a given state variable. It can be determined by simulating the sensor or by taking enough actual measurements to make a histogram of the frequency of measurement values as a function of known inputs.

Given a measurement $\mathbf{z}_t$, the maximum likelihood estimate of the state is the state that maximizes the state-conditional probability:

$$\hat{\mathbf{x}}_t = \arg\max_{\mathbf{x}_t} p(\mathbf{z}_t|\mathbf{x}_t) \qquad (3)$$

One special case is a generalization of the deterministic linear relationship in the previous section. Here $\mathbf{z}_t = H\mathbf{x}_t + \mathbf{v}$. $\mathbf{v}$ is a $l \times 1$ normally distributed noise vector with zero mean and $l \times l$ covariance matrix R, *i.e.* $\mathbf{v} \sim N(\mathbf{0}, R)$. The maximum likelihood estimate is similar to Equation (2), but accounts for the covariance of the noise[2]:

$$\hat{\mathbf{x}}_t = \left(H^T R^{-1} H\right)^{-1} H^T R^{-1} \mathbf{z}_t \qquad (4)$$

Another special case commonly occurs when the state space is discretized into a finite number of classes, $C = \{c_1, c_2, \ldots, c_n\}$. For instance, the classes might represent different rooms of a building or different discrete points on the floor. The state-conditional probability is then conditioned on discrete states -- $p(\mathbf{z}_t|c_i)$ -- and the maximum likelihood estimate is simply the state with the largest state-conditional probability evaluated at $\mathbf{z}_t$.The discrete states mean that this is actually a pattern classification problem[3].

Maximum likelihood is a widely accepted method for making probabilistic inferences in the absence of prior assumptions, dynamic models on the tracked subject, and past data. Dealing with prior assumptions is discussed in the next section on maximum *a posteriori* estimates, and dynamic models and past measurements are discussed in the section on recursive filtering.

MAXIMUM *A POSTERIORI* (MAP) ESTIMATE

MAP estimates depend on prior probabilities about the actor's state, denoted as $p(\mathbf{x}_t)$ for the continuous state case and $p(c_i)$ for the discrete state case. As an example, the priors might encode the fact that people generally don't spend much time in the hallway and are much more likely to be found in their offices.

For the continuous state case, the *a posteriori* probability distribution of state given a measurement is given by Bayes' rule:

$$p(\mathbf{x}_t|\mathbf{z}_t) = p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t)/p(\mathbf{z}_t) \quad (5)$$

The maximum of $p(\mathbf{x}_t|\mathbf{z}_t)$ over $\mathbf{x}_t$ is the MAP estimate $\hat{\mathbf{x}}_t$. $p(\mathbf{z}_t)$ is unaffected by $\mathbf{x}_t$, so it is sufficient to maximize the numerator, which is just the product of the state-conditional probability and the *a priori* probability. Thus the only difference between the maximum likelihood estimate and MAP estimate is the inclusion of *a priori* assumptions for MAP. It is usually easy to make reasonable prior assumptions about peoples' location for tracking, so MAP is an easy way to improve accuracy.

RECURSIVE ESTIMATES

The techniques discussed so far lack the ability to exploit dynamic models of the tracked subject, such as expectations of possible speeds and feasible paths. They also ignore past measurements. Recursive filtering techniques maintain a probabilistic distribution of state that implicitly includes the effect of all past measurements and dynamic assumptions, and they give a technique for updating this distribution with new measurements. By looking back in time, a recursive filter looks at the path of a tracked user instead of just instantaneous position like the memoryless techniques above. Examining a sequence of measurements in time, along with a dynamic model, is an effective way to deal with ambiguious measurements that could have come from more than one location. With these abilities, recursive filtering is generally considered the best way to process sensor data for location. The following sections discuss three recursive filtering techniques: Kalman filter, hidden Markov model, and particle filter.

