Goal-Based Person Tracking Using a First-Order Probabilistic Model

Thomas Geier, Stephan Reuter, Klaus Dietmayer and Susanne Biundo
Faculty of Engineering and Computer Science
Ulm University, Germany
forename.surname@uni-ulm.de

Abstract

This work addresses the problem of person tracking using additional background information. We augment a particle filter-based tracking algorithm with a first-order probabilistic model expressed through Markov Logic Networks to tackle the data association problem in domains with a high occlusion rate. Using a high-level model description allows us to easily integrate additional information like a floor plan or goal information into a joint model and resolve occlusion situations that would otherwise result in the loss of association. We discuss the engineered model in detail and give an empirical evaluation using an indoor setting.

1 Introduction

This work concerns the problem of providing background-knowledge to the specialized application of person tracking using a high-level probabilistic model. We demonstrate that a hand-crafted model, built using Markov Logic Networks (MLN) [Richardson and Domingos, 2006], can help in solving the data association problem in tracking during situations where high occlusion prevents the correct association between past and new tracks. By leveraging additional information, like a floor plan or knowledge about goals of single persons, we can resolve otherwise opaque situations.

The usage of a first-order probabilistic model like MLNs allows for an easier modeling task, because dependencies are represented by weighted first-order logical formulas instead of, e.g., conditional probability tables for the case of directed models like Bayesian Networks. In addition, the model is formulated in a lifted form and can be instantiated for the desired number of concurrent tracks or persons within the scene, which is not possible when using completely propositional models or specialized template models like dynamic Bayesian networks [Murphy, 2002](which can scale only along the time axis). A model given in a lifted representation also makes it possible to leverage structure information contained within the lifted formulation for more efficient inference [Gogate and Domingos, 2011]; although this approach is not investigated here.

We motivate our work in the context of an indoor situation, where multiple persons move in a two-room office, containing the laser range finder and several areas of interest like a printer or a coffee maker. We measure the quality of our model by the extent to which it is able to correctly associate object tracks emerging from the tracking algorithm with the correct persons inside the scene.

The rest of the paper is laid out as follows. After we discuss related work, we give an overview of the applied tracking algorithm and introduce the concept of Markov Logic Networks. Then, we describe the investigated problem in detail and discuss the used MLN model. We give an empirical evaluation of the described setup and conclude with some possible extensions to the model and an overall discussion.

2 Related Work

In multi-object tracking, state dependent detection probabilities of objects are disregarded in most applications. Thus, a track disappears shortly after entering an occluded area and a new track is created when the object leaves the occluded area again. Consequently, one object is represented by different track IDs. Especially in scenarios where persons interact several times with a system, changed track IDs lead to the loss of the objects history. The multi-object Bayes filter [Mahler, 2007] allows to integrate state dependent detection probabilities even if the scenario is characterized by a high object density [Reuter and

Dietmayer, 2011]. In case of short term occlusions, the usage of state dependent detection probabilities leads to an improved track continuity. A direct integration of goals into the prediction of a persons' state is crucial, since the persons' action may be contradictory to the assigned goal.

Markov Logic Networks have been used by Sadilek and Kautz [2010] for multi-agent activity recognition based on GPS data in a game of capture the flag. While their work can leverage more expert knowledge (the rules of the game), they do not encounter the data association problem present in the tracking scenario, since each person was carrying a personal GPS receiver. Tran and Davis [2008] apply Markov Logic Networks to a parking lot surveillance scene using video data to recognize which person enters which car. They also track pedestrians across a scene and face the problem of data association. Their sensory information emerges from image data and their focus lies in integrating different information sources that are all extracted from the video stream. Markov Logic Networks are also used by Singla and Domingos [2006] for entity resolution in text mining. This is the problem of inferring which references refer to the same entity and it is similar to the data association problem in tracking. The two latter works use an `equals` predicate for identity maintenance, whereas we approach the problem using an association mapping to underlying entities. When grounding the model our approach only creates associations between currently instantiated track IDs and their corresponding entity, wheres using an `equals` predicate will introduce relations between all objects, which does not seem reasonable in a dynamic domain.

