Long Term Tracking of Pedestrians with Groups and Occlusions

Pedro M. Jorge  Arnaldo J. Abrantes  J. M. Lemos  Jorge S. Marques*
ISEL/ISR  ISEL  IST/INESC-ID  IST/ISR

Abstract

This paper describes an algorithm for tracking groups of pedestrians in video sequences. The main difficulties addressed in this work concern total occlusions of the objects to be tracked, as well as group merging and splitting. Since there is ambiguity, the algorithm should be able to provide the most probable interpretation of the data. A two layer solution is proposed. The first layer produces a set of spatio temporal trajectories based on low level operations which manage to track the pedestrians most of the time. The second layer performs a consistent labeling of the detected segments using a statistical model based on Bayesian networks. The Bayesian network is recursively computed during the tracking operation and allows the update of the tracker results, every time new information is available. Interpretation/recognition errors can thus be detected after receiving enough information about the group of interacting objects. Experimental tests are included to show the performance of the algorithm in complex situations.

Keywords: object tracking, video segmentation, video analysis.

EDICS: 2-SEQP, 2-MOTD.

*Corresponding author: address: IST /ISR, Torre Norte, Piso 7, Av. Rovisco Pais, 1049-001 Lisboa, Portugal; email: jsm@isr.ist.utl.pt
Figure 1: Group splitting

1 Introduction

Pedestrian tracking has been extensively studied since it is a key operation in many image analysis systems [2]-[17]. Pedestrian tracking is an easy task when pedestrians appear isolated in the scene. The main difficulties concern occlusions i.e., when the pedestrian is occluded by the scene or by a group of pedestrians and cannot be easily detected. In these cases, it is not possible to track each pedestrian with simple image analysis techniques.

The goal of this paper is to develop a tracking algorithm for moving pedestrians, being able to deal with occlusions and groups. The proposed tracker should be able to recover the track of a pedestrian after its occlusion by the scene or by other objects.

This seems to be possible since an human being is able to solve these situations. How do we do that? To solve ambiguous situations as the ones mentioned before we often wait until objects become separate again. Let us consider an example to illustrate the difficulties. Fig. 1, shows a group of four persons. Suppose the group splits into two subgroups, each one with two persons. At this point it is not easy to know who is in each subgroup. However, if after a while, one of the subgroups separates, then we can reliably identify each active region.

Most tracking systems are not able to solve this problem since they attempt to provide an instantaneous and independent classification of the active regions, detected in the scene. In order to obtain a reliable interpretation of the moving regions we must be able to consider multiple interpretations and delay the decision in order to integrate information along time. This provides an uncertainty propagation strategy which is necessary to cope with the occlusion problem.

The tracking algorithm should have the following properties:

- on-line operation,
- detect and track active regions even in the presence of groups and occlusions,
- recover from groups and occlusions,
- correct wrong decisions when new information is available.

The algorithm proposed in this paper meets these requirements in two steps. First, simple image analysis techniques are used to track moving regions in the video signal. These techniques efficiently track most of the pedestrians in the video signal. However, they can not deal with occlusions or groups. This problem is solved by data interpretation techniques in the second step. The estimation of the pedestrians trajectories in a long time horizon (with groups and occlusions) is obtained by associating elementary trajectories detected by the simple tracking methods. This operation is formulated as a labeling problem modeled by a Bayesian network and solved by probabilistic inference methods.
Both steps can be incorporated in an on-line surveillance system. The first step updates the tracks of the moving pedestrians every new frame and the second step periodically (e.g., every 5 sec) assigns labels (object identifiers) to the detected tracks. Other on-line strategies to achieve these goals (e.g., particle filtering trackers) are described in Section 2.

The paper is organized as follows. Section 2 describes previous work. Section 3 presents the overall ideas of the proposed tracker. Section 4 considers the Bayesian network model. Sections 5, 6 address on line tracking and extensions of the tracker to more complex situations. Section 7 presents experimental results and section 8 concludes the paper.