Kalman Filter

The discrete time Kalman filter is based on simplifying assumptions about both the measurement process and system dynamics. The Kalman filter assumes that the relationship between the measurement vector $\mathbf{z}_t$ and state vector $\mathbf{x}_t$ is linear with zero-mean, additive, Gaussian noise. It also assumes that the relationship between the previous state $\mathbf{x}_{t-1}$ and current state $\mathbf{x}_t$ is linear with zero-mean, additive, Gaussian noise. Mathematically, these assumptions are

$$\mathbf{x}_t = \Phi_{t-1}\mathbf{x}_{t-1} + \mathbf{w}_{k-1} \quad (6)$$

$$\mathbf{z}_t = H\mathbf{x}_{t-1} + \mathbf{v}_k \quad (7)$$

Where Φ_{t-1} is an $m \times m$ matrix, H is an $l \times m$ matrix, and $\mathbf{w}_{k-1}$ and $\mathbf{v}_k$ are zero-mean, Gaussian noise vectors.

The Kalman update equations[2] give a simple means of updating the previous state vector with new measurements using closed-form matrix math, resulting in a Gaussian distribution describing the mean and variance of the state estimate.

Some limitations of the Kalman filter for tracking are:

- System dynamics must be linear. This means that sharp turns can be hard to model, and it gives no means of constraining a path from passing through a wall or other barriers.
- Measurements must be linear in state. Most sensor models must be greatly simplified to conform to this assumption.
- Gaussian representation of state. There can be no multimodal estimates of a person's location, and the estimate must always be Gaussian-shaped. This is often much too simplistic for many tracking scenarios.
- Measurement association is fixed. The Kalman filter does not allow any ambiguity in which sensor measurement is associated with which tracked individual. Representing this ambiguity is important for certain "anonymous" sensors like motion detectors and pressure sensors.

Nonlinearities have been addressed with the Extended Kalman Filter (EKF). There are modifications to deal with multimodal distributions. The radar tracking community has developed techniques for reasoning about data association in the context of Kalman filtering[4]. In its natural state, however, the Kalman filter has been surpassed by hidden Markov models and particle filters.

Hidden Markov Model

A hidden Markov model (HMM) represents the state space as a set of n possible discrete states $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\}$. For instance, location on the floor of a building could be represented by a grid of n (x, y) points. These states can only be observed through the measurement vector $\mathbf{z}$, which is related to the states through the state-conditional probabilistic sensor model $p(\mathbf{z}_t|\mathbf{x}^{(i)})$. The states are said to be hidden by $p(\mathbf{z}_t|\mathbf{x}^{(i)})$, thus the "hidden" in HMM. $p(\mathbf{z}_t|\mathbf{x}^{(i)})$ can be any distribution, not just a Gaussian as in the Kalman filter.

The state dynamics of an HMM are governed by a first order Markov assumption that says the current state depends on only the immediately previous state in time. Since the states are discrete, the dynamics is represented by a matrix of transition probabilities $\alpha_{ij} = p(\mathbf{x}_{t+1} = \mathbf{x}^{(j)}|\mathbf{x}_t = \mathbf{x}^{(i)})$. These transition probabilities can be used to suppress impossible jumps between distant points and between points

separated by barriers, and they can be adjusted to reflect assumptions on possible speeds.

For each new measurement $\mathbf{z}_t$, the Viterbi algorithm (see [5]) efficiently computes the maximum likelihood path through the states $\hat{\mathbf{x}}_{0:t} = \{\hat{\mathbf{x}}_0, \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_{t-1}, \hat{\mathbf{x}}_t\}$ that best accounts for the sequence of measurements $\mathbf{z}_{0:t} = \{\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_{t-1}, \mathbf{z}_t\}$.

The main difficulty with using an HMM for tracking location is that all possible states must be explicitly represented. Adding new dimensions to the state representation causes an exponential increase in the number of states, each of which needs its own $p(\mathbf{z}_t|\mathbf{x}^{(i)})$ and transition probabilities to all the other states. This can be mitigated by exploiting independence among subsets of the state variables and using separate HMMs for each subset. For instance, in tracking people, one HMM could be assigned to (x, y) and another could be assigned to speed with, perhaps, the results of the speed inference used to update the transition probabilities for (x, y).

