

Approximate Learning and Inference for Tracking with Non-overlapping Cameras

Wojciech Zajdel, Ben Kröse
University of Amsterdam,
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
{wzajdel,krose}@science.uva.nl

ABSTRACT

Tracking with multiple cameras requires partitioning of observations from various sensors into trajectories. In this paper we assume that the observations are generated by a hidden, stochastic 'partition' process and propose a hidden Markov model (HMM) as a generative model for the data. The state space for the hidden variable is intractable, so the inference and learning in our HMM are based on approximate representation of the distribution on this state space. The proposed approximation truncates the distribution from unlikely states. We test our method on real observations; by tracking people in a university building. The tests show that the described approach is an useful alternative to the existing approximate methods.

KEY WORDS

Probabilistic Reasoning, Tracking, Hidden Markov Model

1 Introduction

In this paper we propose a method for tracking multiple persons with asynchronous, non-overlapping cameras. The task is important for surveillance of wide areas (like shopping centers or large office buildings) that cannot be fully observed with sensors.

The problem was first studied by Huang and Russell in [1] who argued why the 'standard' tracking techniques ([2]) are not applicable for non-overlapping sensors. In a typical tracking setup all objects move within the fully observed area. Tracking requires association of objects' observations at one camera frame with the corresponding observations from the subsequent frames. Most methods exploit smooth position changes of an object between neighboring frames, what allows to use simple motion models (e.g. Kalman filter) for tracking. When the sensors do not overlap the time or position continuity between observations cannot be assumed. Moreover, the observations arrive one at a time.

Huang and Russell [1] developed a method that used two distant cameras to track cars moving in one direction on a motorway. Association of an object observed at one camera with the same object at the other camera was based on probabilistic appearance models. Pasula *et al.* in [3] showed that the appearance models do not scale to prob-

lems with more than two cameras. To model probabilistic dependencies among the observations of the same object Pasula *et al.* saw these observations as a sequence generated by a hidden Markov model (HMM). Tracking was achieved by random proposing of sequences and evaluating the probability that such sequence could be generated by a single HMM. Here the problem stem from an intractable number of possible sequences that can be built out of a set of the all available data. Pasula *at al.* provided a polynomial-time approximation scheme by sampling possible trajectories with Markov chain Monte Carlo (MCMC) method ([4]).

There are two limitations of the MCMC-based approach. First, MCMC technique can approximate only some numerical function of a trajectory (e.g. travel time) but not the trajectory. Second, sampling is effective with an additional assumption that the objects move in one direction and visit a sensor only once (motorway). Otherwise, the association state space becomes much larger, therefore difficult to approximate quickly with samples.

In this paper we describe a method that overcomes the above limitations. We are tracking people that move without constraints in a building. Our method is intended to return the trajectories of the observed people. We use a single HMM as a generative model for the all observations (section 3), and describe approximate inference and learning algorithms (section 4). In section 5 we demonstrate experiments that compare our method with the MCMC-based algorithm.

2 Assumptions

There are M non-overlapping cameras distributed in a building. We want to recover sequences of camera locations visited by the same person. Whenever a person makes a complete pass through a camera viewing field we get an observation y_i , which provides only a summarized description of the passing person. An observation is made of the following components: $y_i = \{a_i, t_i, l_i, e_i, d_i\}$. The term a_i contains summarized appearance information (e.g. average color) computed from a single or multiple frames where a person was observed during the pass. The term t_i is the time of pass (a single moment); the term l_i is the camera location, where the pass happened. The terms e_i and d_i in-

dicating the frame border through which a person entered and departed from the viewing field. (These can be left, right, top, down.)

Given the set of N observations $Y = \{y_{1:N}\}$ we want to identify which observations describe the same person. We are looking for a partition of observations $y_i \in Y$ into several trajectories Y_k . A trajectory is a subset $Y_k \subset Y$, believed to collect the observations that correspond to the same individual. It is also assumed that the observations $y_i \in Y$ are time-ordered: $t_i < t_j$ iff $i < j$.

3 Generative model of observations

We reason about trajectories and observations in a probabilistic framework. The available data Y are assumed to be a probabilistic (noisy) observation of some underlying process that defines the trajectories.

