

International Journal of Pattern Recognition and Artificial Intelligence
© World Scientific Publishing Company

A SEQUENTIAL BAYESIAN ALGORITHM FOR SURVEILLANCE WITH NON-OVERLAPPING CAMERAS

WOJCIECH ZAJDEL and BEN J. A. KRÖSE
Intelligent Autonomous Systems
Informatics Institute, University of Amsterdam
The Netherlands
wzajdel@science.uva.nl
kröse@science.uva.nl

CORRESPONDING AUTHOR:

Wojciech Zajdel
wzajdel@science.uva.nl
University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands,
tel: (+31 20) 525 7550(7490 fax)

ABBREVIATED TITLE: Bayesian algorithm for multicamera surveillance

Visual surveillance in wide areas (e.g. airports) relies on sparsely distributed cameras, that is, cameras that observe non-overlapping scenes. In this setup, multi-object tracking requires re-identification of an object when it leaves one field of view, and later appears at some other. Although similar association problems are common for multi-object tracking scenarios, in the distributed case one has to cope with asynchronous observations and cannot assume smooth motion of the objects. In this paper, we propose a method for human indoor tracking. The method is based on a Dynamic Bayes Network (DBN) as a probabilistic model for the observations. The edges of the network define the correspondences between observations of the same object. Accordingly, we derive an approximate EM-like method for selecting the most likely structure of DBN and learning model parameters. The presented algorithm is tested on a collection of real-world observations gathered by a system of cameras in an office building.

Keywords: Multiple-target tracking; Wide-area video surveillance; Data association; Bayes networks; Probabilistic learning.

1. Introduction

Multi-object tracking is a crucial task in computer vision systems for surveillance,^{5,4,3,29} traffic monitoring^{13,25,20} or intelligent homes³⁰. In all these cases, tracking is based on association of observations (or features derived from them) with objects of interest.

Traditionally, the tracking research has focused on relatively small scenes that can be observed fully with a single or a few cameras. The frame-to-frame tracking at a single camera relies on the continuity of objects' positions between frames. Smooth position changes allow to apply continuous-motion models, such as Kalman filters,^{2,6} for predicting positions in the new frame from the past frames. Appearance features make another association cue, where one employs probabilistic models to account for camera

noise.^{30,20,15} Systems with multiple cameras observing the same scene^{27,28,10}, encounter also the problem of associating objects simultaneously observed by all cameras.²¹ Here, different setups include reducing the overlap area to a minimum guaranteeing smooth camera-to-camera transitions⁴ or pan-tilt active cameras that select the best overlap section.²⁹ However, each of the configurations exploits overlap in the viewing field and smooth motion for data association.

Wide areas, such as airports or motorways, typically cannot be fully covered by a monitoring system. Therefore, surveillance of such places relies on a network of sparsely distributed cameras. Every camera observes a scene disjoint from the scenes observed by the other cameras. Tracking in this scenario requires camera-to-camera association of observations when an object leaves one scene and later appears at some other. There are several reasons why the single-camera association methods are not suitable for such a problem.¹³ Firstly, objects visit the disjoint scenes irregularly and asynchronously. Secondly, the motion between discontinuous scenes cannot be described with a continuous motion model. Finally, in the distributed case, one has to account for scene-specific illumination and camera viewing angle that affect the object's appearance.

Huang and Russell present a probabilistic approach for tracking of vehicles observed with two distant cameras on a motorway.¹³ Their method performs exhaustive search for the best association according to a fully observable model of vehicle's appearance at one camera given its appearance at the other camera. However, in case of systems with more than two cameras fully-observable models involve more than two observations and therefore are difficult to construct.²⁵ Often, one simplifies these models by assuming that the next observation of an object depends only on its previous observation (as in a Markov chain).^{16,19} This choice allows exhaustive search for the best association, however (fully-observable) Markov chains impose rather strong independence assumptions, and their performance has been shown very sensitive to the camera noise.²⁵

An alternative definition of a trajectory model introduces a hidden state that represents object's invariant appearance-related properties.²⁵ The presence of a hidden variable renders exact search for best association infeasible, therefore the hypothetical trajectories are sampled with the Markov chain Monte Carlo (MCMC) algorithm. This method is well suited to approximate some numerical traffic conditions (e.g. travel time) or to learn the environment parameters (e.g. expected travel time). However, it is not designed for exact recovering of individual trajectories.

Another class of methods follow from the general Multiple Hypothesis Tracking (MHT) scheme, e.g. a large surveillance project with distributed cameras.⁵ In this system, the cameras estimate objects' 3D geolocations, and classify the detected objects into categories like human, or car. Then, the joint position-class labels are associated by a MHT. However, this and most of the standard MHT applications assume predefined parameters rather than learned from the data.

Nicholson and Brady proposed an indirect solution to wide area surveillance by using movement detectors that split the monitored terrain into tiled regions.²⁴ Rather than finding correspondences between signals from detectors, they compute posterior proba-

bility distributions on the locations of the objects of interest. On the practical side, the method assumes that the starting positions and the number of objects are known.

In this paper we describe a probabilistic method for on-line association of observations and simultaneous learning of environment parameters (e.g. travel times). We assume a generative model for data in the form of a Dynamic Bayes Network with hidden states that represent appearance-related properties of objects. Our inference algorithm couples iterative reconstruction of trajectories with the Expectation Maximization approach for learning model parameters. In this way, we combine the learning ability of MCMC-based trackers and the ability to recover individual trajectories of the MHT approach.

2. Overview

We consider the problem of camera-to-camera tracking of an unknown number of persons observed with sparsely distributed cameras in an office-like environment. Although our goal resembles the motorway scenario of Ref. 25, there are two key differences. Firstly, we relax the assumption of unidirectional, constant-velocity vehicle movements. In a building people walk in all directions, possibly making long stops between camera locations. Secondly, we seek individual trajectories rather than averaged traffic parameters.

Since our method focuses on camera-to-camera trajectories we do not analyze the maneuvers of an object within a single field of view. We derive a “virtual” observation y_i from the sequence of frames that describe the complete pass of an object through the viewing field (as in Fig. 1). The observation $y_i = \{a_i, t_i, l_i, e_i, d_i\}$ includes: the time of observation t_i — represented as a single moment, the summarized object’s appearance (e.g. color statistics) a_i , the frame border of entering (e_i) and leaving (d_i) the field of view (this can be: right, left, top, down), and the camera location l_i where the object was observed. The term l_i is a discrete indicator of the area uniquely associated with the camera. We process observations from all cameras centrally, and treat i as a central index. We also assume that the observations are time ordered, *i.e.*, $t_i < t_j$ iff $i < j$.

Given the set of N observations $Y = \{y_i; i = 1, \dots, N\}$ we want to identify which observations belong to the same object. We are looking for a partition ω of observations from Y into several trajectories $Y_k \subset Y$ (subsets of Y) such that each trajectory collects observations believed to come from a single person. A valid partition expresses the set of all observations as an exhaustive union of trajectories: $Y = Y_1 \cup \dots \cup Y_{K_\omega}$, where K_ω is the number of objects proposed by a partition ω . The trajectories must be mutually exclusive: $Y_\ell \cap Y_k = \emptyset$, when $\ell \neq k$.

