Incremental Learning of Statistical Motion Patterns with Growing Hidden Markov Models

Dizan Alejandro Vasquez Govea, Thierry Fraichard, Christian Laugier

Incremental Learning of Statistical Motion Patterns with Growing Hidden Markov Models*

Dizan Vasquez[1], Thierry Fraichard[2] and Christian Laugier[2]

[1] ASL, Swiss Federal Institute of Technology Zurich vasquez@mavt.ethz.ch
[2] LIG/CNRS,INRIA Rhone-Alpes France first_name.last_name@inrialpes.fr

Summary. Modeling and predicting human and vehicle motion is an active research domain. Due to the difficulty of modeling the various factors that determine motion (*eg* internal state, perception, etc.) this is often tackled by applying machine learning techniques to build a statistical model, using as input a collection of trajectories gathered through a sensor (*eg* camera, laser scanner), and then using that model to predict further motion. Unfortunately, most current techniques use off-line learning algorithms, meaning that they are not able to learn new motion patterns once the learning stage has finished. In this paper, we present an approach which is able to learn new motion patterns incrementally, and in parallel with prediction. Our work is based on a novel extension to Hidden Markov Models called Growing Hidden Markov models.

Introduction

Predicting the trajectories that vehicles and pedestrians are going to follow in a given environment is fundamental for effective autonomous navigation in cities, parking lots and highways. The main difficulty lies in the fact that these objects move according to a diversity of complex factors – such as their intentions and internal state – which are very difficult to model and parametrize. Thus, instead of explicitly modeling these factors, the preferred approach in the literature assumes that objects tend to follow typical motion patterns; hence, if those patterns are known, it is possible to use them to not only to predict further motion but also, for example, for detecting anomalous behavior, or improving visual tracking.

In practice, former knowledge about motion patterns is seldom available *a priori* and it should be obtained by applying machine learning techniques to motion data obtained through some kind of sensor system. For example: Bennewitz et al.

* This work has been partially supported by the European BACS Project and by a Mexican CONACYT scholarship. We would like to thank Professors Roland Siegwart, Michel Devy and Wolfram Burgard for their insightful comments on this work. Finally, we want to thank Dr. Hannah Dee and the University of Leeds for kindly sharing their experimental data with us.

[1] use the expectation-maximization algorithm to cluster trajectory data gathered with a laser scanner; and Hue et al. [3] apply a two-pass hierarchical clustering algorithm to find patterns on the output of a visual tracker.

In spite of being quite diverse, most motion pattern learning approaches share the significant drawback that they operate off-line, which implies the assumption that at least one example of every possible motion pattern is contained in the learning data set. Given the enormous variety of possible human behaviors, this assumption does not hold in practice and the learned motion models have, in the best case, only limited utility.

It would be better to learn motion patterns incrementally, so that, when a new motion pattern is observed, the system is able to integrate it into its knowledge base. This paper describes such an approach: it incrementally learns motion patterns and, at the same time, uses its current knowledge to predict motion. Our approach develops further our previous work [12] by proposing a unified extension to Hidden Markov Models (HMM)[9], a probabilistic framework which is very popular in the motion pattern learning literature [*eg* 13, 7, 1]. This extension, known as Growing HMM (GHMM) enables incremental and on-line learning of the parameters and the structure of the model.

The rest of this paper is structured as follows: section 1 provides an overview of motion pattern learning using HMMs; sec. 2 presents Growing Hidden Markov Models; in section 3 the application of GHMMs to our particular problem is discussed; our experimental results are outlined in sec. 4; finally, we present our conclusions in section 5.

1 Motion Pattern Learning with HMMs

The literature of motion pattern learning includes a wide variety of approaches like the use of multilayer Self-organizing networks proposed by Johnson and Hogg [5] in their pioneering paper, or the use of co-occurrence statistics proposed by Stauffer and Grimson [10]. In this section we will focus on techniques based on Hidden Markov Models, which is the case of many more recent approaches[3]. Due to lack of space, our discussion of HMMs will be rather summary and heavily biased towards our application, the interested reader may refer to the papers by Juang et al. [6] and Rabiner [9] for a deeper introduction.