To build a generative model we notice that the observations are gradually added to the set Y . Let $Y^{(n)}$ be a set of the first n observations: $Y^{(n)} = \{y_{1:n}\}$. When a new observation arrives, there is a new set constructed $Y^{(n+1)} = \{y_{1:n+1}\}$. Thus, before the total set $Y^{(N)} = Y$ is reported we deal with a sequence of intermediate sets: $Y^{(n)}, n = 1, \dots, N$. These sets share most of the elements, nevertheless we treat each $Y^{(n)}$ as a separate item.

Now, consider a discrete-valued variable ω_n that defines a 'scenario' telling how to gather the first n observations. The variable ω_n defines the number of observed persons (denoted K_{ω_n}) and sets the association of the observations $y_i \in Y^{(n)}$ with the persons. The state space $\Omega^{(n)}$ for variable ω_n is the discrete space of all possible partitions of n observations into any number of trajectories. Accordingly, variable ω_n will also be called a partition.

Variable ω_n decides about grouping of observations $y_i \in Y^{(n)}$ into trajectories, but not about the actual content (appearance, locations, etc.) reported by $y_i \in Y^{(n)}$. It is proposed that the contents are filled by a probabilistic process depending only on the current partition: $p(Y^{(n)} | \omega_n)$. By the time when observation y_{n+1} is reported, the variable ω_n is assumed to have evolved into a new partition variable ω_{n+1} . To reason about this evolution, we assume that ω_n is a stochastic process. The variable ω_{n+1} depends only on the directly preceding ω_n according to some probability $p(\omega_{n+1} | \omega_n)$.

The above propositions lead to the definition of a hidden Markov model (HMM) ([5]). There is a hidden partition process ω_n , which when observed yields data $Y^{(n)}$. The initial state (partition) ω_1 involves only one observation and is deterministic. Note, that our HMM considers a single, hidden partition process, contrary to the approach of Pasula *et al.* who used a static partition variable, and defined a separate HMM for every trajectory. In the rest of this section we complete the definition of HMM, by setting first the transition model, and then the observation model.

3.1 Transition model for hidden process

The transition model for our hidden process says how the partition variable is evolving with time. The evolutions are defined by a discrete probability mass function $p(\omega_{n+1} | \omega_n)$. We have to assign a probability to every possible partition $\omega \in \Omega^{(n+1)}$ of $n+1$ observations from $Y^{(n+1)}$. This probability is conditioned on the fact that the observations from $Y^{(n)}$ were partitioned according to ω_n . Partition ω_n assumed K_{ω_n} trajectories. We propose the following process:

$$p(\omega_{n+1} | \omega_n) = \begin{cases} \frac{1}{K_{\omega_{n+1}}} & \text{iff } \omega_{n+1} \text{ extends } \omega_n \\ 0 & \text{otherwise} \end{cases}$$

A partition ω_{n+1} is said to *extend* the partition ω_n in two cases. The first is when ω_{n+1} assigns the new observation y_{n+1} to one of the K_{ω_n} trajectories proposed by ω_n . This can be done in K_{ω_n} ways. The second case is when ω_{n+1} creates a new trajectory for the new observation, while keeping the existing trajectories of ω_n unchanged. Thus, there are $K_{\omega_n} + 1$ ways to create a partition ω_{n+1} such that it has a non-zero probability. Consequently, the constant $\frac{1}{K_{\omega_{n+1}}}$ assures normalization of the model. We note that the above stochastic process is just one of the possible choices. It is an arbitrary model, used because of its simplicity and lack of parameters.

