Finding location using omnidirectional video on a wearable computing platform

Wasinee Rungsarityotin, Thad E. Starner

College of Computing, GVU Center,
Georgia Institute of Technology
Atlanta, GA 30332-0280 USA
{wasinee|testarne}@cc.gatech.edu

Abstract

In this paper we present a framework for a navigation system in an indoor environment using only omnidirectional video. Within a Bayesian framework we seek the appropriate place and image from the training data to describe what we currently see and infer a location. The posterior distribution over the state space conditioned on image similarity is typically not Gaussian. The distribution is represented using sampling and the location is predicted and verified over time using the Condensation algorithm. The system does not require complicated feature detection, but uses a simple metric between two images. Even with low resolution input, the system may achieve accurate results with respect to the training data when given favourable initial conditions.

1. Introduction and Previous Work

Recognizing location is a difficult but often essential part of identifying a wearable computer user's context. Location sensing may be used to provide mobility aids for the blind [14], spatially-based notes and memory aids [20, 18, 8], and automatic position logging for electronic diaries (as used in air quality studies [6]).

A sense of location is also essential in the field of mobile robotics. However, most mobile robots combine extrinsic (environmental) sensors such as cameras or range sensors with their manipulators or feedback systems. For example, by counting the number of revolutions of its drive wheels, a robot maintains a sense of its travel distance and its location based on its last starting point. In addition, many robots can close the control loop in that they can hypothesize about their environment, move themselves or manipulate the environment, and confirm their predictions by observation. If their predictions do not meet their observations, they can attempt to retrace their steps and try again.

Determining location with the facilities available to the wearable computer provides an additional challenge. Accurate, direct feedback sensors such as odometers are unavailable, and many sensors typical in mobile robotics are too bulky to wear. In addition, the wearable has no direct control of the user's manipulators (his/her feet) and consequently is forced to a more loosely coupled feedback mechanism.

In the wearable domain, small video cameras are attractive for sensing because they can be made unobtrusive and provide a great deal of extrinsic information. This is beneficial since we cannot instrument a person with as many sensors as we do a robot. A major challenge of using vision is to build a framework that can handle a complex multimodal statistical model. A recent approach developed in both statistics and computer vision for problems of this nature is the Condensation algorithm, a form of the Monte Carlo algorithm that simulates a distribution by sampling.

Our approach to determining location is based on a simple geometric method that uses omnidirectional video for both intrinsic (body movement) and extrinsic (environmental changes) sensing. The Condensation algorithm combines these different types of information to predict and verify the location of the user. Inertial data, such as provided by a number of personal dead reckoning modules for the military, can be used to augment or replace the intrinsic sensing provided by the omnidirectional camera. We annotate a two dimensional map (the actual floor plan of the building) to indicate obstacles and track the user's traversal through the building. We attempt to reconstruct a complete path over time, providing continous position estimates as opposed to detecting landmarks or the entrance or exiting of a room as in previous work.

In the wearable computing community, computer vision has traditionally been used to detect location through fiducials [17, 15, 21, 1]. More recently, an effort has been made to use naturally occuring features in the context of a museum tour [13]. However, these systems assume that the user is fixating on a particular object or location and expects visual or auditory feedback in the form of an augmented reality framework. A more difficult task is determining user location as the user is moving through the environment without explicit feedback from the location system. Starner *et al.* [22] use hidden Markov models (HMM's) and simple features from forward and downward looking hat-mounted cameras (Figure 6) to determine in which of fourteen rooms a user is traveling with 82% accuracy. Using a forward-looking hat mounted camera, Aoki *et al.* [10] demonstrate a dynamic programming algorithm using color histograms to distinguish between sixteen video "trajectories" through a laboratory space with 75% accuracy. Clarkson and Pentland [5] use HMM's with both audio and visual features from body-mounted cameras and microphones for unsupervised classification of locations such as grocery stores and stairways. Continuing this work, Clarkson *et al.* [4] use ergodic HMM's to detect the entering and leaving of an office, kitchen, and communal areas with approximately 94% accuracy. Unlike these previous systems which identify discrete events, our system will concentrate on identifying continuous paths through an environment.