2 Previous Work

Many tracking systems are based on two stages. First, active regions are detected in the video signal using motion detection algorithms performed by e.g., optical flow segmentation, background subtraction, frame differences or a combination of techniques [4, 7, 17]. These operations can be considered as low level processing since they do not use specific information about the object shape, color or motion. In a second stage, the detected regions are tracked using region association methods which can be considered as middle level processing.

Region association methods attempt to match pairs of active regions in consecutive images. This operation should be able to deal with the birth and death of video objects, ambiguous matching and detection errors. A simple approach consists of using a nearest neighbor tracker which associates each estimated trajectory with the closest moving region [18]. Since the coordinates of the detected regions are corrupted by measurement noise, Kalman filters have been used to reduce the uncertainty about the target position and velocity. However, the Kalman filter is not able to deal with outliers generated by object detection methods. Robust estimation techniques (e.g., the probabilistic data association filter (PDAF) and its extensions to multiple objects [18]) have been used to overcome these difficulties. Another way of propagating the uncertainty under non-Gaussian conditions is based on particle filters which approximate the probability distribution of the unknown parameters (e.g., target location) by a set of samples (particles) drawn from the a posteriori distribution, given the observed data [9, 13]. These methods were first proposed to deal with outliers and non-Gaussian distributions in a single target problems but they were later extended to cope with multiple targets and temporary occlusions [8, 13]. Particle filtering trackers use motion models to describe the evolution of moving objects in the image and to predict their position in the next frame. Motion models play also an important role during occlusions in which the object position can not be measured, since they allow to estimate the object location even in the case in which no observations are available.

Another alternative is the multi-hypothesis tree (MHT) which considers multiple association scenarios and chooses the best at each instant of time [6]. This approach is able to cope with outliers. Furthermore, it also allows delayed decisions i.e., to delay the labeling of an active region in order to improve the performance.

Some of these techniques were initially proposed in a target tracking paradigm, which has been extensively used in radar surveillance systems since the 80s [18]. However, object tracking in video sequences is a different problem. Video objects are much larger than point targets. Therefore there is much more information about the target properties (e.g., color distribution, shape, texture) which is not available in radar systems. As a consequence, the low level operations associated with object detection in video sequences are more reliable. Furthermore, in the case of pedestrians, the motion is slow compared to the frame rate and it may change in unpredictable ways. Therefore it is not easy to accurately predict the object position many frames ahead. These differences make
There is a difference between many of these trackers and the methods proposed in this paper. While many trackers try to accurately estimate the position of each object, the emphasis of the proposed algorithm lies in the correct identification of pedestrians (labeling problem).

Several attempts have also been recently made to deal with groups of objects. For example, in [7] a method is proposed to locate the position of each pedestrian inside the group, using silhouette boundaries. An histogram representation is used in [12] to compute the visibility indices of each pedestrian inside the group and to recognize the group members.

3 Overall Description

The approach followed in this work is based on two steps. First, simple algorithms are used to track moving regions. Second, the object trajectories are labeled in such a way that trajectories associated to the same object receive the same label. Figure 2 shows the two steps for a specific problem in which two persons meet and separate.

It is assumed that the first step can be easily solved by standard image analysis techniques (we use background subtraction with a simple region association technique based on the mutual choice criterion). Short trajectories lasting less than 10 frames are eliminated and considered as false alarms.

Since pedestrian motion is slow comparing with the frame rate of 25 fps, no motion model is explicitly used i.e., we assume that the position of each pedestrian in the next frame is close to the position of the pedestrian in the current frame\footnote{To deal with objects with faster motions (e.g., vehicles, air planes, sport players) the tracker should explicitly use prediction techniques based on dynamic stochastic models e.g., see [9, 13].}

The problem can be formulated as follows: how can we estimate the unknown labels $x = \{x_i\}$ given the observations $y = \{y_i\}$? This can be seen as an inference problem to be solved by Bayesian inference techniques.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{figure2.png}
\caption{Stroke detection (left) and labeling (right)}
\end{figure}
The most probable labeling configuration is obtained by solving\(^2\)

\[
\hat{x} = \arg\max_x P(x|y) \quad (1)
\]

\[
= \arg\max_x p(y|x)P(x) \quad (2)
\]

where \(p(y|x)\) is the observation model and \(P(x)\) is the prior.

