Motion Prediction for Moving Objects: a Statistical Approach

Dizan Alejandro Vasquez Govea, Thierry Fraichard

Motion Prediction for Moving Objects: a Statistical Approach

Dizan Vasquez & Thierry Fraichard
Inria Rhône-Alpes & Lab. Gravir, Grenoble (FR)
http://www.inrialpes.fr/sharp

Abstract— **This paper proposes a technique to obtain long term estimates of the motion of a moving object in a structured environment. Objects moving in such environments often participate in typical motion patterns which can be observed consistently. Our technique learns those patterns by observing the environment and clustering the observed trajectories using any pairwise clustering algorithm. We have implemented our technique using both simulated and real data coming from a vision system. The results show that the technique is general, produces long-term predictions and is fast enough for its use in real time applications.**

I. Introduction

The ability to navigate their environment is critical for the survival of most animals and intelligent beings. One of the activities involved on navigating in such an environment is to deal with the objects populating it. This is a difficult challenge because many of those objects are moving. Interaction with those objects requires the ability to predict their future motion (e.g. for predator evasion, prey hunting, collision avoidance, etc.).

Motion prediction is a research area with applications in many different fields, ranging from video surveillance [1] to robot navigation [2]. Many research works on motion prediction found in the literature are based upon an a priori motion model (e.g. differential equations), that describes how the state of a particular object (e.g. position and velocity), changes over time when it is subject to a given control (e.g. acceleration) (cf. [3]). In order to predict the future motion of a particular object, its current state and control are estimated first (using proprioceptive and exteroceptive data, and estimation techniques such as the Kalman filter [4]). Then, the estimated state and control are fed into the object motion model in order to get future state estimations.

Provided that the motion model used is faithful and that the state and control estimations are accurate, such techniques compute good motion predictions. Unfortunately, these conditions are rarely met and this kind of techniques is suited for short term motion prediction only.

To address this issue, a completely different approach has emerged recently. It is based on the idea that, for a given area, moving objects tend to follow typical motion patterns that depend on the objects' nature and the structure of the environment. It operates in two stages:

1. *Learning stage:* observe the moving objects in the workspace in order to determine the typical motion patterns.
2. *Prediction stage:* use the learned typical motion patterns to predict the future motion of a given object.

Techniques following this approach can be classified in two main families:

1. *Grid-based techniques:* derived from the occupancy grid concept [5]. The environment is modeled as a grid and the learning stage computes the transition probability for a moving object from one grid cell to another [6], [7], [8]. The grid is used directly for motion predition.
2. *Cluster-based techniques:* sets of partially or wholly similar observed trajectories are clustered together. A representative trajectory for each cluster is computed [9], [10], [11]. Such representative trajectories are used for motion prediction.

Since they permit to take into account not only the current state of the object but also its past states, cluster-based techniques are by far the best ones when it comes to long term motion prediction. Their only weakness lies in their inability to predict atypical trajectories. Ref. [9] group the parts of the observed trajectories for which the moving objects collide with each other whereas [10] and [11] use the popular Expectation-Maximization algorithm [12] to group whole trajectories.

In this paper, we propose a novel cluster-based technique that learns the typical motion patterns using pairwise clustering. We introduce a dissimilarity metric to allow the use of any clustering algorithm which can operate over a dissimilarity matrix. As a result, we obtain a number of clusters corresponding to the different motion patterns. Then, we calculate the mean value for every cluster, which we further use to predict motion for a partially observed trajectory. We have applied our technique to simulated and real data obtained through a vision system. We have implemented the Expectation-Maximization approach proposed in [10], which we consider to represent the state of the art in Cluster-based techniques, in order to have a reference of the performance of our approach. In our experiments the proposed technique performed better than Expectation-Maximization. Moreover, results show that our approach is general, produces long-term predictions and is fast enough for real time applications.