In computer vision, Black [2] and Blake [12, 3] have used the Condensation algorithm to perform activity recognition. In mobile robot navigation, Thrun *et al.* [7] also use the Condensation algorithm with the brightness of the ceiling as the observation model. A camera is mounted on top of a robot to look at the ceiling, and the brightness measure is a filter response. The most recent work by Ulrich and Nourbakhsh [23] is most similar to ours in that they use omnidirectional video and require no geometric model. However, their goal is to recognize a place on a map, not to recover a path. In this sense, there is no need to propagate the posterior distribution over time and thus their nearest-neighbor algorithm is sufficient.

1.1 Our Approach

Our main motivation is to demonstrate a vision-based system that can determine location without recovering the 3D geometry of a scene. In other words, we want to have a framework that works even with low resolution input. We approach this problem using a Bayesian predictive framework. Because the likelihood model is not known, the system must be trained to learn the model.

The first stage in our approach is to capture images of the environment for training. The next stage is labeling of the training data. Because there is no explicit geometric modeling, we need to associate images with positions on an actual blueprint of the environment. The map must also represent obstacles such as walls to assist in the prediction of motion. This is done by the user editing the map to represent obstacles and valid travel areas (Figure 1). The last step in the training is to learn the likelihood model. Given a location on a map and the image measurement, we can view the likelihood as a probability that the observed image is from that location.

We considered two simple image metrics: the L2 norm and a color histogram. We chose the L2 norm because the color histogram did not provide enough discrimination. For example, in our data set, hallways did not have enough color variation to show significant differences in their respective color histograms.

On the other hand, a slight problem with using the L2 norm is maintaining rotational invariance so that different views taken from the same location look similar. We discuss the solution to this problem in Section 2.1.

2. The Method

In a Bayesian framework, we want to find the posterior distribution conditioned on the image measurement L at time t. Define a state at time t as $\P^t = (x, y)$ a position on a 2D map and the observation as the image measurement L. Bayes's rule states that

$$p(\P^t|L) = \frac{p(L|\P^t)p(\P^t|\P^{t-1})}{p(L)} \tag{1}$$

We can assume that the probability of getting a measurement $p(L)$ is constant. Thus, the equation becomes

$$p(\P^t|L) \propto p(L|\P^t)p(\P^t|\P^{t-1}) \tag{2}$$

where $p(L|\P^t)$ is the likelihood conditioned on the prediction and $p(\P^t|\P^{t-1})$ is a probabilistic motion model. The motion model must obey the first order Markov assumption, that the prediction depends only on the previous state. In practice, it is easy to find a parametric motion model that closely reflects the physical system. This can be a dynamic model with velocity and acceleration or simply a normal distribution. However, representing the likelihood $p(L|\P^t)$ is a more difficult pattern recognition problem. In our framework, the system has to learn this distribution for every new environment.

2.1 Image Measurement

Because we are using low resolution images taken from a parabolic camera, finding good features to recover a person's movement is difficult (in this case, the situation is the same as the camera ego-motion problem). One easy choice is to use image similarity to identify a match (we use the L_2 norm). The limitation is that the learned model is not

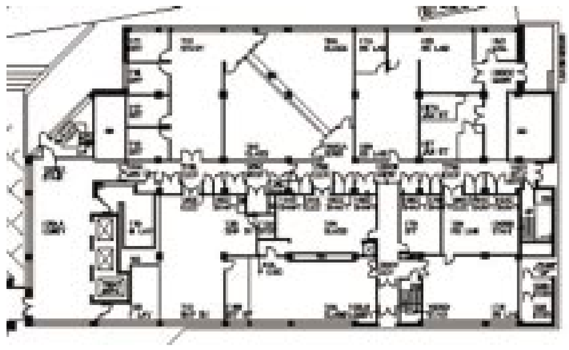
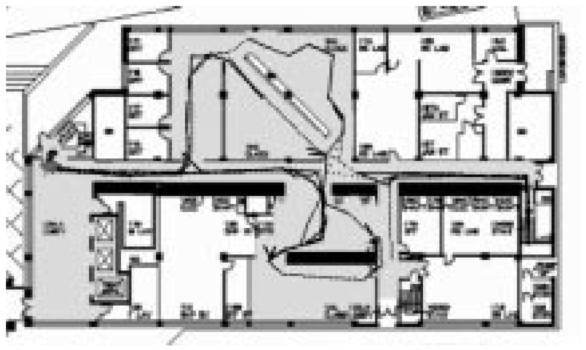
(a)  (b) 

Figure 1. An actual floor plan and a map with a valid area drawn in gray. The training paths are traced on the map in black lines.

invariant over time. Thus, for the purpose of this study, we assumed that both training and testing were taken within two hours of each other.