Furthermore, the observation model can be written as

\[
p(y|x) = \prod_i p(y_i|x_i) \quad , \quad y_i \in \mathbb{R}^m, x_i \in L_i \quad (3)
\]

assuming that observation vectors of different strokes are conditionally independent random variables if the stroke labels are known.

The main difficulty concerns the choice of a prior \(P(x)\) which accounts for object occlusions and interactions such as group merges and splits. This issue will be further developed in the next section. A second problem concerns the optimization of (2) using efficient optimization techniques, compatible with on-line tracking of objects in video sequences. A third problem concerns complexity issues, in order to keep the computational effort within reasonable bounds.

The approach described in this section separates easy tasks (tracking of isolated regions) from difficult ones (ambiguous cases) which are formulated as labeling problems. Furthermore, it also allows to propagate the uncertainty associated to the labeling operation, in order to improve the final decision. This delay allows to improve the labeling results by taking into account more information (temporal integration) and the interaction among several objects. For example, in the case of the 4 persons group splitting, the features extracted from each person allow to improve the recognition of the others.

The joint distribution of the labels and observations \(p(x, y)\), is modeled using a Bayesian network since it allow us to describe the interaction among objects and the interaction of the detected objects with the background. Furthermore, it also provides a sound basis for probabilistic inference. The Bayesian network works as a long term memory which stores past interactions among the detected trajectories allowing a better decision than the one which could be taken if memoryless techniques were used. Fig. 3 shows the structure of the proposed tracker denoted as BN tracker.

The BN tracker relies on the appearance model of each isolated pedestrian \(p(y_i|x_i)\), assuming that \(x_i\) is a single label. These models are initialized the first time a pedestrian enters the scene and updated using the information of future frames. It is assumed in this work that all the pedestrians enter the scene isolated.

### 4 Bayesian Network Model

This section addresses the automatic generation of the Bayesian network from the video stream. A Bayesian network (BN) is a graph model which represents the joint probability distribution of a set of random variables

\(^2\)sometimes is useful to estimate the second most probable data interpretation as well.
\( x_1, \ldots, x_n \) in terms of simple factors involving few variables \([19]\).

A Bayesian network is defined by a directed acyclic graph (an oriented graph without loops) where each node is associated to a random variable \( x_i \) and the links represent causal dependencies \([20]\). After defining the graph, the user must also specify the conditional distribution of each node \( x_i \) given its parents \( a_i \). This is usually a simple distribution, involving a small number of variables.

The joint probability density function associated to a BN is the product of all the node conditional distributions

\[
P(x) = \prod_{i=1}^{n} P(x_i|a_i)
\]

The Bayesian network is therefore a way to model complex distributions of a large number of variables in terms of simple factors. Furthermore, there are inference algorithms to compute the most probable configuration which efficiently exploit the structure of the probability distribution (4) \((e.g., \text{see} \ [19, 21])\).

### 4.1 Network Architecture

As mentioned before the BN is used to track multiple objects in the video signal. The main question is: how is the BN automatically generated from the video signal? Figure 4a shows an example for the 2 people problem shown in Fig. 2. This model associates a node \( x_i \) to each stroke \( s_i \) detected in the video signal; \( x_i \) is the label of the stroke \( s_i \), i.e., a variable which identifies all the objects associated to the stroke trajectory, e.g., \( x_i = 1 \) if \( s_i \) is the trajectory of object 1 and \( x_i = (2, 5) \) if \( s_i \) is the trajectory of a group of two objects with labels 2, 5. In this model, the unknown variables are the node labels \( x_i \) which can not be directly observed.