3 Multi-Object Tracking

Standard multi-object tracking algorithms often use object individual single-object trackers like the Kalman filter. The drawback of this multi-object tracking approach is the need of a data association step which assigns the received measurements to the trackers using hard decisions or probabilistic methods [Blackman and Popoli, 1999]. Especially in scenarios characterized by a high object density, the data association is error-prone and degrades the performance of the tracking system, since false associations are irreversible.

A rigorous approach to multi-object tracking is the multi-object Bayes filter proposed by Mahler [2007]. The multi-object Bayes filter uses the random finite set statistics to represent the complete environment by a single filter state. In the innovation step of the multi-object Bayes filter, a multi-object likelihood function calculates the affinity between the predicted state set and the received measurement set. Thus, no data association is necessary.

Further, the multi-object Bayes filter allows to integrate state dependent detection probabilities into the filtering algorithm. In Reuter and Dietmayer [2011], an approach to calculate state dependent detection probabilities based on the occupancy grid mapping approach [Thrun et al., 2005] is proposed. Thus, it is possible to keep track of an object which is occluded for the sensor for a short period of time. Using constant detection probabilities would lead to a track loss, if an object is not visible to the sensor for a few measurement cycles. We use the state dependent tracking algorithm as a comparison for our final results. But for input into the high-level model, we use state independent object tracking. This produces more track IDs for association, and in particular is less prone to false association, which we cannot correct in the upper stage.

An implementation of the multi-object Bayes filter is possible using Sequential Monte Carlo (SMC) methods [Reuter and Dietmayer, 2011, Sidenbladh and Wirkander, 2003, Mahler, 2007]. In difference to well known SMC implementations of the standard Bayes filter a particle set, which represents a random finite set using a finite number of state vectors, is used instead of a standard particle. Further, the number of state vectors in the particle set may change at each time step. In case of a SMC implementation, the integration of the mentioned constraints is possible by reducing the weight of a particle set.

Since the multi-object Bayes filter does not perform a measurement to track association, an extraction of the individual objects out of the multi-object posterior density function is necessary, e.g. using the k-means algorithm [Bishop, 2006].

4 Markov Logic Networks

Markov Logic Networks [Richardson and Domingos, 2006] are a member of the family of first-order probabilistic languages [de Salvo Braz et al., 2008] and their semantics are based on undirected graphical models (Markov networks). In contrast to propositional models like Bayesian networks and Markov Networks, where every random variable has to be specified explicitly, in first-order models the random variables are relations over objects and the model can be scaled by providing the appropriate number of object constants. Moreover, MLNs allow the specification of dependencies as weighted first-order logical formulas. Higher weights make those interpretations more likely, in which more groundings of the formula evaluate to true. We will now briefly cover the formal semantics of MLNs.

A *Markov Logic Network* $L = \{(f_1, w_1), \dots, (f_n, w_n)\}$ for $n \in \mathbb{N}$ is a set of first-order formulas $f_1, \dots, f_n$ with given weights $w_1, \dots, w_n \in \mathbb{R}$. Together with a finite set of constants C, they define a probability distribution over all interpretations (or possible worlds). An interpretation maps each grounding of each predicate to a truth value. The interpretation of functions must be fixed. Probabilistic functions can be emulated using predicates. Let $g_C(f)$ be the set of groundings of formula f obtained by replacing the free variables in f by all combinations of constants from C. Given an interpretation x, then $n_{C,i}(x) \overset{\text{def}}{=} |\{g \mid g \in g_C(f_i) \text{ and } x \models g\}|$ is the number of groundings of formula f_i that are true under x. Then, the probability distribution $P_{L,C}$ that is defined by the MLN L with constants C is given as

$$P_{L,C}(X = x) \overset{\text{def}}{=} \frac{1}{Z} \prod_i \exp\big(w_i n_{C,i}(x)\big), \qquad (1)$$

where i ranges over all formulas in L, and Z is a normalizing constant.