In the context of our problem, an HMM (see fig. 1(a)) may be seen as a graph whose nodes represent states attainable by the object (*eg* places in the environment) and whose edges represent transitions between states. The system is supposed to be at a given state and to evolve stochastically at discrete time steps by following the graph edges according to a transition probability $P(S_t|S_{t-1})$. Moreover, the object's state is not directly observable, instead, it should be measured through some kind of sensor reading (*ie* observation) which is related to the actual state

[3] See [3] for a more comprehensive review of the literature.

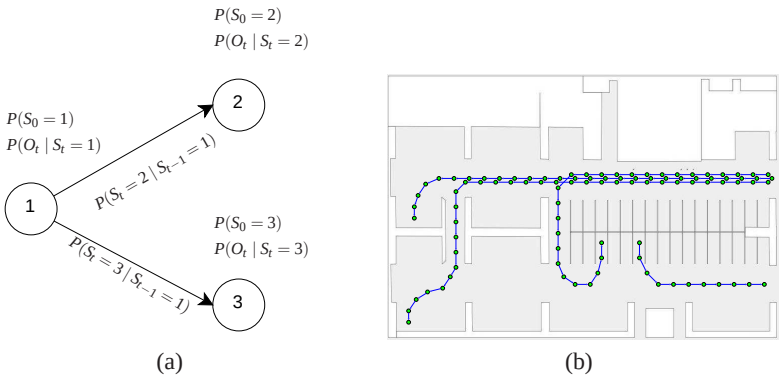


Fig. 1. a) A basic three-state HMM; b) HMM Structure embedded in a parking (only a few motion patterns are displayed)

through an observation probability $P(O_t|S_t)$. Often, the initial state of the system is represented stochastically with a state prior $P(S_1)$.

HMM learning is composed of two sub-tasks:

- *Structure learning:* Determines the number of nodes (*ie* discrete states) in the model, as well as the edge structure for the graph.
- *Parameter learning:* Estimates the parameters for the three probability distributions (state prior, transition and observation probabilities) from data.

Different algorithms for structure and parameter learning exist in the literature, it is the choice of these algorithms what distinguishes different HMM based motion pattern learning approaches. For example, Walter et al. [13] assume that motion patterns are known *a priori* and define the structure as an equal number of chain-like structures, then, parameters are learned using the expectation-maximization algorithm; Bennewitz et al. [1] learn the HMM structure by clustering trajectory data with the expectation-maximization algorithm, and then manually set the model's parameters according to assumptions about object's motion; Makris and Ellis [7] learn the HMM structure in a similar way, but also incorporate parameter learning into the algorithm.

Despite their differences, all these approaches have some points in common: a) typical motion patterns are represented with some sort of trajectory prototype; b) structure learning is independent of parameter learning; and c) learning is first performed off-line and then the system switches to an utilization stage where no further learning is performed. As we will see in the following sections, our approach behaves differently with respect to these points.

2 Growing Hidden Markov Models

In this section we present our proposed extension to HMMs[4]: Growing Hidden Markov Models; which may be described as time-evolving HMMs with continuous observation variables, where the number of states, structure and probability parameters are updated every time that a new observation sequence is available.

Our approach is designed for its utilization as an approximate inference tool for continuous state spaces. It applies to problems where the continuous state space may be discretized into a finite number of regions, so that every such region is represented by a discrete state in the GHMM. It is also assumed that the state evolves continuously and that there is a topological equivalence between state and observation spaces (*ie* states which are near from each other correspond to observations which are also near from each other).

The key intuition behind GHMMs is that the structure of the model should reflect the spatial structure of the state space discretization, where transitions between states are only allowed if the corresponding regions are neighbors. Therefore, structure learning consists basically in estimating the best space discretization from data and identifying neighboring regions. We have addressed this problem by building a *topological map* of the environment using the Instantaneous Topological Map (ITM) algorithm [4]. For parameter learning, we basically have adapted the approach proposed by Neal and Hinton [8] in order to deal with variable state cardinality and continuous observations.

2.1 Probabilistic model

Formally, GHMMs are defined in terms of three variables:

- S_t, S_{t-1}, the current and previous states, which are discrete variables with value $S_t, S_{t-1} \in \{1, \cdots, N_k\}$ for a fixed N_k, which is the number of states in the model after k observation sequences have been processed[5].
- O_t, the observation variable, which is a multidimensional vector in $\mathbb{R}^M$.