3.2 Observation model

The observation model defines a probability density function (pdf) $p(Y^{(n)} | \omega_n)$. It tells how likely are the observations $Y^{(n)}$ under the assumption that persons were moving according to a partition ω_n . The fact that ω_n is a partition of $Y^{(n)}$ into K_{ω_n} trajectories is expressed by:

$$Y^{(n)} = Y_1^{(n)} \cup \dots \cup Y_{K_{\omega_n}}^{(n)}, \quad (1)$$

where the trajectory $Y_k^{(n)}$ is a set of observations of the k th person. We assume that the people in a building move independently of each other. This allows for a convenient factorization of the sought pdf:

$$p(Y^{(n)} | \omega_n) = \prod_{k=1}^{K_{\omega_n}} p(Y_k^{(n)} | \omega_n). \quad (2)$$

Now, we focus on a subset of observations $Y_k^{(n)} = \{y_{k_1} \dots y_{k_N}\}$ assumed to correspond to a single person. The goal is to derive the joint probability of observations in this set $p(y_{k_1}, \dots, y_{k_N} | \omega_n)$. To find the joint probability of this set we use a dynamic Bayes network (DBN) as described in detail in [6]. A DBN ([7]) exploits independences among a large set of variables in order to express the joint probability of these variables as a product of simpler conditional models.

Keeping in mind that each observation has several components, consider a dynamic Bayes network of Fig.1.

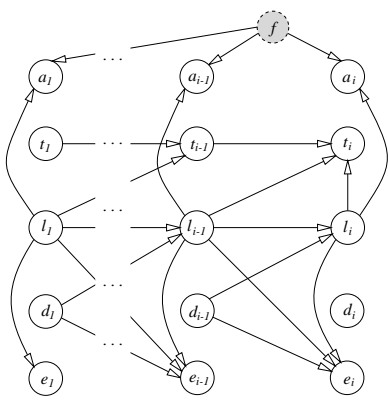


Figure 1. An observation model of an object. The gray node describes an object’s hidden, time-invariant appearance. The arcs show dependencies between the variables.

Each column of white nodes is a single observation y_i . The gray node is a hidden variable that was introduced to describe the time-invariant, intrinsic object’s properties responsible for its appearance. The definition of a BN is complete with a conditional pdf for every vertex conditioned on its parents, and a prior pdf for vertices that do not have parents.

$p(a_i|f, l_i)$; Object’s appearance at a camera depends on the intrinsic properties of the object, represented by variable f and environment at location l_i . Our (heuristic) choice for this model is a simple Gaussian distribution: $\mathcal{N}(a_i; f, S(l_i))$. In the experiments with real world data we verify the ability of such simple distribution to capture the noise present in the camera images. The parameters $S(l_i)$ are unknown and will be learned.

$p(l_i, e_i|l_{i-1}, d_{i-1})$; Object’s location and entry side depend only on the past location and the past departure side. This dependency assumes a Markov relation between visited locations. We specify this discrete distribution using prior knowledge of the building’s layout.

$p(t_i|l_{i-1}, l_i, t_{i-1})$; The time of appearing at location l_i knowing that the object moved from l_{i-1} at time t_{i-1} . We use a truncated normal distribution: $p(t_i|l_{i-1}, l_i, t_{i-1}) = \mathcal{N}(t_i; \delta(l_i, l_{i-1}) + t_{i-1}, R(l_i, l_{i-1}))$. The parameters $\delta(l_i, l_{i-1})$ and $R(l_i, l_{i-1})$ tell the expected transition time between the two locations and the uncertainty of this distribution. These need to be learned.

$p(f)$; Prior distribution of the intrinsics of any object. We use a normal density: $p(f) = \mathcal{N}(f; \mu, R)$. The parameters μ and R will be learned.

$p(l_1, e_1)$; Distribution on locations in a building where a trajectory may start. Given by the user.

$p(d_i), p(t_1)$; Prior probability of observing departure side d_1 and entering the building at time t_1 . We assume that these are uniform, so both terms become scaling factors.

The joint probability of the variables in the graph is the product of pdf’s associated with every vertex. The variable f is unknown, thus we integrate it in the final expres-

sion. The scaling terms are omitted:

$$p(Y_k^{(n)}|\omega_n) = p(y_{k_1}, \dots, y_{k_N}|\omega_n) = \int_f p(f) \prod_{i=k_1}^{k_N} p(a_i|f, l_i) \prod_{i=k_2}^{k_N} [p(l_i, e_i|l_{i-1}, d_{i-1})p(t_i|t_{i-1}, l_i, l_{i-1})] \quad (3)$$

The component densities in (3) depend on unknown parameters, therefore the expression $p(Y^{(n)}|\omega_n)$ is a function of this parameters. It is denoted as $p(Y^{(n)}|\omega_n, \Theta)$. Symbol Θ lists all parameters $\Theta = \{\mu, R, \delta(q, r), R(q, r), S(q)\}$, where q and r are camera locations.