The paper is organized as follows. In the next section, we present an outline of our approach and explain how its two components work together. In section 3, we present the learning algorithm, making emphasis in the construction of

This work has been partially supported by a Conacyt scolarship. We also want to thank the support of the CNRS Robea ParkNav and the Lafmi NavDyn Projects.

the dissimilarity table and the calculation of the representative trajectory for each cluster. Section 4 explains how to use the output of the learning algorithm to estimate motion for a partially known trajectory. In section 5, we discuss our implementation of the approach and describe the two clustering algorithms that were used. Finally, in section 6, we discuss the results we obtained using simulated and real data and compare them with the approach presented in [10].

II. Proposed Approach

Our approach is a cluster-based technique and it consists of two components: a learning algorithm and an estimation algorithm.

The input of the learning algorithm consists of training data obtained in a given environment. Training data $D = \{d_1, ..., d_N\}$ is a collection of N moving objects' trajectories, which are functions $d_i(t) : [0, T_i] \rightarrow \mathbb{C}$ that returns the configuration of a moving object at time t. T_i is the duration of the object's trajectory, $\mathbb{C}$ is the configuration space of the moving object being considered. In this paper, we will assume that configurations in training data consist of 2 dimensional coordinates $[x, y]$. However, the method is applicable to spaces of higher dimensions. Training data is clustered and each resulting cluster is considered to represent a typical motion pattern. For each obtained cluster, we calculate its representation, which consists of a trajectory: the mean value of all the trajectories in the cluster, and its standard deviation.

In the estimation stage, we model the likelihood that a partially observed trajectory belongs to a given cluster as Gaussian probability function. The parameters of that function are the mean value and standard deviation that we have found in the learning stage. We compute this likelihood for all the clusters. The estimated motion is given by the mean value of the trajectory having a maximum of likelihood. An alternative is to use all the motion patterns having a likelihood greater than a given threshold.

III. Learning Algorithm

In order to discover the typical motion patterns, we analyze training data. We expect that trajectories which are very similar correspond to objects engaged on the same motion pattern. Thus, we will try to find groups of similar trajectories. This leads quite naturally to the use of a clustering algorithm.

A. Clustering Trajectories

The selection of a particular clustering technique is somewhat difficult because the best one to be used depends on the particular problem considered [13]. Another problem in selecting a particular clustering algorithm is that we have to find a way to reformulate our problem in such a way that the selected technique can be applied. We have chosen a formulation which does not confines itself to the utilization of a single algorithm, so that different clustering techniques can be tested in order to find the one that produces the best results.

Many clustering algorithms [13], [14] are able to work using a dissimilarity matrix, which is an $n \times n$ matrix containing all the pairwise dissimilarities[1] between n objects. Thus, finding a way to measure dissimilarities between trajectories allows us to use any of those algorithms.

We define the dissimilarity, or distance between two trajectories d_i and d_j as:

$$\delta(d_i, d_j) = \left(\frac{1}{\max(T_i, T_j)} \int_{t=0}^{\max(T_i,T_j)} \left(d_i(t) - d_j(t)\right)^2 dt \right)^{1/2} \quad (1)$$

Where T_i and T_j are the total motion duration of d_i and d_j respectively, and $d(t) = d(T)$ for $t > T$. This function is the average Euclidean distance between two functions; we have chosen the average because we want our measure to be independent of the length of the trajectories being compared. It is important to note that, as we are integrating over time, the velocities of the two trajectories being compared affect the value of δ. Hence this measure takes velocity profiles into account.

Using (1), we can construct a dissimilarity matrix and use it as the input for a clustering algorithm to obtain a clustering consisting on a set of clusters C_k represented as lists of trajectories.

B. Calculating Cluster Mean-Value and Standard Deviation

One drawback of pairwise clustering is that, as it operates directly over the dissimilarity table, it does not calculate a representation of the cluster. So, if we want to use the cluster's representation as an estimate, we have to calculate this representation.