The joint probability decomposition (JPD) for HMMs is:

$$P(S_{t-1}\ S_t\ O_t) = \underbrace{P(S_{t-1})}_{\textit{state prior}}\ \underbrace{P(S_t \mid S_{t-1})}_{\substack{\textit{transition}\\ \textit{probability}}}\ \underbrace{P(O_t \mid S_t)}_{\substack{\textit{observation}\\ \textit{probability}}} \tag{1}$$

[4] Since space is limited, we have opted for providing a general overview which omits some specific information on optimizations and data structures. The interested reader is referred to [11] for more details.

[5] For the sake of notational simplicity, we will often omit the k hereafter, nevertheless, it should be noted that parameters and structure changes with every new observation sequence. Also, notation $O_{1:t}$ will be used as a shortcut for the variable conjunction $O_1\ O_2\ \cdots\ O_{t-1}\ O_t$.

Where the state prior is computed recursively from the previous time step:

$$P(S_{t-1}) = P(S_{t-1} \mid O_{1:t-1}) \tag{2}$$

Both the observation and transition probabilities are assumed to be *stationary*, that is, independent of time:

$$P(O_i \mid S_i) = P(O_j \mid S_j) \quad \forall\, i,\ j \in \{1, \cdots, T\} \tag{3}$$
$$P(S_i \mid S_{i-1}) = P(S_j \mid S_{j-1}) \quad \forall\, i,\ j \in \{2, \cdots, T\} \tag{4}$$

thus, the parametric forms of the three probabilities in the JPD are the same, irrespectively of the value of the time variable:

- $P(S_0 = i) = \pi_i$. The state prior will be represented as a vector $\pi = \{\pi_1, \cdots, \pi_N\}$ where each element contains the prior probability for the corresponding state.
- $P([S_t = j] \mid [S_{t-1} = i]) = a_{i,j}$. Transition probabilities are represented with a set of variables A, where each element $a_{i,j}$ represents the probability of reaching the state j in the next time step given that the system is currently in state i.
- $P(O_t \mid [S_t = i]) = \mathbf{G}(O_t;\ \mu_i,\ \Sigma)$. The observation probability will be represented by a Gaussian distribution for every state, having a single covariance matrix Σ. The set of all the Gaussians' parameters will be denoted by $b = \{\Sigma, \mu_1, \cdots, \mu_N\}$.

The full set of parameters for a GHMM is denoted by $\lambda = \{\pi, A, b\}$.

Besides its time-evolving nature, GHMMs are defined by their learning algorithm, which processes complete observations sequences as they arrive. The general steps of the algorithm are depicted in fig. 2 and are detailed in the following subsections.

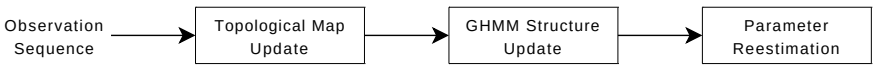


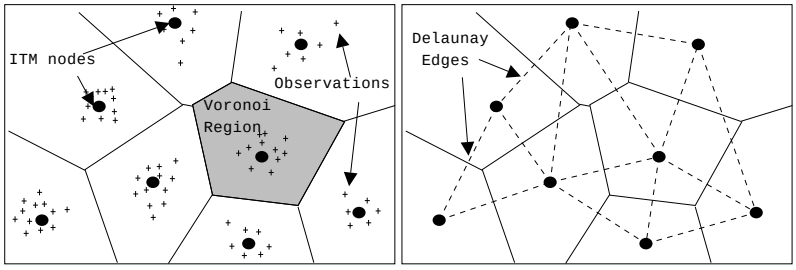
Fig. 2. Overview of the GHMM learning algorithm.

2.2 Updating the Topological Map

Our structure learning approach is based on the construction of a topological map: a discrete representation of state space in the form of a graph where nodes represent regions, and edges connect contiguous nodes. Every node i has an associated weight vector w_i, corresponding to the region's centroid.