We greatly benefit from having an omnidirectional camera because it allows us to use a normalized L_2 distance as a similarity metric. Assuming that changes caused by translation are negligible, we only need to make the L_2 metric invariant to rotation. We do this by incrementally rotating the image until we find a minimum error. This can be viewed as an image stabilization process.

Using the same definition for a state and observation, define the likelihood $p(L_t|\hat{\P}^t) = p(L^t|d_t)$, where $\hat{\P}^t$ is a predicted position at time t, $L^t = ||I_t - I_k||_{L2}$ (the normalized L_2 norm), $d_t = ||\hat{\P}^t - \P_k||$, and $\P_k$ is a state in the training data nearest to $\hat{\P}^t$.

The next section will explain an experiment on finding the likelihood for the localization system. We will focus on modeling a case when the state (user's location) is near a training path (having distance on a map less than 10 pixels (3 feet per pixel) away). A $d > 10$ region defines a low certainty region, and we have chosen this threshold to represent when a state is not on a training path.

2.2 The Likelihood model

As seen in Figure 2, the likelihood is far from being a simple two dimensional function. Although the distribution appears noisy, it exhibits some structure as seen in the contour plot. Rather than performing a full minimization to solve for a closed form, we have chosen to estimate the likelihood with a combination of known distributions. To estimate the likelihood, first we compute a non-parametric form of $p(L, d)$ by uniform bin-size and it follows that $p(L^t|d) = \frac{p(L^t,d)}{p(d)}$. Looking at a plot of $p(L^t|d)$ in Figure 2(a), we have observed that if the image distance L^t is higher than 0.1, it is likely that the state is not a good match. We have used 0.1 as the standard deviation for the normal distribution and our experiment in Section 3.1 confirms that the above observation is reasonable.

In summary, we have tried three different functions to approximates the likelihood. Define the lower and upper limit of $p(L^t, d)$ in Figure 3 as

$$\upsilon(d) = 0.5 - 0.415e^{-0.17d} \quad (3)$$

$$\lambda(d) = 0.12 - 0.055e^{0.26d} \quad (4)$$

Let $p'(L|d)$ be a parametric estimation of $p(L|d)$. We can define our three choices in terms of $L, d, \upsilon(d), \lambda(d)$ as:

1. an exponential decay independent of d,

$$p'(L|d) = N(0, 10^{-2}), \forall L \geq 0 \quad (5)$$

   where $N(\mu, \sigma^2)$ is a normal distribution

2. a normal distribution with the mean and variance controlled by $\upsilon(d)$ and $\lambda(d)$,

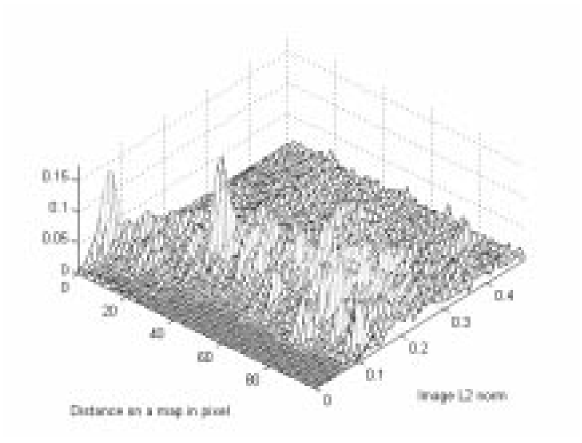
$$p'(L|d) = N(\frac{\upsilon(d) + \lambda(d)}{2}, (\upsilon(d) - \lambda(d))^2) \quad (6)$$

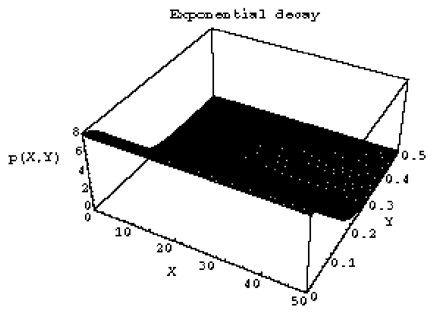
3. a gamma distribution with shape and rate controlled by $\upsilon(d)$ and $\lambda(d)$. Let $g(t, r, y)$ be a gamma distribution with shape parameter t and the rate r. Define the likelihood $p(L|d)$ as a gamma distribution with;