We have chosen to represent each cluster using what we call its mean-value. Let C_k be a cluster having N_k trajectory functions $d_i(t) \mid 1 \leq i \leq N_k, d_i(t) \in C_k$ then, we define the mean value of C_k as the following function:

$$\mu_k(t) = \frac{1}{N_k} \sum_{i=1}^{N_k} d_i(t) \quad (2)$$

Calculating the standard deviation for the cluster C_k using the mean value is straightforward using the following expression:

$$\sigma_k = \left(\frac{1}{N_k} \sum_{i=1}^{N_k} \delta(d_i, \mu_k)^2 \right)^{1/2} \quad (3)$$

Once we have calculated both the mean value and standard deviation for each cluster, we can use those parameters to estimate motion by applying a criterion of Maximum Likelihood as explained in the next section.

[1] Dissimilarities result from comparing two objects: their value is high if the compared objects are very different, and is zero if they are identical. They are always nonnegative [14]

IV. Estimation Algorithm

The output of the learning algorithm consists of a list of mean values and standard deviations corresponding to the different typical detected behaviors.

In order to estimate trajectories, we calculate the likelihood of the known (already observed) fragment of a trajectory $d_p(t)$ under each one of the clusters. To do that, we model classes as Gaussian sources with the mean value and standard deviation that were calculated during learning.

A. Partial Distance

As we are dealing with partial trajectories, we need to modify (1) to account for this. The modification consists in measuring the distances respect to the duration of the partial trajectory:

$$\delta_p(d_p, d_i) = \left(\frac{1}{T_p} \int_{t=0}^{T_p} \left(d_p(t) - d_i(t)\right)^2 dt \right)^{1/2} \quad (4)$$

Where δ_p, d_p and T_p are the partial distance, partially observed trajectory, and duration of the partial trajectory, respectively.

B. Calculating Likelihood

With the partial distance (4), we can directly estimate the likelihood that d_p belongs to a cluster C_k.

$$P(d_p \mid C_k) = \frac{1}{\sqrt{2\pi\sigma_k}} e^{-\frac{1}{2\sigma^2}\delta_p(d_p,\mu_k)^2} \quad (5)$$

Once we have calculated the likelihood, we can choose, for example, to estimate the trajectory using the mean value of the cluster with maximal likelihood, or to present the different possibilities having likelihood greater than a given threshold.

V. Implementation

We have implemented and tested our technique using two clustering algorithms: Agglomerative Complete-Link Hierarchical Clustering (CL) [15] and Deterministic Annealing Pairwise Clustering (DA) [16]. The former one has been chosen because of its simplicity. Moreover, the shape of the clusters it produces is quite regular, and the distance between any two members of the same cluster is guaranteed not to exceed a given threshold. In spite of that, it has a drawback: since it is based on the optimization of a local, pairwise criterion, it may yield unnatural clusters. So, we have also implemented DA, which is based on the optimization of a global function, and is supposed to produce superior results [16]. We will compare the results provided by both techniques in section VI.

A. Complete-Link

The idea of hierarchical clustering is to produce a recursive representation of the groups. The output is a *dendogram* which is a tree-like structure whose root node is a single cluster grouping all the observations and whose branches correspond to splittings of the parent clusters. Leafs are N singletons corresponding to all the data items in the training data set D.

An agglomerative algorithm starts, with N clusters, one for each data item d_i Then, in the first step, it joins the two closest clusters. As both clusters consist of only one data item, this means that the algorithm joins the pair of data items having smallest dissimilarity, leaving us with $N-1$ clusters. This process is repeated until we have only one cluster. However, this calls for a definition of distance between clusters having more than one item. In fact is the definition of this intercluster distance what distinguishes different clustering algorithms [14]. Three of the most common measures, are nearest neighbor (Single-Link), farthest neighbor (Complete-Link) and average distance. It is Complete-Link that guarantees that the distance between any two members of the same cluster does not exceed a given threshold.

We are interested in clustering down to a given dissimilarity threshold, which is chosen according to the environment and the kind of obstacle being studied. Thus, our implementation of CL does not produce a dendogram but a single clustering corresponding to that threshold.