4 Inference and Learning in HMM

HMMs provide an established framework for inference and learning [5]. In our application we are looking for posterior $p(\omega_N | Y^{(N)})$. By selecting the most probable ω_N one obtains the Maximum A Posteriori (MAP) solution to the association problem. The sought posterior density is a function of unknown parameters Θ . Learning refers to estimating these parameters. We will be looking for the parameters that maximize the likelihood of our data Y , i.e., the ML parameters.

4.1 Exact methods

Inference in a HMM is done exactly with the *forward* algorithm. The sought distribution $p(\omega_N | Y^{(N)})$ is computed by a recursive scheme:

$$p(\omega_{n+1}|Y^{(n+1)}) = \alpha p(Y^{(n+1)} | \omega_{n+1}, \Theta) \sum_{\omega_n \in \Omega^{(n)}} p(\omega_{n+1} | \omega_n) p(\omega_n | Y^{(n)}), \quad (4)$$

where α is a normalization constant. The recursions are started with $p(\omega_1 | Y^{(1)}) = 1$, since the state space $\Omega^{(1)}$ has only one element. The eq. (4) uses observation model, which depends on unknown parameters Θ . Consequently, the posterior is also a function of Θ , so we write $p(\omega_N | Y^{(N)}, \Theta)$.

In general HMMs use parametric transition and observation models. The parameters of these models are found by the Baum-Welch method, which is a special case of the approximate Expectation-Maximization (EM) algorithm. EM [8] is a general tool for estimation of ML parameters of a model with hidden variables. Although the EM is an approximate method, if the posterior on hidden variables can be computed exactly, the EM is guaranteed to find at least locally best parameters.

Our application assumed a fixed, non-parametric transition model, so we apply the EM to find only the parameters Θ of the observation model. To estimate Θ from all observed data Y it is enough to consider the model $p(Y^{(N)} |$

ω_N, Θ) since it involves all observations $Y = Y^{(N)}$. The unobserved variables are ω_N , along with the hidden variables f used by the DBNs to model the trajectories assumed by ω_N . We denote these variables as $f_1, \dots, f_{K_{\omega_N}}$. In the E step we find the posterior distribution on all hidden variables given the current parameters $\Theta^{(t)}$:

$$q(\omega_N, f_{1:K_{\omega_N}}) = p(\omega_N | Y^{(N)}, \Theta^{(t)}) \prod_{i=1}^{K_{\omega_N}} p(f_i | Y^{(N)}, \Theta^{(t)}, \omega_N). \quad (5)$$

The term $p(\omega_N | Y^{(N)}, \Theta^{(t)})$ comes from the forward algorithm executed with the current $\Theta^{(t)}$. The term $p(f_i | Y^{(N)}, \Theta^{(t)}, \omega_N)$ is the posterior density on the hidden variable of DBN modeling the i th person proposed by the ω_N (hence conditioning). These can be found by any Bayes network inference method. In the M step we update the parameters by:

$$\Theta^{(t+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega_N \in \Omega^{(N)}} \int_{f_{1:K_{\omega}}} \ln p(\omega_N, f_{1:K_{\omega}}, Y^{(N)} | \Theta) q(\omega_N, f_{1:K_{\omega}}) \quad (6)$$

For the details of solving the above optimization consult [6].

4.2 Approximations

Unfortunately, in most of the tracking problems the state space $\Omega^{(n)}$ has an intractable number of elements. Therefore the summation over the posterior distribution on $\omega_n \in \Omega^{(n)}$, as in eq. (4) and eq. (6), cannot be computed exhaustively. To make this summation computable, we propose a simple way to approximate the posterior distribution $p(\omega_n | Y^{(n)})$ over state space $\Omega^{(n)}$. The approximate distribution $p^*(\cdot)$ is:

$$p^*(\omega_n | Y^{(n)}) = \begin{cases} \gamma p(\omega_n | Y^{(n)}) & \text{iff } p(\omega_n | Y^{(n)}) > \lambda \\ 0 & \text{otherwise} \end{cases},$$

where γ is a scale term ensuring normalization, and λ is a threshold parameter. The threshold is selected every time such that exactly K partitions are preserved keeping a non-zero approximate probability. In fact, the above approximation simply truncates the distribution to the K most likely elements, therefore reducing the terms to be summed. Based on the proposed approximation we derive two data association methods.