Particle Filter

The particle filter represents a probability distribution of the current state as a set of N state samples and associated scalar weights: $\{(\mathbf{x}_t^{(1)}, w^{(1)}), (\mathbf{x}_t^{(2)}, w^{(2)}), \ldots\ldots, (\mathbf{x}_t^{(N)}, w^{(N)})\}$. The weights sum to one, and larger weights indicate more likely states. Each of these "particles" is continuous and evolves as new measurements are processed. This is in contrast to the HMM where the states are discrete and predefined. Upon receipt of a new measurement $\mathbf{z}_t$, a new set of particles is computed in three steps:

1. Create $\{\mathbf{x}'^{(1)}_{t-1}, \mathbf{x}'^{(2)}_{t-1}, \ldots\ldots, \mathbf{x}'^{(N)}_{t-1}\}$ by sampling with replacement from $\{(\mathbf{x}_{t-1}^{(1)}, w^{(1)}), (\mathbf{x}_{t-1}^{(2)}, w^{(2)}), \ldots\ldots, (\mathbf{x}_{t-1}^{(N)}, w^{(N)})\}$, where samples are drawn randomly in proportion to the scalar weights.
2. Propagate these samples to the current time by randomly generating a new sample $\mathbf{x}_t^{(i)}$ from each $\mathbf{x}'^{(i)}_{t-1}$ via the transition probability $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. This models the state dynamics. This is the same Markov assumption as the HMM, only for continuous states.
3. Assign a new weight to each $\mathbf{x}_t^{(i)}$ according to the state-conditional probability $p(\mathbf{z}_t|\mathbf{x}_t)$. Normalize these weights so they sum to one. This gives $\{(\mathbf{x}_t^{(1)}, w^{(1)}), (\mathbf{x}_t^{(2)}, w^{(2)}), \ldots\ldots, (\mathbf{x}_t^{(N)}, w^{(N)})\}$.

These weighted samples give a versatile way of approximating any *a posteriori* state probability distribution. Each iteration through the three steps requires knowledge of only the probabilistic transition probabilities $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and the state-conditional probabilities $p(\mathbf{z}_t|\mathbf{x}_t)$. These two probability functions can be arbitrarily complex, allowing for the modeling of realistic dynamics and sensors.

The main problem with particle filters is that the required number of particles N is hard to predict in advance without experimentation, and the number may be unreasonably large depending on the dimensionality of the state space. As with the HMM, there are techniques to exploit the independence of the state variables, such as Partitioned Sampling[6]. In fact the general topic of sequential importance sampling is still an active area of research, and the algorithm presented above is only one of many possible depending on what prior assumptions can be made about the probabilistic processes involved. One example of the use of a particle filter for tracking inside buildings is explained in [7].

CONCLUSIONS

Deterministic methods of sensor interpretation are burdened by the difficultly of modeling all aspects of a location sensing system. Lumping the unmodeled effects under the label "random" leads to probabilistic models which have the benefit of explicitly representing the inherent uncertainty and, depending on the model used, the multimodal ambiguity. Recursive filtering can efficiently take into account dynamic models and past measurements to compute location, which makes them preferred over the memoryless methods of simple maximum likelihood and MAP estimation. Of the recursive techniques, the Kalman filter is limited to Gaussian distributions and linear dynamics, while the HMM and particle filter offer much more flexibility at the expense of more memory-intensive representations. The continuous state representation of the particle filter has a slight advantage over the inherently discrete-state HMM provided the required number of particles is not too large.

REFERENCES

1. Press, W.H., et al., *Numerical Recipes in C*. 1992, Cambridge, UK: Cambridge University Press.
2. Gelb, A., *Applied Optimal Estimation*. 1974, Cambridge, MA USA: The M.I.T. Press.
3. Duda, R.O. and P.E. Hart, *Pattern Classification and Scene Analysis*. 1973, New York: John Wiley & Sons.
4. Blackman, S.S., *Multiple-Target Tracking with Radar Applications*. 1986: Artech House, Inc.
5. Rabiner, L.R., *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition.* Proceedings of the IEEE, 1989. 77(2): p. 257-286.
6. MacCormick, J. and A. Blake, *Probabilistic exclusion and partitioned sampling for multiple object tracking.* International Journal of Computer Vision, 2000. 39(1): p. 57-71.
7. Liao, L., et al. *Voronoi Tracking: Location Estimation Using Sparse and Noisy Sensor Data*. in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2003.