B. Deterministic Annealing

The other algorithm used, Deterministic Annealing, needs to know *a priori* the number k of clusters to be produced. This is not a trivial task [17], fortunately, we have the value of k produced by the CL algorithm which can be used as an approximation to the real value.

Deterministic annealing, as its stochastic counterpart Simulated Annealing [18] and [19], is a search strategy that uses a control value called *temperature*. The technique tracks solutions from high to low temperatures, where gradually more and more details of the original objective function appear. The complete algorithm is given in [16].

For each temperature value, the algorithm finds the clustering that minimizes a global function. Then, temperature is decreased and the process is started over using the clustering found in the previous iteration as a starting point in the search of the new clustering.

VI. Experimental Results

In order to validate our approach, we have used data coming from two environments: a trajectory simulator and a pedestrian tracking system placed in the Inria entry hall (fig. 1). As the tracking system is under installation, our results with real data are only preliminary. Our main testbed is the simulated environment, which recreates pedestrian motion in the Inria entry hall (fig. 1).

We have separated datasets from each environment into two subsets: training data and test data. We have used the training data to learn the motion patterns, and then, we have used the test dataset to evaluate the obtained results. In addition, we have implemented the Expectation-Maximization algorithm presented in [10] and used it as a comparison basis.

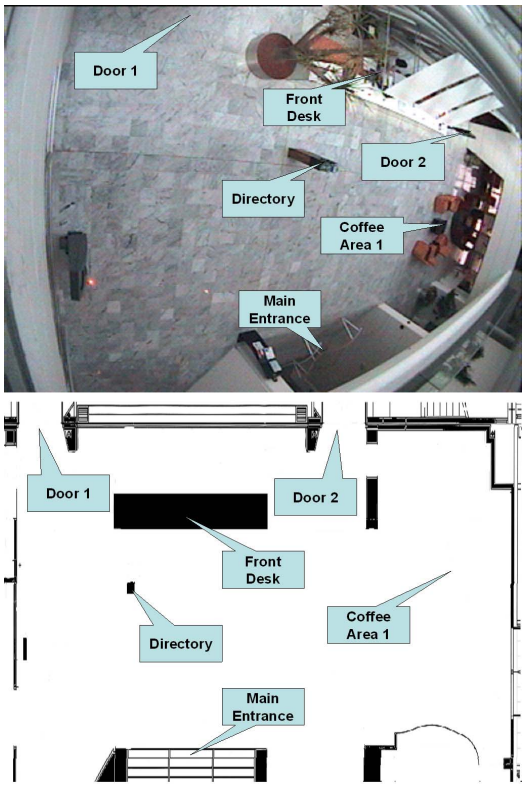

Fig. 1. The Inria entry hall and the simulated environment.

A. Estimation-Maximization

We have chosen to implement the Expectation-Maximization learning algorithm [10] because we consider it as being the state of the art on cluster-based techniques. It models each cluster θ_m as a set of probability distributions $P(x \mid \theta_m^t)$ specifying the probability that the object is located at location x at time t given that it is engaged in this motion pattern. It iteratively refines these models, each iteration consisting of two steps which are repeated until the values converge.

1. *Expectation.* For every trajectory in the data set and each cluster, calculate the expected likelihood that the trajectory belongs to the cluster given the current cluster model.
2. *Maximization.* Calculate new cluster models which maximize the expected likelihood.

An interesting remark about this technique is that, even if it starts with an estimated value of the number k of clusters, it is able to modify it. In particular, it checks for two situations: a) if two models seem to describe the same motion pattern, one of them is eliminated; b) if the likelihood of a particular trajectory under the global clustering is too low, a new cluster is created having a mean value equal to that precise trajectory.

It is important to note that we have used this approach only to perform the clustering. For estimation we have proceeded as described in section IV.

B. The Simulator

For simulated data, we have chosen a number of control points in the environment (some of them can be seen in fig. 1) and then, we have defined 52 trajectory models, each one consisting on a sequence of control points to be traversed. Actual trajectories were generated in two steps: first, for each control point we have drawn a real point from a 2-dimensional Gaussian distribution. Then we have simulated motion by discrete, even-sized steps in a direction drawn from a Gaussian distribution, having the direction of the next control point as mean value. We consider that we have reached a control point when we are closer to it than a given threshold. The particular trajectory model to use is chosen according to a uniform distribution.