We consider a partition as a discrete-valued, random variable $\omega \in \Omega$, where Ω is the space of valid partitions. Conditioned on the data Y we find the posterior probability distribution of partitions, and select the most likely a-posteriori (MAP) ω . In Sec. 3 we define a generative model $p(Y|\omega)$, which becomes a basis for computing partition posterior distribution. The model includes (initially unknown) parameters that describe our environment, like the average travel times between camera locations. Accordingly, selecting the best partition has to be coupled with learning the parameters. Both tasks are difficult to solve exactly, due to an intractable number of possible partitions. Our

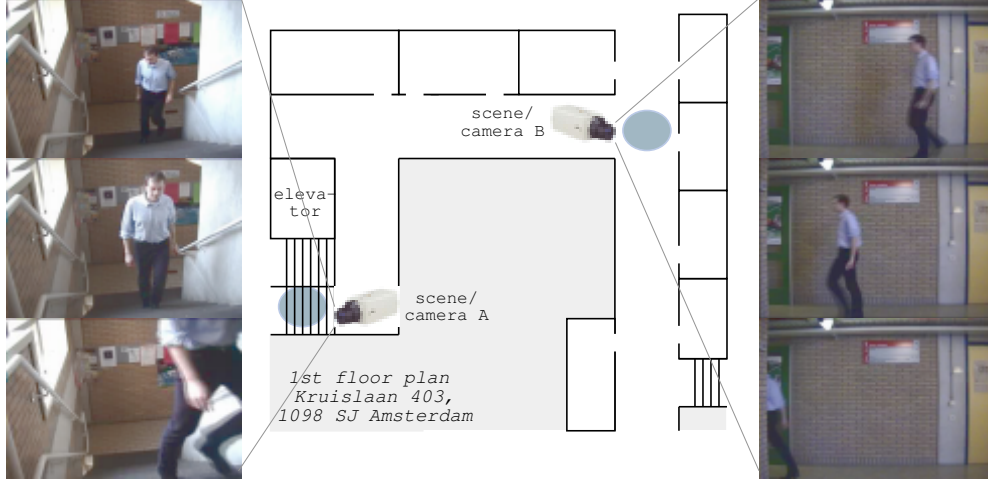


Fig. 1. Camera-to-camera association of observations. The left and right columns are examples of frames taken from two passes through a camera viewing field. Each pass makes a single observation.

approximate method (Sec. 4) is based on an EM⁷ algorithm. It starts with an initial guess on parameters and a tractable partition space obtained using a few initial observations. Each iteration refines the parameters and updates the limited partition space with new observations. The performance of the method on a real data set is evaluated in Sec. 5.

3. Generative model

A generative model $p(Y|\omega)$ provides the basic probabilistic relation that ties the available data Y and the sought partition ω . It gives the likelihood of data Y under the assumption that Y was generated according to the partition ω . Given the data and the model, the partition posterior probability follows from the Bayes' rule: $p(\omega|Y) = \alpha p(\omega)p(Y|\omega)$, where α is a scale factor. By assumption, all partitions are equally likely before the observations arrive, so we take an uniform prior $p(\omega)$.

Let ω define K_ω trajectories $Y = Y_1 \cup \dots \cup Y_{K_\omega}$. A common assumption is that the tracked objects move independently,^{25,6} therefore the generative model factorizes into a product of models for individual trajectories:

$$p(\omega|Y) = \alpha p(Y|\omega) = \alpha \prod_{k=1}^{K_\omega} p(Y_k|\omega), \quad (1)$$

where $p(Y_k|\omega)$ is the model for the k th trajectory. The trajectory model defines the joint probability of the selected observations under the assumption that they describe the same person $p(Y_k|\omega) = p(y_1^{(k)}, \dots, y_{N_k}^{(k)}|\omega)$, where the selection $Y_k = \{y_1^{(k)}, \dots, y_{N_k}^{(k)}\}$ is determined by the partition ω , and N_k is the trajectory length. Since trajectories differ in length, and each observation has several components, we will express the trajectory

model as a dynamic Bayes network. In the rest of this section we discuss a model for only one person and omit the superscript (k).

Bayes networks (BN) offer a convenient way to express complex probability density functions (pdf's) as a product of simpler conditional pdf's.^{26,17} A BN is a directed acyclic graph denoting variables as vertices and probabilistic dependencies as edges (Fig. 2). Dynamic Bayes networks (DBN) represent models for discrete-time series of variables with a series of interconnected subgraphs. DBN's generalize other probabilistic models that describe time series, like hidden Markov models or Kalman filter models.¹¹

The graph of Fig. 2 identifies a set of dependencies between the variables in the sequence $\{y_1, \dots, y_{N_k}\}$. Each column of white nodes (subgraph) represents a single observation y_i . The gray node denotes a hidden variable f that describes the time-invariant, intrinsic properties related to object's appearance. When an object enters a viewing field we get noisy observations of this variable. Thus, the appearance a_i given f is independent of all past and future appearances a_j of this person. In this way we have saved the effort of constructing an explicit joint model $p(a_{1:N_k})$. To complete the definition of our DBN, we provide a 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 a_i at a camera location l_i is a result of observing hidden properties f with noise specific to l_i . We will write $p(a_i|f, \theta(l_i))$, where $\theta(l_i)$ are parameters of the camera at location l_i (each camera is uniquely associated with a location). Our (heuristic) choice for this model is a linear Gaussian distribution: $\mathcal{N}(a_i; f + b(l_i), S(l_i))$. In the experiments with real-world data we verify the ability of such a simple distribution to capture the observation noise. The parameters $\theta(l_i) = \{b(l_i), S(l_i)\}$ will be learned from the data.
- $p(l_i, e_i|l_{i-1}, d_{i-1})$; Probability of arriving at location l_i via border e_i when departing from location l_{i-1} via border d_{i-1} . We specify this discrete distribution using prior knowledge of the building's layout. In contrast to Ref. 3 we do not define a separate motion model for each object. The distribution $p(l_i, e_i|l_{i-1}, d_{i-1})$ is a property of the environment, rather than an object.
- $p(t_i|l_{i-1}, l_i, t_{i-1})$; This pdf models the time of appearing at location l_i knowing that an object left l_{i-1} at time t_{i-1} . We use a truncated normal distribution:

$$p(t_i|l_{i-1}, l_i, t_{i-1}) = \begin{cases} 0 & \text{iff } t_i < t_{i-1} \\ \mathcal{N}(t_i; \delta(l_i, l_{i-1}) + t_{i-1}, R(l_i, l_{i-1})) & \text{otherwise} \end{cases}, \quad (2)$$

where $\delta(l_i, l_{i-1})$ and $R(l_i, l_{i-1})$ are the expected travel time between the two locations and the variance of this distribution. These parameters will be learned.