The first is a straightforward application of the EM algorithm. Since EM requires only the posterior on the last state $\omega^{(N)}$ it is enough to do only the forward pass. After each execution of eq. (4) the computed posterior $p(\omega_n | Y^{(n)})$ is approximated with $p^*(\omega_n | Y^{(n)})$. Because of the particular form of the transition model in the next execution of (4) one needs to sum only over those

ω_{n+1} that extend the K partitions ω_n which have a non-zero posterior likelihood. When the forward pass is completed we use the approximate posterior on partitions ω_N to update the parameters. The procedure is sketched in Tab. 1. For the purpose of this paper we call it a Truncated Baum-Welch method.

```

start with random  $\Theta$ 
iterate until convergence
  iterate  $n = 1 : N$  (forward pass with  $\Theta$ )
    compute  $p(\omega_n | Y^{(n)})$  with eq. (4)
    replace  $p(\omega_n | Y^{(n)})$  with  $p^*(\omega_n | Y^{(n)})$ 
   $\Theta :=$  updated  $\Theta$  with EM
return  $\operatorname{argmax}_{\omega_N} p^*(\omega_N | Y^{(N)})$ 

```

Table 1. Truncated Baum-Welch procedure.

The second algorithm is a variation of the first method. During the forward pass, the first method removes some partitions because of their low likelihood, before improving the likelihood by fitting better parameters. In the second method we update Θ with EM using the partial data $Y^{(n)}$ every time before applying the approximation. To execute EM with the data $Y^{(n)}$, we have to replace $Y^{(N)}$ with $Y^{(n)}$ and ω_N with ω_n in eqs (5) and (6). The outline of the second method is shown in Tab.2. Here we will call it a Modified Truncated Baum-Welch. Another way to improve the method is to apply the approximation after the initial N_0 observations. For $n > N_0$ truncate the distribution $p(\omega_n | Y^{(n)})$ every $R > 1$ executions of eq. (4).

```

start with random  $\Theta$ 
iterate until convergence
  Iterate  $n = 1 : N_0$ 
    compute  $p(\omega_n | Y^{(n)})$  with eq. (4)
  while  $n < N$ 
    iterate for  $r = 1 : R$ 
      compute  $p(\omega_{n+r} | Y^{(n+r)})$  with eq. (4)
     $n := n + R$ 
   $\Theta :=$  updated  $\Theta$  with EM
  replace  $p(\omega_n | Y^{(n)})$  with  $p^*(\omega_n | Y^{(n)})$ 
return  $\operatorname{argmax}_{\omega_N} p^*(\omega_N | Y^{(N)})$ 

```

Table 2. Modified Truncated Baum-Welch procedure.

5 Experiments

In this section we show an experimental evaluation of the two proposed methods. We also perform an experiment with an MCMC-based approximate method for a comparison.

The input data Y for all experiments is a set of real object observations, that were obtained at 7 disjoint locations

at the ground and the first floor of the university building as in Fig. 2. In total we gathered 70 observations of 5 persons, with 14 observations per person. For this set we manually resolved the data association to have the 'ground truth' partition ω_T . In the building we set a distribution $p(l_i, e_i | l_{i-1}, d_{i-1})$ giving the probability of arriving at location l_i at the side e_i when an object departs from position l_{i-1} through side d_{i-1} . In the building we also set manually the probability $p(l_1, e_1)$ of entering the building at location l_1 from side e_1 .

The appearance description a_i was a 9D vector containing 3D color means (RGB) computed over 3 fixed regions in a blob image of an object. The regions (bottom, center, top; as in Fig.2) are a heuristic choice based on the assumption that we observe people. To suppress the effect of the color of illumination on object's observed color the original RGB representation was transformed to a normalized RGB space [9].