The topological map is updated for every available observation O_t using the ITM algorithm which has the following properties:

- It minimizes the number of nodes while trying to keep the same average distance between neighbors.
- Has linear time and memory complexity with respect to the number of nodes.
- Edges are a subset of the Delaunay triangulation, meaning that they can exist only between nodes representing adjacent Voronoi[6] regions (fig. 3).



(a) Nodes and regions (b) Edges

Fig. 3. Example ITM space decomposition

The ITM algorithm consists of the following steps (c.f. [4]):

1. **Matching**: find the nearest b and second nearest s nodes to O_t. We use Mahalanobis distance with the same σ than observation probabilities.
2. **Weight adaptation**: move w_b towards O_t by a small fraction $\Delta_b = \varepsilon(O_t - w_b)$.
3. **Edge adaptation**: a) create an edge connecting b and s unless that edge exists; b) for every neighbor m of b check if O_t lies in the Thales sphere going through w_m and w_b and delete the corresponding edge if that is the case. Delete any node which has no neighbors.
4. **Node adaptation**: a) if O_t lies outside the Thales sphere going through w_b and w_s and its distance from w_b is greater than a given threshold τ, create a new node n with $w_n = O_t$. Connect nodes b and n. Remove node s if it's distance from b is smaller than $\frac{\tau}{2}$.

2.3 Updating the Model's Structure

Changes in the topological map are reflected in the GHMM structure as follows:

1. For every new node i in the topological map, add a corresponding node in the GHMM, initializing its prior to a preset value: $\pi_i = \pi_0$. Do the same for the self-transition probability: $a_{i,i} = a_0$.

[6] The Voronoi region associated with a node is defined by the set of all the points which are closer to that node's centroid than to any other centroid in the graph. Delaunay edges link the centroids of Voronoi regions that have a common border.

2. For every new edge (i, j) in the topological map, initialize the corresponding transition probabilities to a preset value: $a_{i,j} = a_0$ and $a_{j,i} = a_0$.
3. For every deleted node and edge in the topological map, set to zero (*ie* delete) the corresponding state prior and transition probabilities.
4. For every added or modified weight w_i, set the corresponding Gaussian mean value: $\mu_i = w_i$.

2.4 Reestimating Parameters

Parameter learning is performed immediately after structure learning. It reestimates the parameters using an incremental version of the Baum-Welch algorithm based on the work from Neal and Hinton [8]. Since all of the observation probabilities' mean values have been assigned in §2.3 and their covariance Σ is fixed, only the state prior and transition probabilities are reestimated. Computations are based on the sum of expected state and transition count values, which are kept in two sets of temporal variables π' and a'. Then, the actual probabilities are obtained by normalization:

1. Precompute forward (α_i), backward (β_i) and joint observation probabilities (p_O) for the observation sequence $O_{1:T}$.
2. For every discrete state i in the GHMM, reestimate the state prior:

$$\pi'_i = \pi'_i + \frac{\alpha_1(i)\ \beta_1(i)}{P_O K} \tag{5}$$

$$\pi_i = \frac{\pi'_i}{\sum_{j=1}^{N} \pi'_j} \tag{6}$$

3. Reestimate every non-zero transition probability in the GHMM:

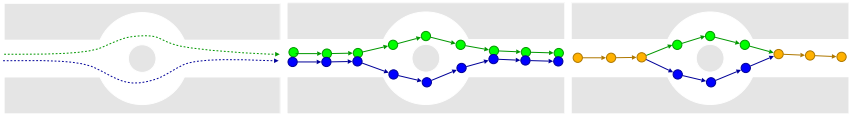
$$a'_{i,j} = a'_{i,j} + \frac{\sum_{t=2}^{T} \alpha_{t-1}(i) p([S_t = j \mid [S_{t-1} = i]\ \lambda) p(O_t \mid [S_t = j]\ \lambda) \beta_t(j)}{\sum_{t=2}^{T} \alpha_{t-1}(i) \beta_{t-1}(i)} \tag{7}$$

$$a_{i,j} = \frac{a'_{i,j}}{\sum_{k=1}^{N} a'_{i,k}} \tag{8}$$

3 Learning and Predicting Motion with GHMMs

Having presented GHMMs, this section focuses in their concrete application to learning and predicting the motion of vehicles and pedestrians. This application is based on the key observation that, often, objects move in function of their intention to reach a particular state (*ie* its goal). Accordingly, we model the object's motion in terms of an augmented state vector, composed of two sets of variables describing its *current* and *intended* state, respectively.