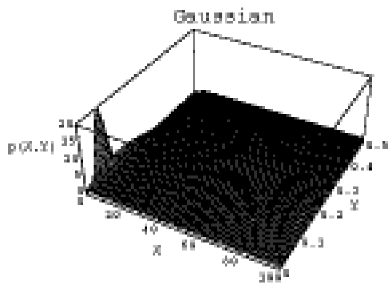
$$\mu = \frac{\upsilon(d) + \lambda(d)}{2} \quad (7)$$

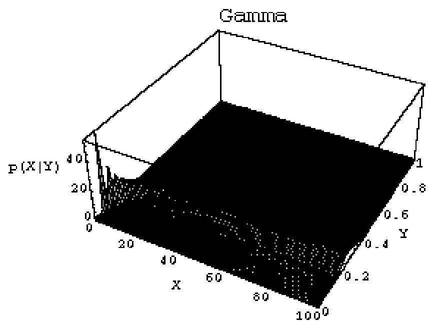
$$\sigma = \frac{\upsilon(d) - \lambda(d)}{4} \quad (8)$$

$$p'(L|d) = g(t, r, y), t = \frac{\mu^2}{\sigma^2}, r = \frac{\sigma^2}{\mu} \quad (9)$$

(a) 

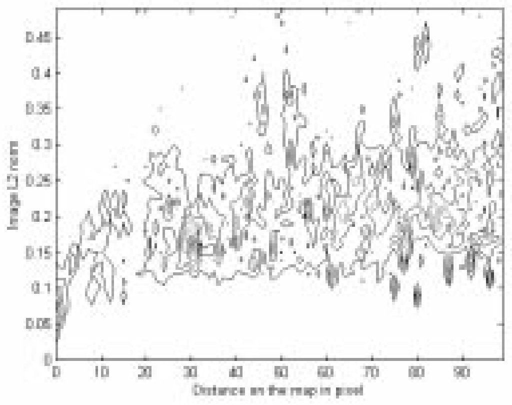
(b) 





(c) (d)

Figure 2. These figures show the estimation of $p(L|d)$ with (a) a histogram with uniform bin-size, (b) an exponential decay (Equation 5), (c) a Gaussian for each d (Equation 6), and (d) a Gamma distribution for each d(Equation 9).

(a) 

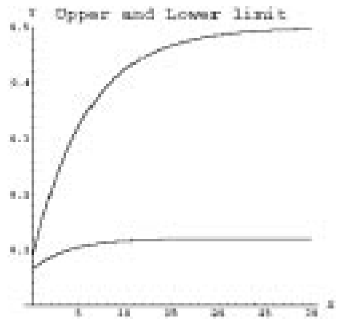
(b) 

Figure 3. Figure (a) shows a two dimensional contour of Figure 2(a) illustrating that our density estimation has lower and upper limits (approximated by two curves in Figure (b)) that control the shape of the likelihood.

2.3 The Motion Model

A full recovery of ego-motion is a difficult problem even with an omnidirectional camera. Most of the algorithms presented in computer vision require good features to track [19]. For further information, a detailed algorithm to recover the ego-motion of a parabolic camera is presented by Gluckman and Nayar [9]. Given a low resolution imaging system such as ours (7), finding good features can be expensive.

To keep the computation simple, we took the same approach by Yagi *et al.* [24] with the assumption that the camera moves in a horizontal plane with constant height above the ground. We simplified the model more by only computing the motion of the ground-plane. The model has three degrees of freedom, translation t_x, t_y and rotation α around the Z-axis. Because we only account for the motion of the ground-plane, motion in other planes will contain more error. To represent the uncertainty of the estimated motion, we distributed a set of samples around the solution and applied it to the Condensation algorithm. A random variable $\{t_x, t_y, \alpha\}$ was then transformed into a two dimensional space to be rendered on a map by rotating the translation vector t_x, t_y by the angle α.

Although we only mention estimating a person's displacement from a camera, the framework is not restricted to a single motion model. We could replace the recovery of the ego-motion with the inertial sensor or combine both. For our experiments, we only tried to estimate from the camera because we have a better way to quantify the uncertainty.