The methods are compared on the basis of how accurately they associate the observations. Every method returns a single partition ω , which is evaluated with two criteria. The primary evaluation q_ω of this partition is the average trajectory accuracy $q_\omega = \frac{1}{K_\omega} \sum_{k=1}^{K_\omega} q_{k,\omega}$, where K_ω is the number of trajectories defined by ω . The term $q_{k,\omega}$ is the association accuracy of the k th trajectory Y_k proposed by ω . Let the 'true' partition be $\omega_T = \cup_l Y_l^T$. The association accuracy $q_{k,\omega}$ is then:

$$q_{k,\omega} = \frac{\max_l |Y_l^T \cap Y_k|}{|Y_k|} \cdot 100\%, \quad (7)$$

where $|\cdot|$ is the number of observations in a set. The supplementary criterion is the number of identified distinct objects (trajectories) in the data. (Note that according to (7) proposing one trajectory per observation would always give 100% accuracy.)

5.1 Our approximations

We test the proposed methods on the above data. The tracking results for different settings of parameter K are shown in Tab. 3 as averages from 5 runs. An example run the second method is show in Fig. 2(bottom).

In the table we observe that the intermediate learning steps of the second method always allowed to identify the correct number of distinct persons from the data, while the first method tends to propose too many objects. The likelihood of partitions that group more observations into a single trajectory largely depends on the observation parameters. In the first method, the parameters are not learned before truncation, therefore such partitions may receive a zero likelihood after the approximation.

We also observe in the case of the second method (Truncated Modified B-W) that increasing the number K of preserved partitions improves the solution up to some value of K . When the EM is executed many times for small data sets $Y^{(n)}$, and there are preserved more partitions with

non-zero likelihood to choose from, then the method might overfit the parameters to a wrong partition.

K	Truncated B-W		Modified Truncated B-W	
	q_ω [%]	number of objects	q_ω [%]	number of objects
1	64 ± 4.8	8.8 ± 0.9	48 ± 3.6	5.0 ± 0.0
2	66 ± 6.0	8.4 ± 0.8	79 ± 0.0	5.0 ± 0.0
5	56 ± 8.8	7.4 ± 0.8	82 ± 3.5	5.0 ± 0.0
10	54 ± 1.4	7.2 ± 0.4	70 ± 1.9	5.0 ± 0.0
20	54 ± 1.4	7.0 ± 0.0	66 ± 0.8	5.0 ± 0.0

Table 3. Tracking with the Truncated Baum-Welch (B-W) and the Modified Truncated B-W, with $R = 2$, $N_0 = 6$. The averages and standard deviations from 5 runs are shown.

5.2 MCMC-based approximation

Another way to approximate the partition space is to view ω_N as a static variable, and sample possible partitions from the state space Ω^N as described in [3]. We implemented the MCMC sampling scheme to compare it with our approximations. The MCMC algorithm was run to build a set of a fixed number of samples from $\Omega^{(N)}$. For the comparison we take the partition with the highest likelihood among the sampled partitions found by MCMC.

The evaluation of the best (in terms of the posterior likelihood) partition is shown in Tab. 4. However increasing the number of samples improves the solution, it is still difficult to find a good single partition. The results confirm, that the MCMC is, in principle, designed to approximate some expected value over a distribution, rather than to find the most likely element of a distribution. We cannot build an expected partition because the partitions cannot be added. This limits the applicability of the MCMC-based approximation for the direct trajectory estimation.

number of samples	q_ω (best sample)	no of objects (best sample)
10^2	31 % ± 6.5	20 ± 4.4
10^3	30 % ± 6.6	8.8 ± 3.0
10^4	31 % ± 4.3	2.4 ± 1.0
10^5	38 % ± 5.3	2.4 ± 0.4

Table 4. Sampling partitions with an MCMC approximation. The results are averages and standard deviations from 5 runs.