Given a set of constants, a MLN can be converted to a Markov network, where nodes correspond to atoms and each ground formula induces a clique over all nodes whose atoms appear inside this formula. For practical reasons, a sorted (or typed) logical language is used to describe MLNs. Using sorted terms, we can limit the size of the grounded network. Also, in their basic form, MLNs do only allow restricted usage of logical functions. Usually functions are simulated by specially marked predicates, which enforce a functional dependency of one or more arguments on the remaining arguments. We notate functional arguments of predicates by underlining them. Such a predicate can be translated to a multi-valued random variable.

5 Problem Description

We consider an indoor scene which resembles an office setting. The corresponding floor plan is depicted in Figure 1a. A laser range finder is placed in one corner of the main room and provides distance information in a plane about one meter above ground. The beam almost completely covers the main room, but there exists a second room that has virtually no sensor coverage. There is only one entrance to the room complex and the separate room has only a single exit, which is the door to the main room. The setting contains several features that may serve as goal destinations for persons navigating inside the scene; like a printer or a coffee maker. Knowledge about destinations of persons is taken as given; although it is easy to motivate the existence of such information depending on the application. Issued print jobs could be recognized by a special program installed on the PC, or visits to the coffee maker could be predicted from personal habits.

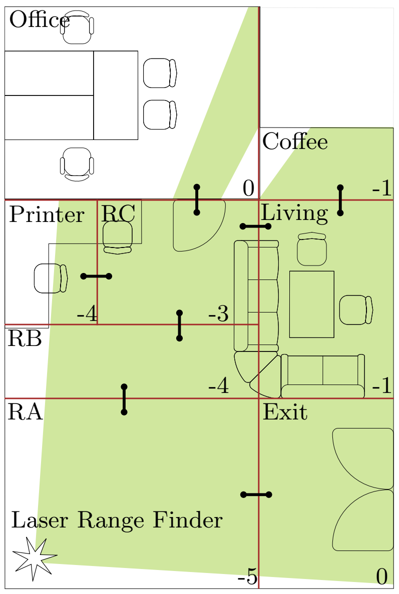
There are one to three persons inside the scene simultaneously. Major occlusion caused by static objects, like walls, occurs when people enter the second room. Minor static occlusion can occur near the coffee maker. During the scenes with more than one person, dynamic occlusion occurs when persons are covered by other persons standing between them and the sensor.

Using only the particle filter-based tracking algorithm to process the output of the laser range finder, problems arise when people produce no measures for an extended period of time because they are inside the separate room or because they are hidden by another person. For shorter occlusion durations it is possible to keep the track of a single occluded person alive for long enough for the person to reappear and re-association is completely handled by the tracking algorithm. If two persons enter the same occlusion area, their estimated positions begin to mix spatially and once they emerge again, re-association becomes more and more arbitrary with increasing occlusion duration. In these scenarios, a direct integration of the Social Force model into the prediction of the persons state of the tracking algorithm may increase the performance of the system [Reuter and Dietmayer, 2010]. The Social Force model aims at describing pedestrian movement by virtual forces exerted by other persons and environmental features [Helbing and Molnar, 1995]. Since the model heavily depends on the destinations of the person, a tight integration with a high-level knowledge base, as described in this work, seems promising for such an approach.

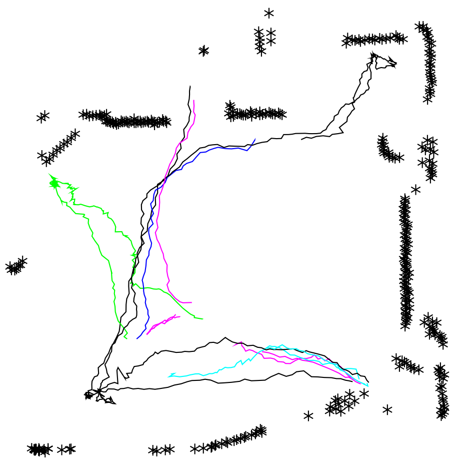
Figure 1b shows an example of the tracking results for one of the sequences with three persons. The trajectories are illustrated by solid lines. Since the results are generated without the usage of the state dependent detection probability, the trajectories are interrupted quite often in the area corresponding to region RA, where a dynamic occlusion occurs.