2.4 The Condensation algorithm

Initial condition Start with an initial position $\P^0$ and the prior density $p(\P^0)$. Let $\hat{\P}$ denote a prediction and I^t be an image. To estimate $p(\P^t|L)$ at time t given the motion model $p(\hat{\P}|\P^{t-1})$ and the prior from the previous step $p(\P^{t-1}|I^{t-1})$, the Condensation algorithm states as follow:

1. Start with a set S^{t-1} of N samples that represents $p(\P^{t-1}|I^{t-1})$.
2. For all samples $\{s_i, \P_i^{t-1}\}$ with a position $\P_i^{t-1}$, make a prediction by applying a motion model $p(\hat{\P}|\P^{t-1})$.
3. Update the weight for each sample, for all $\{s_i, \hat{\P}_i\}$, $w_i = p(L_2|\hat{\P}_i)$.
4. Sample from a discrete set of $\{\hat{\P}_i, w_i\}$ and iterate with this new set of samples that represents $p(\P^t|I^t)$.

Note that the Condensation algorithm is a form of Monte Carlo algorithm. There are other related algorithms such as Importance Sampling [3, 11] and Markov Chain Monte Carlo methods (MCMC) [16, 7]. A review by Neal [16] provides a comprehensive review of MCMC with attention to their applications to problems in artificial intelligence.

2.5 Initialization and Recovery

Initial conditions for navigation can be determined by defining the prior density $p(\P^0)$. Alternatively, it is possible to allow the initial condition to approach a steady state in the absence of initial measurements. Provided that a unique solution exists and the algorithm can converge fast enough, we can populate an entire map with samples and let the Condensation algorithm [7] converges to the expected solution. For all of our experiments, an initial position $\P^0$ is manually specified by looking at the test sequence. In this case, we use a Gaussian for $p(\P^0)$ and thus direct sampling can easily be used. We can use a similar scheme to recover from getting lost–this is the same as finding a new starting point. Deciding that we are lost can be done by observing the expected distance from the training path or the expected weight assigned to samples. Decision regions for the confidence measure can be learned from the likelihood function. Empirically, we make a plot for both parameters and define a confidence region for being certain, uncertain, or confused. In Section 3.2, we will discuss how we define these decision regions from experimental results.

3. Experimental Results

3.1 Finding the likelihood: A case of strong prior

We use two independent test sequences to compare three functions. Additional testing on Equations 5 and 6 shows that the first choice performs much better than the others. One test sequence lasts about $1.5 - 3$ minutes. The Gaussian with varying parameters does not work at all, while the gamma function has an error rate of about 50%. Equation 5 performs much better than the others, with only a 10% error rate.

For the training data, the results show that estimating $p(L|d)$ with exponential fall off gives the best result. It was clear that if the similarity measure was high (low image difference), a new sequence would follow a training path. Thus we can use the training data as the ground truth. For our data set, this prior was so strong that the likelihood independent of d worked well. This explained why the first choice, Equation 5 performed better than the rest. For other data sets, Equations 6 or 9 may work better. A better way

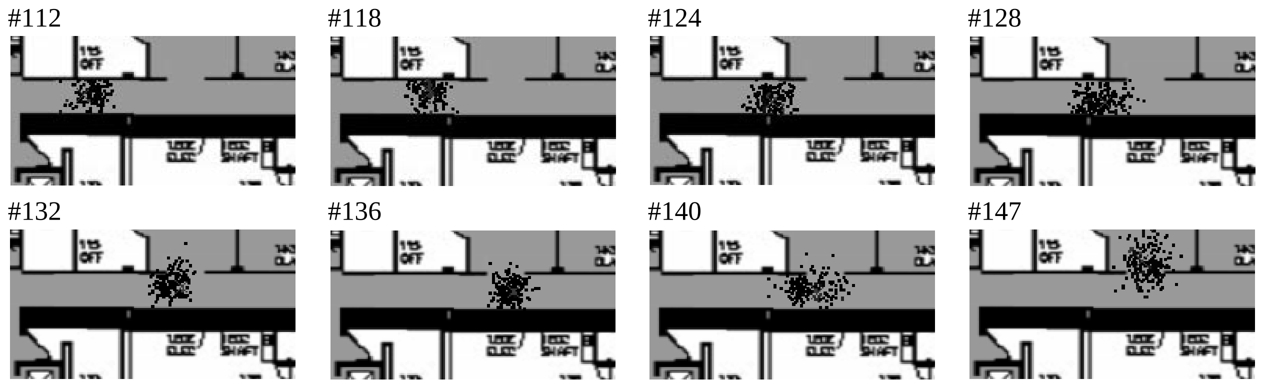


Figure 4. These figures show results from simulating the Condensation algorithm on one test sequence. The real event was that a person came in from the left side of a hall way continuing down the hall and turned right at about frame 147. A small cloud of samples at every time step shows that the localization system was confident and the path was correctly followed.

to learn the likelihood is to estimate a mixture of Gaussian or Gamma distributions.