- $p(f)$; Prior distribution of the intrinsic properties of any object. We use a normal density: $p(f) = \mathcal{N}(f; \mu, R)$. The parameters μ and R will be learned.
- $p(l_1), p(e_1|l_1)$; Distributions on locations and entry sides where a trajectory may start in the building. This distributions are given by the user since they follow from the layout of cameras in the building.
- $p(d_i)$; Prior probability of observing a particular departure side. We take it the

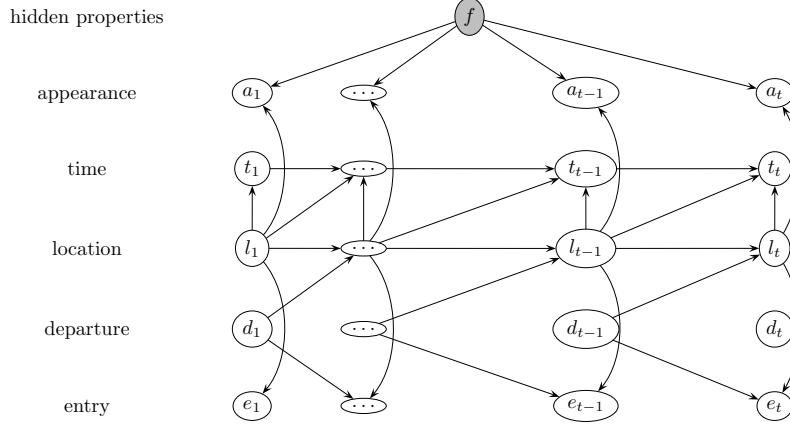


Fig. 2. An object as a hidden process. The gray node (variable f) describes an object's hidden, time-invariant appearance. The arcs show dependencies between the variables. When a new observation is associated, the network is extended with a new column of nodes.

same for every value of d_i , so the term $p(d_i)$ becomes a scaling factor.

- $p(t_1|l_1)$; Prior probability distribution on the time of object's first visit to some camera location. We assume that within the considered period a trajectory may start at any time. Thus, $p(t_1|l_1)$ is a scale term.

The joint probability of the variables in the graph is the product of pdf's associated with every vertex.¹⁷ In case of Fig. 2:

$$p(f, y_1, \dots, y_{N_k}) = p(l_1)p(e_1|l_1)p(t_1|l_1) \prod_{i=1}^{N_k} p(d_i) \prod_{i=2}^{N_k} [p(l_i, e_i|l_{i-1}, d_{i-1})p(t_i|t_{i-1}, l_i, l_{i-1})] p(f) \prod_{i=1}^{N_k} p(a_i|f, l_i).$$

The variable f is unknown, thus we integrate it in the final expression. The scaling terms are omitted:

$$p(Y_k|\omega, \Theta) = p(y_1, \dots, y_{N_k}|\omega) \propto p(l_1)p(e_1|l_1) \times \left\{ \prod_{i=2}^{N_k} p(l_i, e_i|l_{i-1}, d_{i-1})p(t_i|t_{i-1}, l_i, l_{i-1}) \right\} \int_f p(f) \prod_{i=1}^{N_k} p(a_i|f, l_i). \quad (3)$$

The product of Gaussian appearance models $p(a_i|f, l_i)$ is another Gaussian function, therefore the integral in Eq. 3 has an analytical solution. See Appendix A for a discussion on how to compute this term.

Conditioning on Θ in Eq. 3 follows from the dependency of a trajectory likelihood on the unknown parameters: $\Theta = \{\mu, R, \delta(q, r), R(q, r), b(q), S(q)\}$, where q and r are camera locations. Consequently, this dependency reappears in the posterior probability (Eq. 1), and we have to learn Θ before or during selection of the best ω .

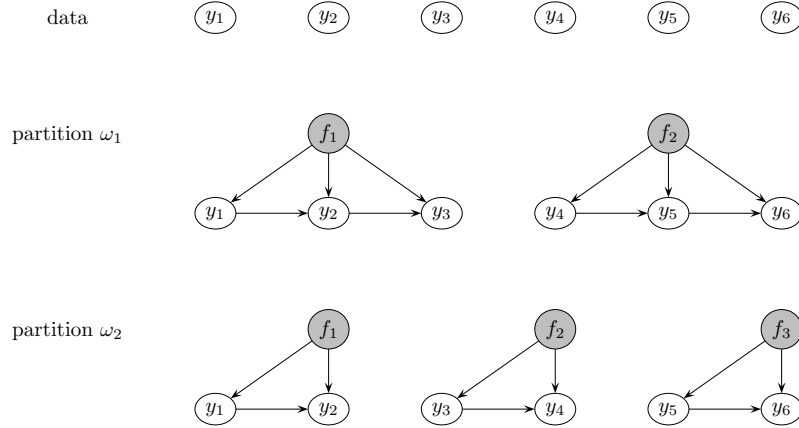


Fig. 3. Selection of a Bayes network structure as a data association problem. The six observations could be generated from many different models (structures). Two possible structures ω_1 and ω_2 are shown. A single network (e.g. $\{f_1, y_1, y_2\}$) is a compressed version of the corresponding full network as in Fig. 2.

4. Associating observations

In the previous section we described a trajectory of a single person with a dynamic Bayes network. Accordingly, to describe multiple persons, we construct several disconnected DBNs as in Fig. 3. In this way, every partition ω translates to a structure of a larger network that comprises multiple disconnected subgraphs, each for every proposed object. In this section we present a deterministic approximate method for estimating the sought partition by recovering the structure that explains the data best.

Typical structure selection procedures for BNs compare candidate structures on the basis of a criterion (such as BIC/MDL) that tries to balance the complexity of a structure against its fit to the data (data likelihood).²³ Such criteria prevent favoring very complex structures that naturally fit the data better. In our case every candidate ω uses the same parameters Θ , but may propose a different number of hidden variables f . These variables increase the expressive power of the candidate model. However, we treat f in a Bayesian manner, *i.e.*, we set a prior on every f and integrate it from $p(\omega|Y, \Theta)$. Such an approach is shown in Ref. 22 to sufficiently penalize over-complex models, therefore the posterior $p(\omega|Y, \Theta)$ is a valid criterion.

Another issue is the exponential number of possible structures (*i.e.*, partitions) that can be defined for a dataset,¹⁸ and the fact that the parameters Θ need to be learned. Most of the approximate methods for these problems apply the EM algorithm.²³ For a tractable structure space, one could apply the “Structural EM” method, that assumes that the structure is another parameter and exhaustively searches for the optimal value.⁹ When the structure space is intractable, EM considers the structure as a hidden random variable. However, EM relies on expected values over the hidden variables. These expectations become difficult to compute for large structure spaces. In such cases the

expected values are approximated with Markov chain Monte Carlo (MCMC) sampling methods.

In practice, applications using MCMC may suffer from slow mixing times, e.g. Ref. 12, where 10^5 MCMC iterations are used to find a distribution over the space of $3 \cdot 10^6$ structures with only 6 nodes. The success of MCMC for motorway surveillance partially follows from constraining the partition space by a natural assumption that on a motorway vehicles travel in one direction (thus, the next locations of the objects are roughly known).²⁵ Apart from the fact that our domain is less constrained, we not only have to find expected values over this space, but also need the most likely partition. MCMC is useful only for approximating the expected value. Therefore, we are looking for a different approximate execution of the EM, such that it would also estimate the most likely structure in the space of all possible structures.

4.1. EM for a tractable structure space

Below we discuss a theoretical application of EM for learning of parameters Θ , based on the assumption that the structure space is tractable. The parameter estimates will be only locally optimal due to the approximate nature of EM. From this method, we will later derive an approximate learn-search algorithm for intractable spaces.