Due to the fact that our model is goal-oriented, in our approach a motion pattern is no longer a trajectory prototype, but a directed graph indicating all the possible ways in which a goal may be reached (fig. 4).



(a) Input trajectories (b) Trajectory based (c) Goal based

Fig. 4. Pattern representations generated from input data: (b) trajectory prototypes; (c) single graph (the goal is the rightmost node).

3.1 Notation and Base Assumptions

We assume that tracking data is available as a collection of observation sequences (*ie* trajectories). Every individual sequence $O^k_{1:T^k} = \{O_1, \cdots, O_{T^k}\}$ corresponds to the tracker's output for a single object and its observations are evenly spaced on time. Different observation sequences may have different lengths T^k.

In the rest of this section, we assume that the state of an object is defined by its position (x,y). It should be noted, however, that our approach is applicable to spaces of arbitrary dimensions. The augmented state of the object consists of its current and intended position $(x,y,\hat{x},\hat{y})$.

We assume that observations are also expressed in terms of the object's coordinates $O_t = (x_t, y_t)$. Since learning is performed on the basis of complete observation sequences, we assume that the last observation $O_T = (x_T, y_T)$ of each sequence corresponds to the object's goal. Hence, it is posible to build an augmented observation sequence, which constitutes the actual input to our algorithm:

$$\bar{O}_{1:T} = \{(x_1, y_1, x_T, y_T), (x_2, y_2, x_T, y_T), \cdots, (x_T, y_T, x_T, y_T)\}$$

3.2 Probabilistic Model

Since our approach is based on GHMMs, it uses the same probabilistic model that has been described in §2.1. Nevertheless, we also need to distinguish between current and intended components of the state. Thus, we will decompose the observation variable into a current O'_t and intended O''_t components: $O_t = [O'_t, O''_t]$.

In order to define the JPD, we will assume that the current and intended components of observations are conditionally independent given the current state, which allows us to rewrite the observation probability as:

$$P(O_t|S_t) = P(O'_t\ O''_t|S_t) = P(O'_t|S_t)P(O''_t|S_t) \tag{9}$$

and the whole JPD as:

$$P(S_{t-1}\ S_t\ O'_t\ O''_t) = P(S_{t-1})P(S_t|S_{t-1})P(O'_t|S_t)P(O''_t|S_t) \tag{10}$$

Since the observation probability is now written as a product of probabilities, $P(O'_t\ O''_t|S_t) = P(O'_t|S_t)P(O''_t|S_t)$ we need to define their parametric forms:

$$P(O'_t \mid [S_t = i]) = \mathbf{G}(O'_t;\, \mu'_i,\, \Sigma') \tag{11}$$

and:

$$P(O''_t \mid [S_t = i]) = \begin{cases} \mathbf{U}_{O''_t} & \text{if } O''_t \text{ is not available} \\ \mathbf{G}(O''_t;\, \mu''_i,\, \Sigma'') & \text{otherwise} \end{cases} \tag{12}$$

where μ'_i and μ''_i are the mean values of the current and goal components for state i; and Σ' and Σ'' are the respective values of the covariance matrix for all the states.

By noting that $P(O_t \mid S_t)$ is either a product of Gaussians, or a product of a constant and a Gaussian, we may rewrite this probability as a single Gaussian:

$$P(O_t \mid [S_t = i]) = \frac{1}{Z}\mathbf{G}(O_t;\, \mu_i,\, \Sigma) \tag{13}$$

where $\mu_i = [\mu'_i, \mu''_i]$, and Σ is a a block diagonal matrix[7] having the form:

$$\Sigma = \begin{bmatrix} \Sigma' & 0 \\ 0 & \Sigma'' \end{bmatrix} \tag{14}$$

and Z is a normalization variable, which permits to compute the uniform on the goal component using the same Gaussian representation. Since, during prediction, the intended part of observation is not available, this is done by setting[8] $O''_t = 0$.