6 Description of the Model

In this section, we describe the used MLN model and discuss some of the difficulties and design choices we have encountered in its engineering. The complete model is factored into four modules. We begin with a discussion of two concepts that cannot be associated with distinct model parts but influence nearly every aspect – the representation of space and time.



(a) Floor plan



(b) Example trajectories

Figure 1: (a) Floor plan of the location used for the experiments. In addition the picture shows the eight discrete regions that are used for the MLN model, how they are connected via the handles and their static occlusion value in their lower right. (b) Example trajectories of a three person sequence: trajectories are illustrated by solid lines in different colors. The black stars are measurements of static objects like walls.

From continuous to discrete space. The basic MLN can only represent discrete random variables. There exists an extension of MLNs to continuous variables [Wang and Domingos, 2008], but no working implementation is available. For this work, we reduce the continuous spatial estimates obtained from the tracking algorithm to a few discrete regions. For ease of modeling and processing of data, we choose a rectangular shape. We do not create a uniform grid, but try to respect functional aspects of the environment concerning the problem. For example, it does not make sense to further split the office into smaller areas if there is no distinction for the sensor (everything is one connected occluded area) and there is only a single goal inside. Even having two separate goals like two work stations might not justify the introduction of separate regions for each, as long as the rest of the model can not discriminate between them. We have defined a total of eight regions, which are depicted in Figure 1a. Sticking with a low number of regions also made an exact evaluation of the final model feasible. Depending on the inference approach, there might be no significant overhead when using a larger state space for the spatial component. Although, the model engineering may become more intricate when opting for more fine grained regions. Taking the characteristics of the sensor into account, a radial layout for regions seems like a promising approach, but this was not investigated.

Representation of time. In order to model dynamic domains, we assign a dedicated time sort, whose constants are elements from the natural numbers. One usually aims to construct a model that fulfills the Markov property, i.e., the state at time $t+1$ only depends on the state at time t. This means that formulas may only contain predicates of at most two different times, which then must be successive. But in the presented model, the predicates that represent the association of tracks to persons are not time-indexed, which makes them static. This makes it difficult to apply standard dynamic inference algorithms, which usually assume the Markov property. But a static variable can be considered as a dynamic variable, for which the same value is enforced in every time step. Fortunately, these static association are only referenced over a limited period of time — the life-time of a track — so they do not pile up over the course of the complete sequence. For time resolution we have settled for the duration of about one second, which seems like a good compromise between inference complexity and accuracy for the given problem.

The tracking model. The basic functionality for interfacing with the tracking algorithm is provided by a MLN module that contains objects of the sorts `Track` and `Person`. To notate variables of some sort, we use the initial letter of the sort name in lower case. For the sort `Track`, the letter 'm' is used because of the ambiguity with sort `Time`. For both sorts `Track` and `Time`, there exist time-dependent predicates $\mathtt{at}_T : \mathtt{Time} \times \mathtt{Track} \times \underline{\mathtt{Region}}$ and $\mathtt{at}_P : \mathtt{Time} \times \mathtt{Person} \times \underline{\mathtt{Region}}$ that give the current location of a track or person, respectively. The time-independent predicate $\mathtt{a} : \mathtt{Track} \rightarrow \mathtt{Person}$ associates `Track` objects to `Person` objects. The correspondence of tracks to persons inside the MLN is similar to the correspondence of measurements to tracks inside the tracking algorithm. The output of the tracking algorithm is converted to observations of the $\mathtt{at}_T$ function and an additional completely observed predicate $\mathtt{act} : \mathtt{Time} \times \mathtt{Track}$, whose purpose is mainly to be able to prune groundings of formulas that depend on inactive track IDs. The usage of this predicate is omitted for clarity. Information about goals of persons can attach to the location of the person objects. The core tracking model then consists of the following two formulas.