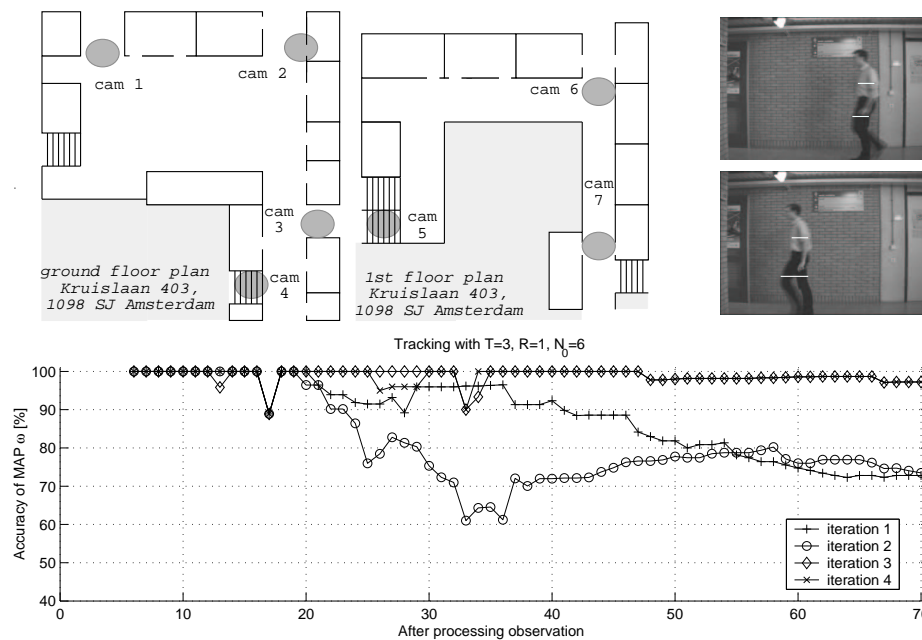


Figure 2. Top: A building plan where the observations were taken. The gray areas show camera viewing fields. Top-right: Camera frames with a passing person. The bars separate the regions for computing the average color. Bottom: An example run of the Modified Truncated Baum-Welch $N_0 = 6$, $R = 1$, $K = 3$. After every truncation we compute q_ω of the MAP partition.

6 Conclusions

This paper described a solution to the data association problem, which requires finding a partition of the set of observations into trajectories. We described a partition process as a hidden Markov Model that generates the observations. For this model we have shown an approximate inference and learning method, derived from the EM algorithm. The described method is another approach for the approximation of, in general, intractable space of all possible assignments of observations with objects. In the unconstrained traffic setup, the method was shown to be a useful alternative to the MCMC-based approximation. The experiments suggest that when the number of tracked objects is unknown, or difficult to infer, our method better recovers the number of distinct trajectories present in the data.

Acknowledgment This research is supported by the Technology Foundation STW (project no ANN5312) and the Dutch Ministry of Economic Affairs.

References

[1] T. Huang and S. Russell. Object identification: A Bayesian analysis with application to traffic surveillance. *Artificial Intelligence*, 103(103):1–17, 1998.

[2] K. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[3] H. Pasula, S. Russell, M. Ostland, and Y. Ritov. Tracking many objects with many sensors. In *Proc. of Int. Joint Conf. on Artificial Intelligence*, 1999. Stockholm.

[4] Radford M. Neal. Probabilistic inference using Markov Chain Monte Carlo methods. Technical Report CRG-TR-93-1, University of Toronto, 1993.

[5] Lawrence R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In A. Weibel and Kay-Fu Lee, editors, *Reading in Speech Recognition*, pages 267–296. Morgan Kaufmann, 1990.

[6] W. Zajdel and B. Kröse. Visual surveillance with non-overlapping cameras. UvA technical report, University of Amsterdam, 2003.

[7] Z. Ghahramani. Learning dynamic Bayesian networks. In C. L. Giles and M. Gori, editors, *Adaptive Processing of Temporal Information. Lecture Notes in Artificial Intelligence*. Springer-Verlag, 1997.

[8] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via EM algorithm. *Journal of the Royal Statistical Society Series B*, 39(1):1–38, 1977.

[9] M.S. Drew, J. Wei, and Z.N. Li. Illumination-invariant color object recognition via compressed chromaticity histograms of color-channel-normalized images. In *Proc. of Int. Conf. on Computer Vision*, pages 533–540, 1998.