3.3 Prediction

We have not yet discussed prediction, which can be performed using the same algorithms that are used for standard HMMs, whithout interfering with learning. This is possible because learning takes place immediately after a trajectory has finished and, thus, it does not affect prediciton in any way. For our particular case, we have chosen to apply exact inference:

For every new observation, the current belief state for the object is reestimated using:

$$P(S_t \mid O_{1:t}) = \frac{1}{Z}P(O_t|S_t) \sum_{S_{t-1}} [P(S_t|S_{t-1})P(S_{t-1}|O_{1:t-1})] \tag{15}$$

where $P(S_{t-1} \mid O_{1:t-1})$ comes from the state estimation for the previous time step (or from the state prior, in the case of the first observation in a sequence). Then, prediction is peformed by propagating this estimate H time steps ahead into the future using:

$$P(S_{t+H} \mid O_t) = \sum_{S_{t+H-1}} P(S_{t+H}|S_{t+H-1})P(S_{t+H-1}|O_t) \tag{16}$$

[7] A block diagonal matrix is a square diagonal matrix in which the diagonal elements are square matrices of any size and the off-diagonal elements are zero.

[8] It is easy to show that this is equivalent to a multiplication by a constant, and – when normalized – becomes effectively equivalent to the uniform in (12).

4 Experimental Results

We have implemented our approach and conducted extensive experiments using several real and synthetic data sets. In this section we will discuss some of the results we have obtained on two of these data sets (fig. 4): a) real data obtained through a visual tracker in a parking environment and then corrected by hand, as described in [2]; and b) synthetic data, generated by a simulator and having the particularity that it includes velocity information in observations.

Both data sets are sampled at 10 Hz, and the tests have been executed in a 512 MB Athlon XP 2800 computer running Linux.

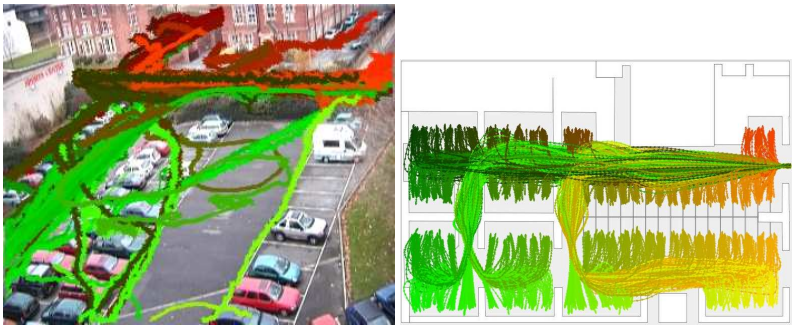


Fig. 5. Data sets: left) real data; right) synthetic data.

4.1 Measuring Prediction Accuracy

We have evaluated our prediction results using the average error for a complete data set containing K observations sequences. This has been computed as the expected distance between the predicted position for a time horizon H and the effective observation O_{t+H}:

$$\langle E \rangle = \frac{1}{K} \sum_{k=1}^{K} \frac{1}{T^k - H} \sum_{t=1}^{T^k - H} \sum_{i \in \mathcal{S}} P([S_{t+H} = i] \mid O^k_{1:t}) \| O^k_{t+H} - \mu_i \|^{1/2} \qquad (17)$$

4.2 Model size vs. Prediction Accuracy

Figures 6(a) and 6(c) show the model size (number of edges) and the average prediction error as a function of the total number of processed trajectories, for the real and simulated environments, respectively. As one could expect, the average error decreases as more trajectories have been processed and, at the same time,