3.2 Tracking

For experimental verification, we labeled all the test sequences to provide the ground truth. We randomly picked a starting position to avoid bias. We ignored the recovery problem by avoiding an area which we have not seen before. Two motion models were used: a simple random walk and the ego-motion of a camera. Examples of images taken from our omnidirectional parabolic camera are shown in Figure 7. We then masked out some visual artifacts caused by the ceiling, the camera and the wearer (Figure 7).

Table 1 summarizes the performance of our localization system running on one hundred cross-validation tests. A cross-validation is based on two different sequences of images taken from the same path, but acquired at different times. One of the sequences is used as training, while the other is used as test data. We generated a hundred test sequences by choosing one hundred segments from a nine-minute sequence. For each test, the starting position is known and the system tracks for one minute.

For each test, the standard deviation of the likelihood model was 0.1 for the reason given in Section 3.1. We needed to capture at 30 Hz to recover the camera ego-motion, but the likelihood was only computed every sixth frame to increase efficiency. The task of the localization system is to keep track of a person's location and report a confidence measure. After one iteration of the algorithm, it reported a confidence measure as a cumulative weight of all samples. By observing the weight reported by the system, we defined three confidence types as confident, uncertain, or confused. Being uncertain means that a system has competing hypotheses. This results in one or more clouds of samples; in this case, the expected location may lie in a wall between two areas. Being lost implied that the system encountered a novel area or the likelihood was not giving enough useful information. In this case, the prediction was no longer useful. Although it may be possible to recover if a good match appears at a later time, no recovery method was implemented. One possible recovery method is to distribute samples over the entire map to find better candidates to continue tracking.

If the total reported weight was less than 200, we classified the system as being confident, from 200 to 800 as being uncertain, and beyond 800 being confused. For both tests with the random walk and motion estimation, the system was confident for 30% of the time, uncertain for 40% and confused for 29.5% (Table 1). With motion estimation, the error rate was improved for the uncertain case because additional knowledge was provided as to which hypotheses to choose. To show that our confidence measure was meaningful, we associated the measure with a deviation from the true path. If a deviation is more than 30 pixels from the actual path, then this is an error. The error rates were then reported for three confidence measures. As shown in the Table 1, when the system is very confident, the error rate is low. Two tests were performed to study if a motion estimation could reduce the uncertainty. While it did reduce the uncertainty, our simplified motion model introduced more noise to the system which results in an increase in the error rate even though the system is confident.

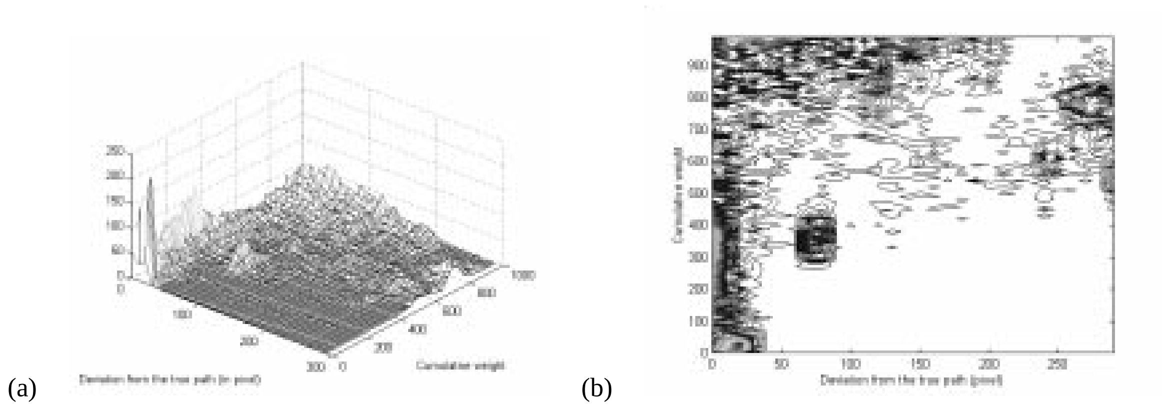


Figure 5. Confidence measure obtained without motion estimation: based on a density plot, we divided the measurement W into three regions as *confidence* when $W < 200$, *uncertain* when $200 < W < 800$ and *confused* when $W > 800$.