$$5 \quad \mathtt{at}_T(t, m, \underline{r}) \wedge \mathtt{a}(m, \underline{p}) \Rightarrow \mathtt{at}_P(t, p, \underline{l}) \qquad (2)$$

$$-3 \quad \mathtt{a}(m_1, \underline{p}) \wedge \mathtt{a}(m_2, \underline{p}) \wedge m_1 \neq m_2 \qquad (3)$$

Formula 2 probabilistically forces a person to be in the same region as its associated track. By design a track can only be associated to one person at a time because the last parameter of the predicate `a` is declared functional. Formula 3 probabilistically enforces the association to also be a one-to-one relation. The engineering of the formula weights is explained at the end of this section. This formula is limited to concurrently instantiated tracks using the `act` predicate (not listed).

The floor plan. We use the static predicate $\mathtt{adj} : \mathtt{Region} \times \mathtt{Region}$ to encode the connectedness of the regions. All instances are fully observed and adhere to the floor plan given in Figure 1a. A single formula forces persons to move between regions only according to the given layout:

$$\infty \quad \mathtt{at}_P(t, p, \underline{l_1}) \wedge \mathtt{at}_P(t+1, p, \underline{l_2}) \Rightarrow \mathtt{adj}(l_1, l_2) \quad (4)$$

This formula is deterministic to prevent persons from "teleporting" through the scene. If regions allow for a traversal in less than a second (one time step), this rule becomes invalid. But since the association of tracks to persons allows for some slack, a person in the model can "catch up" to the location of its real counterpart after some time steps, only violating Formula 2.

The occlusion model. To prevent persons without an associated track from wandering across the scene (since no track influences their current location), we need to express that persons usually have a track unless they are indeed occluded. In our setting both static and dynamic occlusions occur, being caused by walls or other persons, respectively. For our experiments, only static occlusion information is modeled. This is done by assigning a certain probability to each region that it may contain untracked persons. The probability is larger for areas of high static occlusion, like the separate room. We also assign a higher occlusion to regions that are more likely to be dynamically covered, like the region around the coffee maker. By assigning low occlusion probabilities to central regions that have a good sensor coverage we penalize persons silently slipping past the sensor. Formula 5 is provided once for each region r. The weight w_r is the occlusion value, which is given in Figure 1a.

$$w_r \quad \mathtt{at}_P(t, p, \underline{r}) \wedge \neg \exists m : (\mathtt{act}(t, m) \wedge \mathtt{a}(m, \underline{p})) \quad (5)$$

Goals and their dynamics. The last model part handles the goals of persons. We associate goals with regions and add another time-dependent function $\mathtt{goal} : \mathtt{Time} \times \mathtt{Person} \times \underline{\mathtt{Region}}$, where goals are also allowed to assume the special location `NULL` to signal that a person currently has no goal. The following formulas describe the dynamics of goals:

$$\infty \quad \mathtt{goal}(t, p, \underline{l}) \wedge \neg \mathtt{at}_P(t, p, \underline{l}) \Rightarrow \mathtt{goal}(t+1, p, \underline{l}) \, (6)$$

$$\infty \quad \mathtt{goal}(t, p, \underline{l}) \wedge \mathtt{at}_P(t, p, \underline{l}) \Rightarrow \mathtt{goal}(t+1, p, \underline{l}) \vee \mathtt{goal}(t+1, p, \underline{\mathrm{NULL}}) \quad (7)$$

$$0.1 \quad \mathtt{goal}(t, p, \underline{\mathrm{NULL}}) \qquad (8)$$

Formulas 6 and 7 achieve that goals can only be cleared when a person reaches their associated region. Formula 8 encodes the urge of people to clear their goal. Stating this rule in this particular form results in persons trying to reach their goal as soon as possible, since otherwise the penalty accumulates over time. Another way to make people reach their goals is to make it more likely for a person to be inside the region of their goal. Both rules work equally well for our dataset but might make a difference when applied to longer sequences or under a different setting.