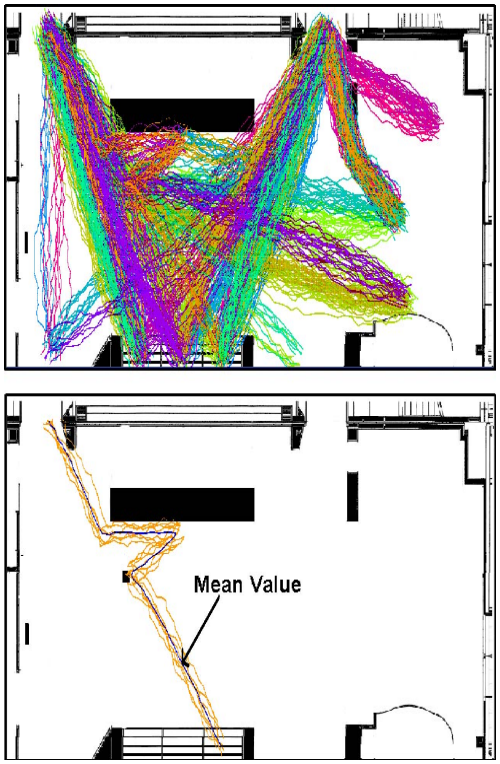

Fig. 2. Raw trajectories and a cluster obtained using DA.

C. Learning Motion Patterns

We have generated training and test datasets of 500 and 100 trajectories, respectively. For the purpose of this experiment, samples are assumed to occur at regular time steps. In order to use (1) we have modeled trajectories as piecewise defined functions consisting of straight segments joining consecutive samples. An example of raw data and a resulting cluster is shown in fig. 2.

We have processed the training dataset to learn patterns using the two chosen clustering algorithms and applying the parameters that can be seen in table I. The last row in the table represents the parameters and the resulting clusters for our implementation of the algorithm presented in [10].

TABLE I
CLUSTERING OF SIMULATED DATA

Algorithm	Parameters	K
CL	$threshold = 20\,\text{cm}$	$k = 205$
DA	$k = 205$	$k = 138$
EM	$\sigma = 10\,\text{cm}$	$k = 133$

D. *Estimating Trajectories and measuring the estimation error*

In order to test the performance of our approach we measure the difference between estimated and real trajectories (fig. 3). For each of the trajectories in the test dataset we take a fraction of its total length. Using this fraction, we search for a match in the set of clusters obtained in the learning stage. The selected cluster will be that having the highest likelihood. We calculate the estimation error as the distance between the mean value of the selected cluster and the complete real trajectory. The error is measured for trajectory lengths between 10% and 80% of the complete trajectory. This procedure is repeated for each of the clustering methods. The results we have obtained for CL, DA and EM can be seen in fig. 4.

There are many interesting observations about the graph. We can see that, if we know only 10% of the total trajectory, DA has a mean error of about 75% of the other algorithms. However, this difference decreases when we know more of the observed trajectory. For all the three algorithms, we can see that, for estimates obtained using 40% of the total trajectory, the mean error is ≈30 cm which can be considered quite accurate for the kind of motion being analyzed.

In figure 5 we can see a sequence of motion estimates for a person (dark disk). The black line represents the complete, real, trajectory while the others correspond to motion hypothesis, darker ones being more probable than lighter ones. We can see that, at the beginning (fig. 5.a) many of the hypothesis describe motion going to the right door. After a short motion (fig. 5.b), these estimates are discarded. Even more hypothesis are discarded as motion progresses in fig. 5.c and 5.d.