EM takes an initial value of Θ at random and improves it iteratively executing two steps per iteration. In the E step it finds the posterior pdf on all hidden variables given the current parameter estimates $\Theta^{(n)}$. In our application there is a hidden discrete-valued variable ω and as many hidden continuous-domain variables f as objects proposed by a particular ω . We denote the variables f_1, \dots, f_{K_ω} as $f_{1:K_\omega}$. The interesting posterior is:

$$q(\omega, f_{1:K_\omega}) = p(\omega|Y, \Theta^{(n)}) \prod_{k=1}^{K_\omega} p(f_k|Y_k, \omega, \Theta^{(n)}), \quad (4)$$

where $p(f_k|Y_k, \omega, \Theta^{(n)})$ is the posterior distribution on the hidden properties of the k th object proposed by the partition ω . The above factorization followed from conditional independence of trajectories given ω . In the M step, EM finds improved parameters $\Theta^{(n+1)}$ by maximizing the expected log-likelihood of the observations under $q(\omega, f_{1:K_\omega})$

$$\Theta^{(n+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega \in \Omega} \int_{f_{1:K_\omega}} \ln p(\omega, f_{1:K_\omega}, Y|\Theta) q(\omega, f_{1:K_\omega}) \quad (5)$$

Appendix A provides the details of solving the above maximization problem.

The above procedure is guaranteed to find Θ that locally maximize the data likelihood. Given these parameters we can use $p(\omega|Y, \Theta)$ to find the MAP partition.

4.2. An approximate iterative solution

In practice, the space of possible partitions Ω is intractable, and one cannot maximize Eq. 1 or execute the summation in Eq. 5 for every possible partition ω . Our

method exploits the time order of observations to approximate the posterior distribution $p(\omega|Y)$, $\omega \in \Omega$ over the full space Ω with a tractable subspace Ω_T of the T most likely partitions:

$$p^*(\omega|Y) = \begin{cases} \gamma p(\omega|Y) & \text{iff } \omega \in \Omega_T \\ 0 & \text{otherwise} \end{cases}, \quad (6)$$

where γ is a normalization constant. We construct the subspace Ω_T iteratively as a part of the EM procedure.

Figure 4 presents an iterative scheme to process the observations sequentially. First we process an initial subset $Y^{(0)}$ of N_0 observations, and construct a tractable partition space $\Omega^{(0)}$. Since the parameters Θ were randomly initialized we need to execute several EM iterations. When EM stops we have posterior distribution on partitions of observations from $Y^{(0)}$. At this point we build a subset $\Omega_T^{(0)}$ by selecting T partitions $\omega \in \Omega^{(0)}$ that are the most likely and remove the remaining partitions from $\Omega^{(0)}$. The subsequent observations will be used for refining the parameters and for updating the retained partitions. Each of the preserved partitions $\omega \in \Omega_T^{(0)}$ defines some K_ω trajectories. By extending every trajectory with a single observation or creating new trajectory, a single partition gives rise to $(K_\omega + 1)$ new partitions. Accordingly, after extending every preserved ω there is a new space of partitions $\Omega^{(1)}$. The new space is tractable, since we created it from a tractable $\Omega_T^{(0)}$. It is further extended with the subsequent observations until we process a batch of R observations. (R is a parameter.) The new tractable space $\Omega^{(1)}$ becomes a basis for updating the parameters with EM and searching for a new subset of the most likely partitions.

The described procedure combines the iterative EM algorithm with the sequential search for the best partitions. By updating Θ after processing each batch of R observations we enforce that Θ follows changes in the noise environment. However, our scheme is not exact, because we discard unlikely partitions from Ω without considering all observations from Y . One should also note, that we estimate maximum-likelihood parameters from an incomplete data sequence (partial dataset). In order to avoid overfitting, it is recommended to run only a single EM iteration after each batch of R data.

4.3. Relation to other methods

The key problem faced by any data association method stems from the intractability of the partition space. An alternative solution to this problem applies MCMC sampling algorithms (mostly used Metropolis-Hastings)²⁵ to obtain a limited, sample-based representation of the partition space. The strength of MCMC methods follows from the fact the one can easily sample from the posterior $p(\omega|Y)$ that incorporates the *complete* dataset. However, when the partition space is unconstrained, the slow mixing time becomes a disadvantage. Our algorithm evaluates trajectories using partial datasets. In this way we keep the partition space tractable, but reject certain solutions on the basis of partial data.

Our method resembles a Multiple Hypothesis Tracker (MHT), because one may interpret a partition ω as a hypothesis about underlying trajectories. MHT evaluates

```

 $Y^{(0)} := y_{1:N_0}$ 
build  $\Omega^{(0)}$  space of partitions of obs. in  $Y^{(0)}$ 
use EM to find parameters  $\Theta^{(0)}$ ,
build  $\Omega_T^{(0)}$  by taking  $T$   $\omega \in \Omega^{(0)}$  with the highest  $p(\omega|y_{1:N_0}, \Theta^{(0)})$ 

 $n = N_0 + 1$ ;  $s = 1$ 
while  $n < N$ 
     $Y^{(s)} := y_{n:n+R}$ 
    build  $\Omega^{(s)}$  by extending every trajectory of  $\omega \in \Omega_T^{(s-1)}$  with  $y \in Y^{(s)}$ 
     $\Theta^{(s)} :=$  updated  $\Theta^{(s-1)}$  with EM (single iteration)
    build  $\Omega_T^{(s)}$  by taking  $T$   $\omega \in \Omega^{(s)}$  with the highest  $p(\omega|y_{1:n+R}, \Theta^{(s)})$ 
     $n := n + R$ ;  $s := s + 1$ 

```

Fig. 4. Pseudo code for the learn-search procedure. Symbol $y_{n:m}$ denotes a set $\{y_n, \dots, y_m\}$.

$p(\{y^*, Y_k\}|\omega)$ in order to decide which of the trajectories Y_k proposed by some ω should be extended with a new observation y^* . We first consider all options by building a full new partition space derived from the current ω , learn the parameters and then prune unlikely partitions. In most of the standard MHT applications parameters are predefined rather than learned.

Similarly as MHT methods, the procedure of Fig. 4 might also be considered a simple form of a (Rao-Blackwellized) particle filter (PF).¹ In visual tracking PFs are used for approximating intractable filtering densities, typically, on object's position.¹⁴ PF represents the density of interest with a set of samples (particles), which are resampled when new observations become available. In our case the hidden variable is the partition ω with a discrete, finite state space. We represent the posterior distribution with a few most likely elements (particles), because the whole state space is too large for exact representation. These particles are not however a result of resampling, but greedy search. In most of the PF applications for visual tracking the observation model is assumed to be known. In our method we assume a parametric observation model and we learn the model parameters.