Elicitation of weights. There exist two major ways to determine the weights of the probabilistic formulas: Learning from data and elicitation from experts; where for common sense domains, like the one we are dealing with, everyone is usually an expert. Both the learning of weights and the direct specification approaches have been followed in the literature. For the case of our related work, Sadilek and Kautz [2010] and Singla and Domingos [2006] are employing learning and Tran and Davis [2008] specify the weights by hand. Due to the limited size and the common sense nature of our dataset we decided to specify the weights ourself.

The approximation to consider the weight as the logarithmic odds of the formula being true [Richardson and

Domingos, 2006] can serve as a good starting point, but it only holds as long as formulas do not share predicates. After assigning some reasonable initial values, we iteratively looked at predictions of the model for selected sequences and adjusted the weights if the predictions did not conform with our expectations. We began by adjusting the weights for sequences with only one persons and switched to larger test sequences once the model made sensible predictions for the training data at hand. Since MLNs are based on undirected graphical models (which means they are locally unnormalized), there are no absolute correct values, but the weights of different formulas have to be balanced against each other.

7 Preprocessing of Tracking Information

We go on and describe how tracking data is processed for input to the MLN model. After extraction of the individual objects in the multi-object Bayes filter, we obtain a set of single object particles X_m^t for each track ID m and time step t. We then apply two data reduction steps. First, the MLN model works on a coarser time scale of 1.25 steps per second, while the tracking algorithm runs with 12.5 steps per second. We drop the intermediate steps without further processing. A different approach might aggregate them, e.g., by averaging, but this would also distort the meaning of the data, because it cannot be considered a snapshot of the situation anymore.

Depending on the quality of the tracking algorithm and the used object model, there can be many false positive tracks, e.g., when people spread their arms away from their body, crossing the plane of laser beams. To reduce these false tracks, we use the existence probability to eliminate insignificant tracks. It is given by $|X_m^t|/N$; the number of particles for track ID m divided by the total number of particles N. We drop all tracks from a time step whose existence probability is below 0.5. For our test sequences, the output of the tracking algorithm usually contains about thirty tracks per sequence, but only less than ten remain after applying both reduction processes.

For each time step t and each track ID m that survive the described process we add the track as active to our MLN model via observation of the `act` predicate. We then bin the single object particles into the discrete regions. Most of the time all particles are contained in a single region and we create an observation of the $\mathtt{at}_T$ function. In cases where the particles of a track m spread over several regions we reflect this as probabilistic evidence by adding a formula $(w_l, \mathtt{at}_T(t, m, \underline{l}))$ for each location l and calculating the weight as the logarithmic odds $w_l = \log \frac{p_l}{1-p_l}$, where p_l is the relative frequency of a particle of track m being in region l.

8 The Inference Problem

Markov Logic Networks can be seen as template models for undirected graphical models [Koller and Friedman, 2009]. Their semantics are defined using the ground version of these networks. As such the described MLN represents an undirected version of a dynamic Bayesian network [Murphy, 2002]. The effort for exact inference in such models is usually exponential in the number of variables within one time slice, because most variables within one time step become dependent on each other after some steps in most models.

The model described in this work also suffers from this problem. The cause that all variables of a time slice become dependent lies in the probabilistic data association; which is a hard problem at its core. In our case the problem of exact inference is exponential in $m_T + n_P$, where m_T is the maximum number of simultaneous tracks and n_P is the total number of persons in the model. Here m_T stems from the association predicates and n_P are the instantiations of the $\mathtt{at}_P$ predicate for one time step.

For our evaluation we perform exact inference on the model by exploiting context-specific independence [Koller and Friedman, 2009, pp. 171]. Given an assignment to all association variables, the model factorizes into components for each person and thus becomes tractable. Our largest sequence contained only 10 tracks, which results in 3^7 possible associations to three persons after observing the correct association for three initial tracks. After conditioning on the association variables we calculate the partition function for each association using variable elimination along a min-degree variable ordering. This approach is not suited for online filtering. For this purpose a rao-blackwellized particle filter, which collapses all but the association variables, seems like a good solution [Koller and Friedman, 2009, pp. 526]. Evaluating the performance of this inference approach on the presented model is open for future work.