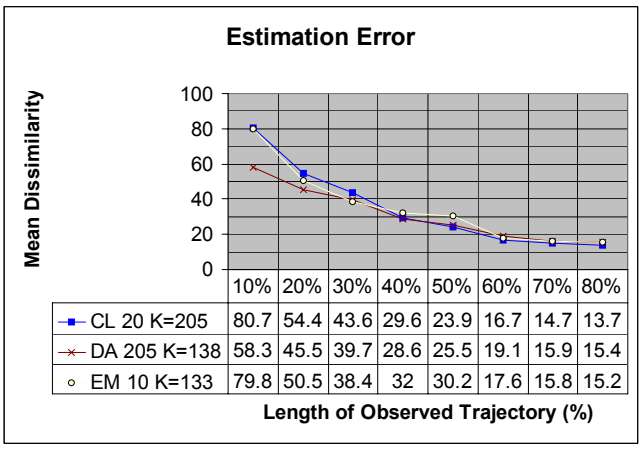

Fig. 4. Mean estimation error for different algorithms.

Our unoptimized implementation of the technique is able to produce estimates with a frequency of 60-100 Hz, which we consider adequate for real-time systems involving vehicles and pedestrians.

VII. CONCLUSIONS

In this paper, we have presented a technique which is able to learn typical motion patterns of objects in a given area. We have defined a dissimilarity measure which allows the use of pairwise clustering algorithms in order to group observed trajectories into patterns. Then, we show how to calculate a representation of the obtained clusters and how to use it in order to calculate the likelihood of a partially known trajectory under a given motion pattern. Furthermore, we propose a way to use the calculated likelihood and the clusters representation to estimate motion.

We have implemented our technique and tested it with simulated data. We have also implemented another technique [10] in order to benchmark our approach. In our experiments, our technique performed slightly better than

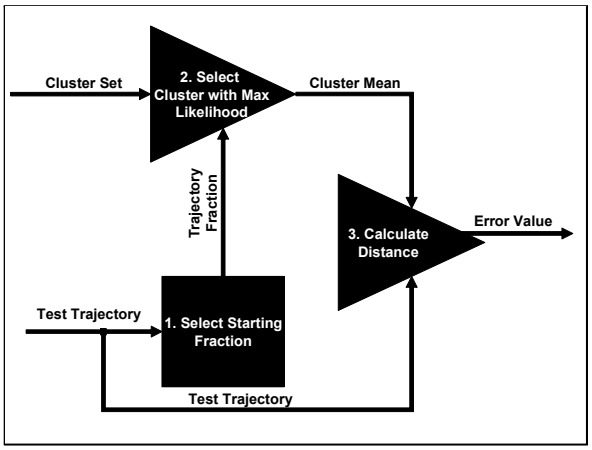

Fig. 3. Measuring the estimation error.

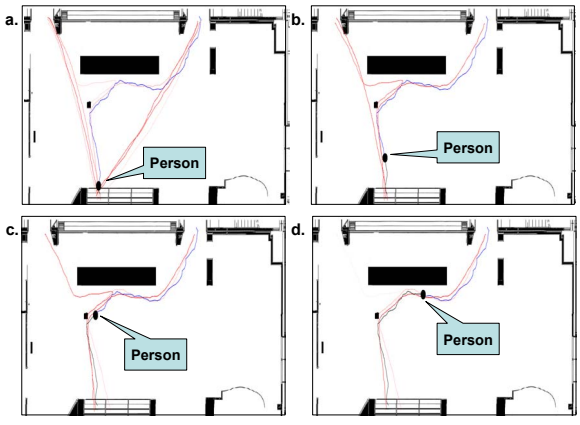

Fig. 5. Motion hypothesis at different moments.