5. Experiments

In a series of experiments we measure the performance of our method. First, we study the performance as a function of the observation model. In Sec. 3 this model was set to a linear Gaussian distribution with bias: $p_B(a_i|f, l_i) = \mathcal{N}(a; f + b(l_i), R(l_i))$. The camera dependent bias $b(l_i)$ accounts for local illumination and pose of objects relative to the camera. Since the choice of this model is arbitrary we also try its simpler version, without the bias $p_U(a_i|f, l_i) = \mathcal{N}(a; f, R(l_i))$. The next experiment studies the effect of the parameter T which determines the size of the approximate partition space Ω_T . Finally, we compare our method with the best sampled solution found by an MCMC-

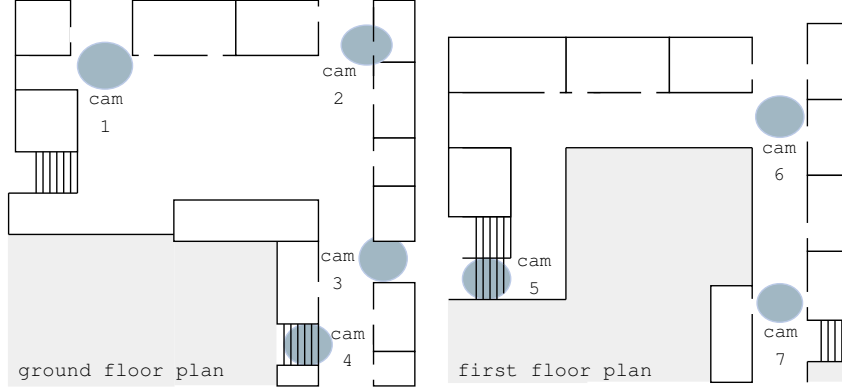


Fig. 5. Building plan where the observations were taken. The seven gray ovals indicate viewing fields of the seven cameras.

based algorithm.

5.1. Setup

We test our method on real-world human observations that were collected at seven disjoint locations at the ground and first floor of an office building as in Fig. 5. In total we gathered 70 observations of 5 persons, with an equal number of observations per person. For this set we manually resolved the data association to have the “ground truth” partition ω' . Because the layout of the building is known, we can set the probability $p(l_i, e_i | l_{i-1}, d_{i-1})$ 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} . This distribution is sketched in the Fig. 6a. In the building, we also manually set the models $p(l_1)$ and $p(e_1 | l_1)$ that describe starting location and entry side of a trajectory.

The appearance features a_i were set to a 9D vector containing 3D color means (RGB) computed over three regions selected in the object’s image. The regions (Fig. 6b) are a heuristic choice based on the assumption that we observe people. These features provide a simple way to summarize color while preserving partial information about geometrical layout. The means are approximately pose and shape invariant. To suppress the effect of the illumination color on the object’s observed color, we transformed the original RGB representation to a channel-normalized space as proposed in Ref. 8.

5.2. Evaluation

The methods are compared on the basis of how accurately they associate the observations. It is desirable that: (1) the observations within a trajectory correspond to the same person, and (2) that the recovered partition defines the correct number of trajectories. Therefore, a partition ω is evaluated with two criteria against the “ground truth”.

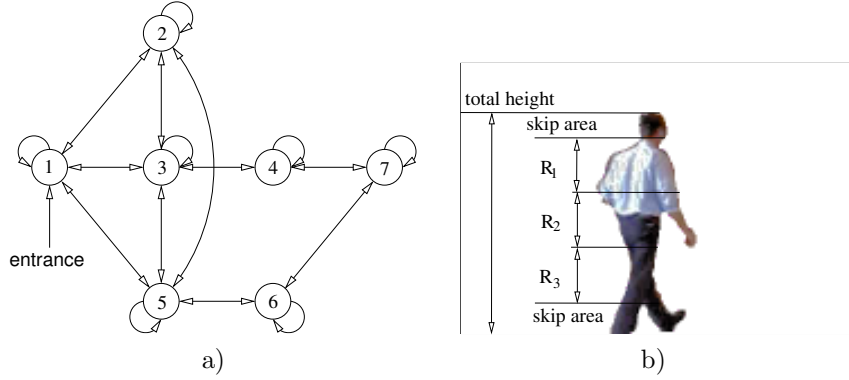


Fig. 6. a) An illustration of possible camera-to-camera movements in the considered environment. The numerical labels correspond to cameras from Fig. 5, the arrows indicate possible transitions as defined by distribution $p(l_i, e_i | l_{i-1}, d_{i-1})$. In the drawing we omitted the entry/departure side indication. b) Appearance description. Three regions: R_1, R_2, R_3 ; each of 25% of the total height. Within every region, we compute a 3D (RGB) average pixel color. The skip areas are meant to remove parts that have little discriminative power. These are fixed to 12.5% of the total height.

The primary criterion q_ω is the *partition accuracy*:

$$q_\omega = \frac{1}{K_\omega} \sum_{k=1}^{K_\omega} q_{k,\omega}, \quad (7)$$

where K_ω is the number of trajectories defined by ω . The term $q_{k,\omega}$ tells how many observations of a *proposed* subset Y_k are present in a single subset Y_i' of the *true* partition. Let the “true” partition be $\omega' = Y_1' \cup \dots \cup Y_N'$. Since the labeling of trajectories is not known, we have to check the k th recovered trajectory against all true Y_i' . The $q_{k,\omega}$ is then the accuracy of the best match:

$$q_{k,\omega} = \frac{\max_i |Y_i' \cap Y_k|}{|Y_k|} \cdot 100\%, \quad (8)$$

where $|\cdot|$ is the number of observations in a set. The secondary criterion is the *number of identified objects* (distinct trajectories). Note that according to Eq. 8 proposing one trajectory $Y_k = \{y_k\}$ per observation would always give a 100% accuracy.

5.3. Results

A sample run of our method with the general linear Gaussian model $p_B(a_i | f, l_i) = \mathcal{N}(a; f + b(l_i), R(l_i))$ is shown in Fig. 7a. The parameters were chosen as $T = 10, N_0 = 6, R = 2$. Each dot represents a partition from the estimated subspace Ω_T . The line in the figure connects the partitions that maximize the posterior probability. After an initial period of correct associations, the performance levels at around 60%, meaning that only 6 out of 10 observations assigned to a single trajectory really describe the same person.

Table 1. Tracking accuracy of the described method, averaged from 10 runs. Parameters $N_0 = 6$, $R = 2$. The actual number of objects was 5. Computation times correspond to a MATLAB implementation and an 800 MHz desktop PC.

parameter T	unbiased noise model			biased noise model		
	q_ω	objects	time [s]	q_ω	objects	time [s]
1	48% \pm 3.6	5 \pm 0	54 \pm 1	44% \pm 1	5 \pm 0	102 \pm 4
5	77% \pm 8.6	5 \pm 0	258 \pm 1	57% \pm 2.7	5 \pm 0	471 \pm 1
10	78% \pm 6.4	5 \pm 0	515 \pm 1	54% \pm 2.3	5 \pm 0	943 \pm 5
20	76% \pm 9.0	5 \pm 0	1036 \pm 3	46% \pm 2.1	5 \pm 0	1892 \pm 5

The average accuracy of MAP ω indicated in the figure is the average of q_ω for MAP partitions after processing each batch of R observations.

Fig. 7b presents the corresponding run of our method with a simple Gaussian noise model $p_U(a_i|f, l_i) = \mathcal{N}(a; f, R(l_i))$. All parameters were kept the same as in the previous run. In the case of the simpler model the method provided nearly perfect assignments for approximately the first 36 observations, whereas in the case of the general model – for only 16. For the subsequent observations (starting at around observation 54) we observe that the MAP partition coincides with the highest-accuracy partition. Fig. 8 presents selected trajectories as defined by the estimated MAP partition after processing 70 observations.