9 Evaluation

We recorded 9 sequences in total; three sequences with one, two and three persons, respectively. The duration of each sequence is about one minute. The course of events is the same among sequences with the same number of persons; the sequences vary during the part where multiple persons wander around the main room.

The setups with one person only feature static occlu-

		Unassigned Tracks		Model M		Model MO		Model MOG	
Sequence	Persons	$\downarrow p_D = c$	$\downarrow p_D(x)$	$\downarrow a_-$	$\uparrow p_{\text{true}}$	$\downarrow a_-$	$\uparrow p_{\text{true}}$	$\downarrow a_-$	$\uparrow p_{\text{true}}$
S1-1532	1	1	1	0	0.34	0	0.99	0	1.00
S1-1640	1	1	1	0	0.34	0	0.99	0	1.00
S1-1737	1	1	1	0	0.34	0	1.00	0	1.00
S2-2056	2	3	2	2	0.10	2	0.27	0	0.54
S2-2207	2	4	4	3	0.00	4	0.01	3	0.03
S2-2329	2	3	3	2	0.09	2	0.26	0	0.51
S3-4628	3	5	1	3	0.25	0	0.45	0	0.89
S3-4734	3	5	3	2	0.15	0	0.52	0	0.55
S3-5306	3	7	4	1	0.13	1	0.12	1	0.25

Table 1: For each of the nine sequences used for evaluation, we give the number of invented tracks minus the number of persons for tracking with constant detection probability $p_D = c$ and with state dependent detection probability $p_D(x)$. For the three MLN models, we give the number of false associations a_- and the probability of the true assignment p_{true}.

sion caused by walls, caused by the single person staying inside the office for several seconds. This results in its track being reinvented upon entering the main room again. With two persons there is dynamic occlusion, where one person covers the other person. Both persons enter the office together and thus cannot be distinguished once they reappear. Goal information for one person can resolve this issue and we can obtain a good association again. In the scenes with three persons, one person enters the office while the two remaining persons stay inside the main room. Dynamic occlusion occurs while all three persons are walking around in front of the sensor. Tracks are lost and recreated often, which can also be observed in Figure 1b inside the area corresponding to region RA. When two persons are simultaneously shadowed by the third one, it is not possible to associate the reappearing tracks to the correct persons just by means of the sensor. In this case goal information can be used to identify the correct association.

We have evaluated three models that incorporate an increasing amount of domain knowledge. The first model `M` uses only the basic tracking model and the floor plan. The second model `MO` comprises all of the first model and the static occlusion model. The third model `MOG` adds information about personal goals. One person visits the coffee maker in each sequence. We assign the `Coffee` region as goal for this person in every sequence. All other persons have no goal assigned. The goal information is able to resolve confusions that happen before the designated person visits its goal region. This does not happen in every sequence.

The MLN is instantiated for three persons in every setup, regardless of the number of persons appearing in the scene. For each model and each sequence, we observe the correct association for the first track of each person and evaluate how well we can associate new tracks. In our dataset, the number of tracks that remain unassigned after labeling the starter tracks varies between one and seven.

The results of our evaluation are given in Table 1. For each sequence, we give the number of false track assignments of the most probably association. In addition, we provide the probability of the correct joint association. This is interesting for cases where no false associations were made even with a simpler model, and can show an improved significance of the correct association when using a more sophisticated model.

To provide a baseline, the number of track confusions and losses of a state-of-the-art multi-object Bayes filter with state dependent detection probability ($p_D(x)$) and the ones using the same filter with constant detection probability ($p_D = c$) are given [Reuter and Dietmayer, 2011]. For both filters, the number of persons inside the scene is subtracted from the total number of significant tracks. If the tracking algorithm works perfectly, this number will be zero. The output of the $p_D = c$ filter is used as input for the MLN stage; so this number equals the number of associations made by the high-level model and thus equals the possible maximum number of false associations.