Another way to compute the confidence measure is to learn the class distribution from the likelihood. Our result also confirms this because the distribution in Figure 5 appears similar to the likelihood (Figure 2 in Section 2.2). A similar approach has been explored in [23]. More than anything, the high uncertainty and confusion is mainly contributed from having a sparse training set.

It took about three hours to complete a simulation of one hundred test sequences that added up to 100 minutes in real time. Thus, we expect an update rate for an on-line system to be about 2 Hz. With the current system, the most time consuming part for every sample is finding the nearest state from the training set. This can be greatly improved by using an adaptive representation of a 2D map such as an adaptive quad-tree or a voronoi diagram.

4. Conclusion and Future Work

In this paper, we have proposed a probabilistic framework for localization on wearable platforms using data collected from an omnidirectional camera. The framework based on the Condensation algorithm was formulated to determine the user's path without explicit feedback in the form of an augmented reality framework (e.g. fiducials). In addition to having the same challenges presented in continuous tracking of mobile robots, our system has to determine location with limited facilities available to the wearable computer. Direct intrinsic feedback sensors typical in mobile robotics are too bulky to wear. Under these circumstances, the wearable is forced to rely on loosely coupled feedback between intrinsic and extrinsic estimation or sometimes purely on the extrinsic estimation.

On video recordings of real situations in an unmodified environment, we have demonstrated that our system can continuously track independent test sequences 95% of the time given a favourable starting location. The results show that a robust localization system will need a better motion model.

Future work should concentrate on combining intrinsic information from the camera with the inertia data and improving the statistical model of the observation. Future implementations can also use the confidence measure to remain noncommittal and explore the solution space for a good location to restart the Condensation algorithm. Using this information allows the system to recover from situations where sufficient data to match does not exist.

References

[1] M. Billinghurst, J. Bowskill, M. Jessop, and J. Morphett. A wearable spatial conferencing space. In *IEEE Intl. Symp. on Wearable Computers*, pages 76–83, Pittsburgh, PA, 1998.

[2] M. Black. Explaining optical flow events with parameterized spatio-temporal models. In *CVPR99*, 1999.

[3] A. Blake and A. Yuille. Active vision. In *MIT Press*, 1992.

[4] B. Clarkson, K. Mase, and A. Pentland. Recognizing user's context from wearable sensors: Baseline system. Technical Report 519, MIT Media Laboratory, 20 Ames St., Cambridge, MA, March 2000.

[5] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. In *ICASSP*, 1999.

[6] J. Wolf et al. Probabilistic inference using markov chain monte carlo methods. Technical Report GTI-TR-99001-5, Georgia Transportation Institute, Georgia Institute of Technology, 1999.

[7] D. Fox F. Dellaert, W. Burgard and S. Thrun. Using the condensation algorithm for robust, vision-based mobile robot localization. In *CVPR99*, 1999.

[8] S. Feiner, B. MacIntyre, T. Hollerer, and T. Webster. A touring machine: Prototyping 3d mobile augmented reality systems for exploring the urban environment. In *IEEE Intl. Symp. on Wearable Computers*, Cambridge, MA, 1997.

Confidence type	*Confidence*	*Uncertain*	*Confused*
Random walk	4.57%	51.8%	83.89%
With motion estimation	20.23%	44.2%	78.3%

Table 1. The error rate with respect to different confidence types

[9] J. Gluckman and S.K. Nayar. Ego-motion and omnidirectional cameras. In *ICCV98*, pages 999–1005, 1998.

[10] B. Schiele H. Aoki and A. Pentland. Real-time personal positioning system for wearable computers. In *ISWC99*, 1999.

[11] M. Isard. *Visual Motion Analysis by Probabilistic Propagation of Conditional Density*. PhD thesis, Oxford University, 1998.

[12] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. *ECCV*, A:343–356, 1996.