The evaluation of our method with different noise models and varying setting of parameter T is given in Tab. 1. The results are averages from 10 runs. We observe that keeping only a single ($T = 1$) partition was not enough to obtain an accurate solution. When the number of preserved ω 's was slightly higher ($T = 5, 10$) the performance improved. However increasing T further did not improve the results, or even the performance deteriorated in case of the general observation model.

There are two main reasons why the simpler noise model performs better on the considered dataset. First, the EM method for parameter estimation is prone to finding only the locally optimal estimates. The more parameters assumed by the model, the more local maxima exist in the data likelihood function. In our application this could be especially the case since we always run EM on a partial dataset. The bias estimates found from the initial observations may not fit well to the subsequent data. The second reason follows from our image pre-processing step. The camera-specific bias parameter might be unnecessary because we suppress camera-dependent noise artifacts by using pose-invariant features and a channel-normalized color space.

Finally, we note that the appearance features are computed from multiple frames, where an object was visible during his/her pass through the field of view. For individual frames, the appearance features might be strongly affected by various noise artifacts (e.g. resulting from bad segmentation or occlusion). Nevertheless, we given a complete pass, such artifacts can be easily eliminated as “outliers”.

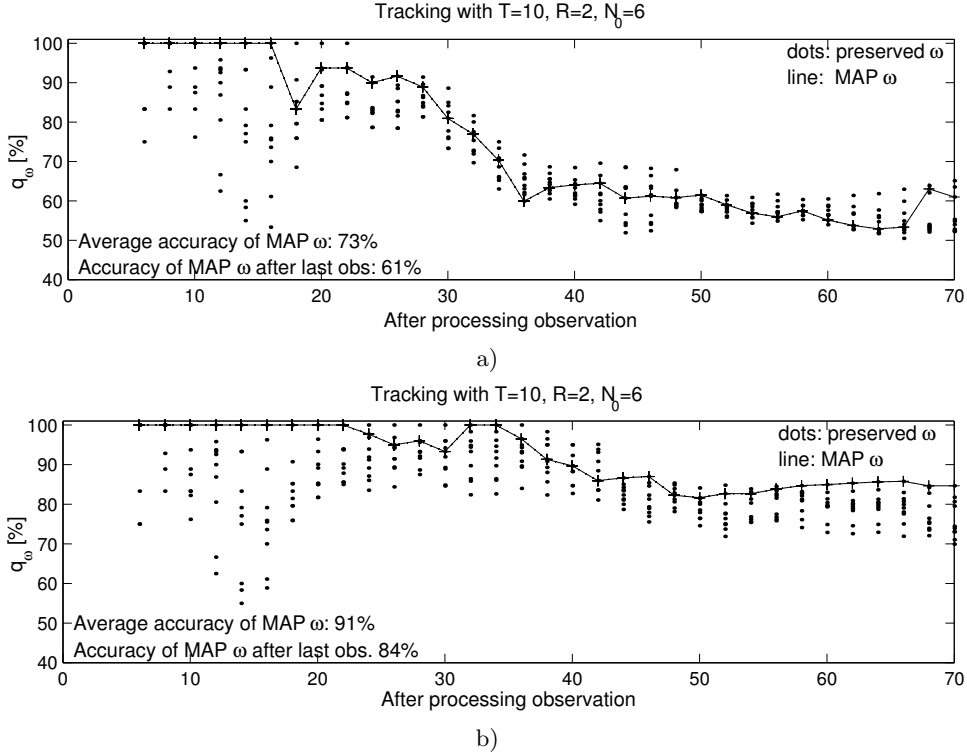


Fig. 7. Results of tracking with normal prior distribution on features f . The parameters of the distribution were randomly initialized and learned from data. a) Observation model is linear Gaussian. b) Observation model is a simple Gaussian (with linear bias parameter set to zero).

5.4. MCMC-based algorithm

Another series of experiments evaluate the MCMC-based approximations to the partition space. We implemented the Metropolis-Hastings sampling algorithm of Ref. 25. In essence, the partitions $\omega_i, i = 1 \dots N$ are sampled as follows: a new ω_{i+1} is obtained from ω_i by random selection of two observations assigned to different trajectories and swapping their assignment. Then one accepts ω_{i+1} at random with probability proportional to $p(\omega_{i+1}|Y)/p(\omega_i|Y)$. In the motorway scenario²⁵, construction of the initial sample exploits the natural motorway constraint that objects (vehicles) move in a single direction. In our office scenario, the objects (people) move without constraints. Therefore, we consider two different ways to initialize sampling. First, we build the initial sample in a purely random way. Second, we obtain the initial sample as a result of a greedy search, using a simple MHT tracker with only one hypothesis.

We set the observation model to be the Gaussian $p_U(a_i|f, l_i) = \mathcal{N}(a; f, R(l_i))$, and prior models the same as in our algorithm. Since we cannot average over the sampled partitions, as the final solution we took the sample that maximized the data likelihood

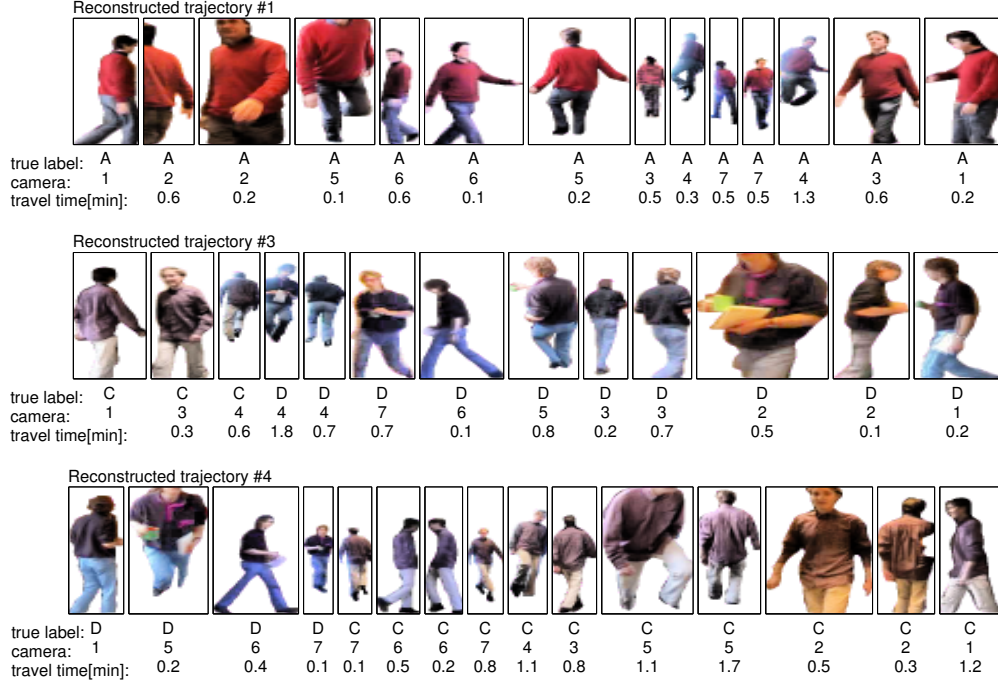


Fig. 8. Three selected trajectories defined by a partition ω estimated by our algorithm. Every image represents a frame from a person’s pass through the field of view of some camera. The “true labels” indicate the “ground truth” association. The numerical camera indicators correspond to Fig. 5. From the “true labels”, we see that the trajectory #1 has been reconstructed correctly, the trajectories #3 and #4 are mixed. The trajectories #2 and #5 (not shown) have been reconstructed correctly.