We observe that the algorithm with the state dependent detection probability reduces the number of unassigned tracks dramatically in the scenarios with three persons, where a lot of short term occlusions occur. In the scenarios with one or two persons, where the long-term occlusions due to static objects dominate, the usage of the state dependent detection probability has nearly no influence on the number of unassigned tracks. The high-level model `MO` using only the floor plan and the static occlusion model delivers results that are at least on par with the state dependent tracking algorithm. Using goal information for one person can further improve the results. By our judgment, this is not possible by relying solely on the data obtained from the laser range finder in general.

One of the two person sequences (S2-2207) shows very bad performance of the MLN model for all three cases. Our investigation has shown that an outstretched arm has caused a significant track that made it through the data reduction process. The relatively high weight on Formula 3 prevented the association of this track to the owner of the arm. Thus, a third person was forced by the model to appear at this spot and remained present over the course of the scene.

10 Conclusion

We have described an approach to solving the data association problem for person tracking with a high-level probabilistic model described using Markov Logic Networks. We showed how to map the output of a regular tracking algorithm into a discrete spatial representation, which makes it easy to attach additional information, e.g., personal goals, and allows the use of inference techniques for discrete probabilistic models.

Especially in scenarios with long-term occlusions, where even the multi-object Bayes filter is not able to continue to track hidden objects, the association using MLN outperforms the tracking-based approach when using exact evaluation. On the other hand, a sophisticated tracking algorithm is adequate in scenarios with occlusions of no more than one second and might be able to scale more easily to larger domains.

In future, we plan to integrate more information out of the knowledge-base directly into the tracking algorithms like, e.g., probabilistically modeled destinations or goals of a person.

Acknowledgements

This work is done within the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” funded by the German Research Foundation (DFG).

References

C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer, 2006.

S. Blackman and R. Popoli. *Design and Analysis of Modern Tracking Systems.* Artech House Publishers, 1999.

R. de Salvo Braz, E. Amir, and D. Roth. A survey of first-order probabilistic models. In D. Holmes and L. Jain, editors, *Innovations in Bayesian Networks*, Studies in Computational Intelligence. Springer, 2008.

V. Gogate and P. Domingos. Probabilistic theorem proving. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence*, pages 256–265. AUAI Press, 2011.

D. Helbing and P. Molnar. Social force model for pedestrian dynamics. *PHYSICAL REVIEW E*, 51: 4282–4286, 1995.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques.* MIT Press, 2009.

R. P. Mahler. *Statistical Multisource-Multitarget Information Fusion.* Artech House Inc., Norwood, 2007.

K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD thesis, University of California, 2002.

S. Reuter and K. Dietmayer. Adapting the state uncertainties of tracks to environmental constraints. In *Proceedings of the 13th International Conference on Information Fusion*, pages 1–7, 2010.

S. Reuter and K. Dietmayer. Pedestrian tracking using random finite sets. In *Proceedings of the 14th International Conference on Information Fusion*, pages 1–8, 2011.

M. Richardson and P. Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

A. Sadilek and H. Kautz. Recognizing multi-agent activities from GPS data. In *Proceedings of 24th AAAI Conference on Artificial Intelligence*, pages 1134–1139, 2010.

H. Sidenbladh and S.-L. Wirkander. Tracking random sets of vehicles in terrain. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, page 98, 2003.

P. Singla and P. Domingos. Entity resolution with markov logic. In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 572–582, 2006.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents).* The MIT Press, 2005.

S. Tran and L. Davis. Event modeling and recognition using markov logic networks. *Computer Vision–ECCV 2008*, pages 610–623, 2008.

J. Wang and P. Domingos. Hybrid markov logic networks. In *Proceedings of the 22th AAAI Conference on Artificial Intelligence*, pages 1106–1111, 2008.