[13] T. Jebara, B. Schiele, N. Oliver, and A. Pentland. Dypers: dynamic and personal enhanced reality system. Technical Report 463, Perceptual Computing, MIT Media Laboratory, 1998.

[14] J. Loomis, R. Golledge, R. Klatzky, J. Speigle, and J. Tietz. Personal guidance system for the visually impaired. In *Proc. First Ann. Int. ACM/SIGCAPH Conf. on Assistive Technology*, pages 85–90, Marina del Rey, CA, October 31–November 1 1994.

[15] K. Nagao and J. Rekimoto. Ubiquitous talker: Spoken language interaction with real world objects. In *Proc. of Inter. Joint Conf. on Artifical Intelligence (IJCAI)*, pages 1284–1290, Montreal, 1995.

[16] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, January 1993.

[17] J. Rekimoto, Y. Ayatsuka, and K. Hayashi. Augment-able reality: Situated communication through physical and digital spaces. In *IEEE Intl. Symp. on Wearable Computers*, pages 68–75, Pittsburgh, 1998.

[18] B. Rhodes and T. Starner. Remembrance agent: a continuously running automated information retrieval system. In *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi Agent Technology (PAAM '96)*, pages 487–495, 1996.

[19] J. Shi and C. Tomasi. Good features to track. In *CVPR94*, pages 593–600, 94.

[20] T. Starner. *Wearable Computing and Context Awareness*. PhD thesis, MIT Media Laboratory, Cambridge, MA, May 1999.

[21] T. Starner, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R. Picard, and A. Pentland. Augmented reality through wearable computing. *Presence*, 6(4):386–398, Winter 1997.

[22] T. Starner, B. Schiele, and A. Pentland. Visual contextual awareness in wearable computing. In *IEEE Intl. Symp. on Wearable Computers*, pages 50–57, Pittsburgh, PA, 1998.

[23] I. Ulrich and I. Nourbakhsh. Appearance-based place recognition for topological localization. In *the 2000 IEEE International Conference on Robotics and Automation*, pages 1023–1029, 2000.

[24] Y. Yagi, W. Nishii, K. Yamazawa, and M. Yachida. Rolling motion estimation for mobile robot by using omnidirectional image sensor hyperomnivision. In *ICPR96*, page A9E.5, 1996.

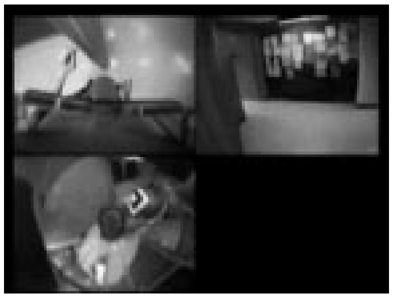


Figure 6. Multiple views One extension to our system is to have the observation taken from multiple views. Starner *et al.* [22] use simple image measurement from forward and downward looking views as shown in the top row, while our system considers only the omnidirectional view (shown in the lower left corner). Measurements from all views can be combined through the observation model.

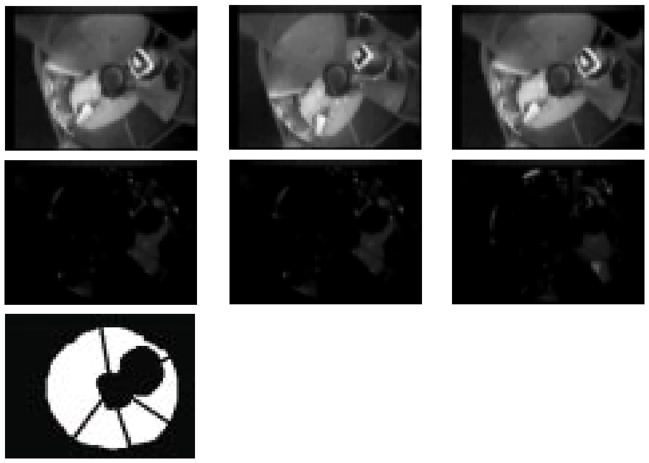


Figure 7. Input images and an image mask The mask was used to eliminate visual artifacts caused by the ceiling and the wearer. All images were sampled down to 80×60 pixels. Images in the first row were taken in a room with good lighting condition. Images in the second row were taken from a dim area. We decided to use the image similarity metric as the measurement because finding good features from these images was not reliable.