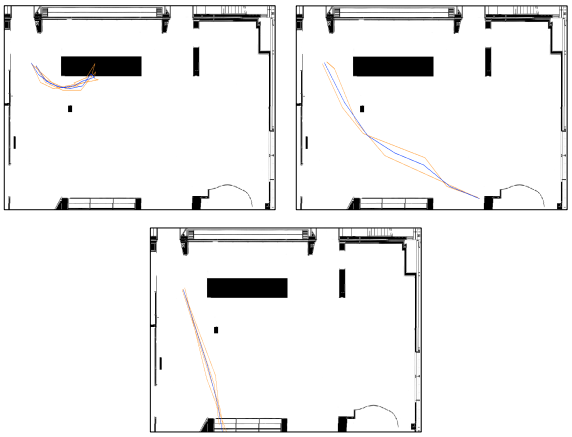

Fig. 6. Clusters obtained from real data obtained through a vision system developed by PRIMA

the other approach. As a result of our experiments, we have shown that our technique is able to learn motion patterns from observations and to produce sound, long-term motion estimates in real time.

VIII. Future Work

Future work includes further experimentation with the real tracking system installed in the Inria entry hall (fig.6).We will also apply our approach to a tracking system currently being installed on the Inria parking lot. This environment has the extra challenge of having heterogeneous moving objects (pedestrian and vehicles). We plan to address this situation by identifying the different types of objects and performing separate training processes for each one of them.

An interesting theoretical direction we want to explore is to extend our approach to model what we call semi-static objects, which are objects whose motion can be characterized by a binary value, examples include: a door, which can be either open or closed; a parking place which can be available or occupied, etc.

References

[1] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawarea, B. Rao, and S. Russell, "Towards robust automatic traffic scene analysis in real-time," in *Proceedings of the 33rd Conference on Decision and Control*, pp. 3776–3781, December 1994.

[2] K. Kyriakopoulos and G. Saridis, "An integrated collision prediction and avoidance scheme for mobile robots in non-stationary environments," in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Nice France), pp. 194–199, May 1992.

[3] Q. Zhu, "A stochastic algorithm for obstacle motion prediction in visual guidance of robot motion," *The IEEE International Conference on Systems Engineering*, pp. 216–219, 1990.

[4] R. Kalman and R. Bucy, "New results in linear filtering and prediction theory," *Journal of Basic Engineering ASME Transactions*, vol. 83, pp. 95–107, 1960.

[5] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, pp. 46–57, 1989.

[6] S. Tadokoro, M. Hayashi, Y. Manabe, Y. Nakami, and T. Takamori, "Motion planner of mobile robots which avoid moving human obstacles on the basis of stochastic prediction," in *IEEE International Conference on Systems, Man and Cybernetics*, pp. 3286–3291, 1995.

[7] E. Kruse, R. Gutsche, and F. Wahl, "Estimation of collision probabilities in dynamic environments for path planning with minimum collision probability," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 1288–1295, 1996.

[8] K. Tanaka, "Detecting collision-free paths by observing walking people," in *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Lausanne , Switzerland), pp. 55–60, 2002.

[9] E. Kruse, R. Gusche, and F. M. Wahl, "Acquisition of statistical motion patterns in dynamic environments and their application to mobile robot motion planning," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 713–717, 1997.

[10] M. Bennewitz, W. Burgard, and S. Thrun, "Learning motion patterns of persons for mobile service robots," in *Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3601–3606, 2002.

[11] S. Gaffney and P. Smyth, "Trajectory clustering with mixtures of regression models," Tech. Rep. 99-15, University of Californa, Irvine, 1999.

[12] N. Dempster, A.and Laird, , and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 9, no. 1, pp. 1–38, 1977. Series B.

[13] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, pp. 265–322, September 1999.

[14] L. Kaufman and P. J. Rousseeuw, *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series In Probability And Mathematical Statistics, John Wiley and Sons, Inc., 1989.

[15] B. King, "Step-wise clustering procedures," *Journal of the American Statistical Association*, vol. 69, pp. 86–101, 1967.

[16] T. Hofmann and J. M. Buhmann, "Pairwise data clustering by deterministic annealing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 1–14, 1997.

[17] C. Fraley and A. Raferty, "How many clusters? which clustering method? answers via model-based cluster analysis," Tech. Rep. 329, University of Washington.

[18] S. Kirkpatrick, C. Gelatt, and M. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[19] V. Cerny, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *Journal of Optimization Theory and Applications*, vol. 45, pp. 41–51, 1985.