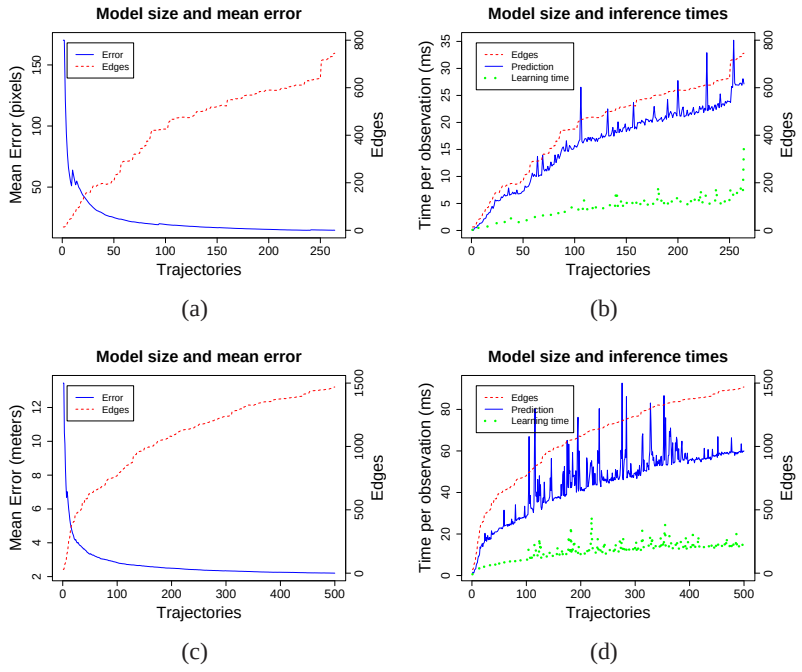


Fig. 6. Error and computation times. First row: real data. Bottom row: synthetic data.

the model's growth quickly decelerates as existing knowledge covers more of the observed motion patterns. In the case of real data, it is also possible to see some sudden jumps in the size of the model that correspond to the addition of motion patterns that occur in regions where no motion has been observed previously or that differ significantly from all the rest.

4.3 Processing time vs. Prediction Accuracy

Figures 6(b) and 6(d) plot the time taken by prediction (middle line) and learning (lower line) with respect to the number of processed trajectories. The model size (upper line) is also given as a reference.

As it may be expected, time seems to depend linearly on the model size. Moreover, prediction times are below 25 ms per observation for real data and 60 ms for simulated data. This is explained in part by the fact that the simulated environment is much bigger than the real one. Even in the case of real data, prediction times are quite close to full camera frame rate.

Lastly, we may observe that there are sharp peaks in processing times which have been caused by some unrelated processes that have been executed concurrently with our tests.

5 Conclusions

We have developed a novel extension to HMMs which is able to learn both the models parameters and structure incrementally. We have applied this extension to vehicle and pedestrian motion by defining an augmented state which adds the intended goal to the classic state variables. We have validated our approach using real and synthetic data, the obtained results show quite good prediction accuracy and real-time applicability. Moreover, our approach improves over other HMM based techniques by implementing a model –even if rather crude– of the object's intentions.

References

1. M. Bennewitz, W. Burgard, G. Cielniak, and S. Thrun. Learning motion patterns of people for compliant robot motion. *Internationl Journal of Robotics Research*, 24(1):31–48, January 2005.
2. H.-M. Dee. *Explaining Visible Behaviour*. PhD thesis, University of Leeds, 2005.
3. W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank. A system for learning statistical motion patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1450–1464, September 2006.
4. J. Jockusch and H. Ritter. An instantaneous topological map for correlated stimuli. In *In Proc. of the International Joint Conference on Neural Networks*, volume 1, pages 529–534, Washington (US), July 1999.
5. N. Johnson and D. Hogg. Learning the distribution of object trajectories for event recognition. In *Proc. of the British Machine Vision Conference*, volume 2, pages 583–592, September 1995.
6. B.-H. Juang, S. E. Levinson, and M. M. Sondhi. Maximum likelihood estimation for multivariate mixture observations of markov chains. *IEEE Transactions on Information Theory*, 32(2):307–309, March 1986.
7. D. Makris and T. Ellis. Spatial and probabilistic modelling of pedestrian behavior. In *Proc. of the British Machine Vision Conference*, pages 557–566, Cardiff, UK, 2002.
8. R. M. Neal and G. E. Hinton. A new view of the em algorithm that justifies incremental, sparse and other variants. In M. I. Jordan, editor, *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.
9. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.
10. C. Stauffer and E. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, August 2000.
11. D. Vasquez. *Incremental Learning for Motion Prediction of Pedestrians and Vehicles*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, FR, February 2007.
12. D. Vasquez and T. Fraichard. Intentional motion on-line learning and prediction. In *Field and Service Robotics*, Port Douglas, Australia, July 2005.
13. M. Walter, A. Psarrow, and S. Gong. Learning prior and observation augmented density models for behaviour recognition. In *Proc. of the British Machine Vision Conference*, pages 23–32, 1999.