$p(Y|\Theta, \omega_i)$. Table 2 shows the evaluation results of the partitions found by MCMC runs with varying number of samples N and different methods for sample initialization.

In comparison with the Tab. 1 it turns out that the MCMC approximation is less suited for finding a single reliable partition ω in an office-like tracking scenario. In particular, it is difficult to estimate the correct number of objects. The original application of MCMC for tracking largely exploited the specific motorway traffic properties for disambiguating the number of distinct objects.²⁵ In that case a vehicle enters a motorway in one of the known locations, exits it only in specific off-ramps, and never (implicit assumption) visits the same location twice. We also note that the MCMC runs tend to be computationally more intensive, and their results highly depend on the initial sample.

6. Conclusions

We proposed a data association algorithm for tracking people observed with sparsely distributed cameras in an office environment. In particular, we have focused on re-identification of a person when he/she leaves the viewing field of one camera and later appears at some other camera. The algorithm assumes a probabilistic relation between

Table 2. Tracking accuracy of the MCMC based-algorithms. The results are averages over 5 runs. The actual number of objects was 5.

samples	random init			MHT init		
	q_ω	objects	time [s]	q_ω	objects	time [s]
10^2	57% \pm 9.8	18.8 \pm 8.0	89 \pm 4	62% \pm 7.7	7.4 \pm 1.0	97 \pm 0
10^3	66% \pm 5.3	13.8 \pm 2.1	789 \pm 16	73% \pm 4.9	7.2 \pm 0.4	759 \pm 3
10^4	78% \pm 2.1	9.6 \pm 0.8	7689 \pm 53	87% \pm 1.4	6.6 \pm 0.5	7401 \pm 19
10^5	86% \pm 3.4	7.0 \pm 0.7	76631 \pm 50	88% \pm 1.2	6.2 \pm 0.5	73992 \pm 17

the available object observations and the sought association. Given the data, our approximate inference procedure iteratively reconstructs a limited space of a-posteriori plausible associations. The proposed scheme is derived from the EM algorithm in order to combine learning of noise model parameters with the search through the association space.

The described EM-based algorithm for learning of model parameters resembles the recent approaches used by the MCMC-based trackers. However, in contrast to the methods employing MCMC sampling, our method is able to recover a single reliable partition (association) of the observation sequence into individual trajectories. Our method could be also viewed as an extension to Multiple Hypothesis Tracking with on-line learning of model parameters. Therefore, it is particularly suited for on-line applications.

The presented greedy search for the optimal association offers an alternative to the methods employing stochastic approximations of the association space. This space is particularly difficult to approximate in the indoor tracking domain, which lacks strong traffic constraints. The presented experiments suggest that for this domain, sequential search for trajectories from an incomplete dataset provides better results than sampling associations from the complete dataset.

Appendix A. EM for Bayes network parameters

Below we provide implementation details for finding maximum-likelihood model parameters Θ with EM. In the E step we find posterior distribution on all hidden variables:

$$\begin{aligned}
 q(\omega, f_{1:K_\omega}) &= p(\omega, f_{1:K_\omega} | Y, \Theta^{(n)}) = \\
 &= p(\omega | Y, \Theta^{(n)}) p(f_{1:K_\omega} | Y, \omega, \Theta^{(n)}) = p(\omega | Y, \Theta^{(n)}) \prod_{k=1}^{K_\omega} p(f_k | Y_k, \omega, \Theta^{(n)}), \quad (\text{A.1})
 \end{aligned}$$

where $p(f_k | Y_k, \omega, \Theta^{(n)})$ is the posterior distribution on the hidden properties of the k th object proposed by a partition ω . The factorization in Eq. A.1 followed from trajectory conditional independence given ω (the subgraphs of Fig. 3 are disconnected). The term $q(\omega) = p(\omega | Y, \Theta^{(n)})$ is given by Eq. 1. To find the function $q(f_k | \omega) = p(f_k | Y_k, \omega, \Theta^{(n)})$ we exploit the independences implied by the trajectory model, as in Fig. 2

$$q(f_k | \omega) = p(f_k | Y_k, \omega, \Theta^{(n)}) \propto p(f_k) \prod_{i=1}^{N_k} p(a_i^{(k)} | f_k, l_i^{(k)}), \quad (\text{A.2})$$

where N_k is the number of observations assigned to the k th object, $a_{1:N_k}^{(k)}$ and $l_{1:N_k}^{(k)}$ are sequences of appearance features and locations of the k th object as proposed by ω . Since all models in Eq. A.2 are linear and Gaussian, the posterior pdf will be a Normal density, what makes Eq. A.2 a special case of the Kalman filter.

In the M step we solve the following maximization problem:

$$\Theta^{(n+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega \in \Omega} \int_{f_{1:K_\omega}} \ln p(\omega, f_{1:K_\omega}, Y|\Theta) q(\omega, f_{1:K_\omega}). \quad (\text{A.3})$$

We express the joint $p(\omega, f_{1:K_\omega}, Y|\Theta)$ as a product of $p(f_{1:K_\omega}, Y|\omega, \Theta)$ and $p(\omega|\Theta)$. The latter is fixed uniform, so it is omitted. We solve

$$\Theta^{(n+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega \in \Omega} \int_{f_{1:K_\omega}} \ln p(f_{1:K_\omega}, Y|\Theta, \omega) q(\omega, f_{1:K_\omega}). \quad (\text{A.4})$$

The joint pdf $p(f_{1:K_\omega}, Y|\Theta, \omega)$ factorizes into a product of trajectory models (see Fig. 3). As a result the maximized term becomes:

$$\Theta^{(n+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega \in \Omega} \int_{f_{1:K_\omega}} \sum_{k=1}^{K_\omega} \ln p(f_k, Y_k|\Theta, \omega) q(\omega, f_{1:K_\omega}). \quad (\text{A.5})$$

Solving the integral for every component of the sum yields:

$$\Theta^{(n+1)} = \operatorname{argmax}_{\Theta} \sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \int_{f_k} \ln p(f_k, Y_k|\Theta, \omega) q(\omega, f_k). \quad (\text{A.6})$$

The above expression is our starting point for finding $\Theta^{(n+1)}$.

As an example we show how to update the prior distribution of object hidden properties. The distribution was assumed normal $p(f) = \mathcal{N}(f; \mu, R)$. Thus we find parameters μ and R . Since $\ln p(f_k, Y_k|\Theta, \omega) = \ln p(f_k) + \ln p(Y_k|f_k\Theta, \omega)$ and $p(Y_k|f_k\Theta, \omega)$ does not depend on the sought μ and R ; we maximize:

$$\sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \int_{f_k} \ln p(f_k) q(\omega, f_k) = \sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \int_{f_k} \ln \mathcal{N}(f_k; \mu, R) q(\omega, f_k). \quad (\text{A.7})$$

In order to maximize Eq. A.7 w.r.t μ we take the derivative w.r.t μ , set it to zero and solve for μ :

$$\mu^{(n+1)} = \frac{\sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \int_{f_k} f_k q(\omega, f_k)}{\sum_{\omega \in \Omega} K_\omega q(\omega)} = \frac{\sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \mu_k q(\omega)}{\sum_{\omega \in \Omega} K_\omega q(\omega)}, \quad (\text{A.8})$$

where the joint pdf $q(\omega, f_k)$ was expressed as a product: $q(f_k|\omega)q(\omega)$ (see the E step). The posterior $q(f_k|\omega)$ is a Normal pdf, with mean μ_k and covariance R_k . The integral $\int_{f_k} f_k q(f_k|\omega) = \mu_k$ is the expected value of this distribution (we have it from the E step).

Similarly one can find the R that maximizes Eq. A.7:

$$R^{(n+1)} = \frac{\sum_{\omega \in \Omega} \sum_{k=1}^{K_\omega} \left(R_k + \mu_k \mu_k' - \mu_k \mu^{(n)'} - \mu^{(n)} \mu_k' + \mu^{(n)} \mu^{(n)'} \right) q(\omega)}{\sum_{\omega \in \Omega} K_\omega q(\omega)}, \quad (\text{A.9})$$

where $\mu^{(n)}$ is the past estimate of parameter μ , R_k is the posterior covariance of $q(f_k|\omega)$ and μ' indicates transposition. Note, that the summation over ω is tractable because our approximation replaces the full partition space Ω with a tractable subspace Ω_T as in Eq. 6. In a similar way we find the observation model parameters (per camera) and travel time parameters (per camera pair).

References

1. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing* **50**, 2 (2002) 174–188.
2. Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.
3. H. H. Bui, S. Venkatesh, and G. West. Tracking and surveillance in wide-area spatial environments using the abstract hidden Markov model. *International Journal of Pattern Recognition and Artificial Intelligence* **15**, 1 (2001) 177–197.
4. Q. Cai and J. K. Aggarwal. Tracking human motion in structured environments using a distributed-camera system. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **21**, 11 (1999) 1241–1247.
5. R. Collins, A. Lipton, H. Fujiyoshi, and T. Kanade. Algorithms for cooperative multisensor surveillance. *Proceedings of the IEEE* **89**, 10 (2001) 1456–1477.
6. I. J. Cox. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision* **10**, 1 (1993) 53–66.
7. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via EM algorithm. *Journal of the Royal Statistical Society B* **39**, 1 (1977) 1–38.
8. M.S. Drew, J. Wei, and Z.N. Li. Illumination-invariant color object recognition via compressed chromaticity histograms of color-channel-normalized images. In *IEEE Int. Conf. on Computer Vision*, pp. 533–540, 1998.
9. N. Friedman. The Bayesian structural EM algorithm. In *Uncertainty in Artificial Intelligence*, 1998.
10. D.M. Gavrila and L.S. Davis. 3-d model-based tracking of humans in action: A multi-view approach. In *IEEE Computer Vision and Pattern Recognition*, p. 73–80, 1996.
11. Z. Ghahramani. An introduction to hidden Markov models and Bayesian networks. *International Journal of Pattern Recognition and Artificial Intelligence* **15**, 1 (2001) 9–42.
12. P. Giudici and R. Castelo. Improving Markov chain Monte Carlo model search for data mining. *Machine Learning* **50**, (2001) 127–158.
13. T. Huang and S. Russell. Object identification: A Bayesian analysis with application to traffic surveillance. *Artificial Intelligence* **103**, 1–2 (1998) 1–17.
14. M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision* **29**, 1 (1998) 5–28.
15. D. S. Jang, G. Y. Kim, and H. I. Choi. Model-based tracking of moving object. *Pattern Recognition* **30**, 6 (1997) 999–1008.
16. O. Javed, Z. Rasheed, K. Shafique, and M. Shah. Tracking across multiple cameras with disjoint views. In *IEEE Int. Conf. on Computer Vision*, pp. 952–957, 2003.
17. F. V. Jensen. *Bayesian Networks and Decision Graphs*. Springer-Verlag, New York, 2001.
18. M. Jerrum and A. Sinclair. The Markov chain Monte Carlo method: an approach to approximate counting and integration. In D.S. Hochbaum, ed., *Approximation Algorithms for NP-hard Problems*. PWS Publishing, 1996.
19. V. Kettner and R. Zabih. Bayesian multi-camera surveillance. In *IEEE Computer Vision and Pattern Recognition*, pp. 2253–2261, 1999.

20. D. Koller, J. W. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *European Conf. on Computer Vision*, pp. 186–196, 1994.
21. Y. Li, A. Hilton, and J. Illingworth. A relaxation algorithm for real-time multiple view 3d-tracking. *Image and Vision Computing* **20**, 12 (2002) 841–859.
22. D. J. MacKay. Bayesian Interpolation. *Neural Computation* **4**, 3 (1992) 415–447.
23. K. Murphy. Learning Bayes net structure from sparse data sets. Technical report, University of California, Berkeley, 2001.
24. A. E. Nicholson and J.M. Brady. Dynamic belief networks for discrete monitoring. *IEEE Transactions on Systems, Man, and Cybernetics* **24**, 11 (1994) 1593–1610.
25. H. Pasula, S. Russell, M. Ostland, and Y. Ritov. Tracking many objects with many sensors. In *Int. Joint Conf. on Artificial Intelligence*, pp. 1160–1171, 1999.
26. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, CA, 1988.
27. F. Pedersini, A. Sarti, and S. Tubaro. Multi-camera parameter tracking. *IEE Proceedings Vision, Image & Signal Processing* **148**, 1 (2001) 70–77.
28. J. Ruiz-Alzola, C. Alberola-Lopez, and J. R. Corredera. Model-based stereo-visual tracking: Covariance analysis and tracking schemes. *Signal Processing* **80**, 1 (2000) 23–43.
29. N. Ukita and T. Matsuyama. Real-time cooperative multi-target tracking by communicating active vision agents. In *Int. Conf. on Pattern Recognition*, pp. II 14–19, 2002.
30. C. R. Wren, A. Azerbayejani, T. Darrel, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **19**, 7 (1997) 780–785.

Biographical Sketch and Photo



Ben Kröse is an Associate Professor at the University of Amsterdam, where he leads a group in Computational Intelligence and Autonomous Systems. He teaches courses on Computational Intelligence and Neurocomputing, and supervises 4 Ph.D. students and many M.Sc. students. As of 2004 he is part time lecturer at the Institute of Information Engineering, where he leads a group in the field of Digital Life and Smart Homes. He published numerous papers in the fields of learning and autonomous systems. He is member of IEEE, Dutch Pattern Recognition Association and Dutch AI Association.



Wojciech Zajdel received the M.Sc. degree in computer science from AGH University of Science and Technology, Poland in 2001. Since 2001 he is working toward a Ph.D. degree in computer science at the Intelligent Autonomous Systems Group at University of Amsterdam. His research involves probabilistic graphical models for multi-camera multi-object tracking. He studies low-level computer vision methods for detecting people and higher-level data association techniques.