The next question is how to define the links which represent causal dependencies. The main idea is simple: two hidden nodes \( x_i, x_j \) are linked if \( s_i, s_j \) may correspond to sub trajectories of the same object or group. Link creation is performed using simple criteria: i) causality (\( s_j \) should start after the end of \( s_i \)); ii) maximum speed (the mean speed of the pedestrian during occlusion should be smaller than a threshold). The speed threshold, allows the system to eliminate fast motions during occlusion times which could not be performed by a pedestrian. The threshold may depend on the image location as well as the average velocity of the pedestrian before the occlusion.

The measurements \( \{y_i\} \) can also be included in the BN since they are considered as realizations of a random variable which depends on the stroke labels \( \{x_i\} \): \( x_i \) is the parent of \( y_i \). Figure 4b shows the complete BN for the 2 people problem. \((r_{x0} \) is a restriction node associated to the split, which will be explained below). The network conveys information about the stroke interaction namely the propagation of causal dependencies in the presence of occlusions, group merging and splitting. It also describes the relationship between the stroke labels \( x_i \) and the image features \( y_i \). Finally, it also guarantees that physical restrictions hold, e.g., the same object cannot belong to two strokes at the same time. This is the role of the restriction nodes explained below.

To keep the network simple each node is only allowed to have a maximum number of two parents and two children (this restriction is relaxed in section 5). This is done by pruning the network. Pruning can be based on several criteria. For example, if a node has more than two children, the links corresponding to larger occlusion times are eliminated since they are considered as less probable.

Assuming that the number of parents and children is limited to two, there are six basic topologies (see Fig. 5). They correspond to the following situations: total occlusion, group merge, group split and three types of merge-splits. The first topology is used to represent the trajectory of a pedestrian (or group) which is being tracked (stroke \( s_i \)), becomes occluded and it is then detected after a while (stroke \( s_j \)). The second topology corresponds
Figure 4: Bayesian network: a) hidden labels; b) full network

Figure 5: Basic topologies: occlusion, merge, split and three merge-splits

to a merge of two trajectories \((s_i, s_j)\) into a single trajectory \((s_k)\). This topology corresponds to a merge of two pedestrians (or groups) into a single group. It also accounts for the possibility of \(x_k\) being the continuation of \(x_i\) (or \(x_j\)) and \(x_j\) (or \(x_i\)) disappears. The third topology corresponds to a split of a trajectory into two trajectories. This accounts for group splits and it may also represent other situations e.g., \(x_j\) (or \(x_k\)) is the continuation of \(x_i\) and \(x_k\) (or \(x_j\)) is a new stroke. The other topologies correspond to merge-splits.

Every time there is a split, a mutual dependency is created between the children \(x_i, x_j\). This happens because the same person can not be in two active regions at the same time. To model this dependency a binary restriction node \(r_{ij}\) is created. This node takes the binary value 1 if we have a consistent labeling (no common labels are assigned to the children nodes) and it takes the value 0 otherwise. This variable guarantees that the network does not produce inconsistent configurations in which the same object belongs to two or more trajectories. To guarantee that the inference algorithm produces a consistent labeling, the restriction nodes are considered as observable nodes with \(r_{ij} = 1\).

4.2 Admissible Labels

Let us consider the computation of the set of admissible labels \(L_i\) for each node \(x_i\). Isolated objects are identified by an integer label \(l\) and groups are characterized by a set of labels of the group members e.g. \((1, 2)\) is a group
Table 1: Admissible node labels: creation (1,2), merge (3), occlusion (4) and split (5,6)

The set of admissible labels for each node is obtained by recursively propagating the labels through the network (see below). This operation depends on the graph links in the vicinity of each node $x_k$ which can be classified into one of the six basic topologies: occlusion, merge, split or merge-splits. Therefore only six label propagation rules have to be defined.

The label propagation rules for the first three topologies are

occlusion:
$$L_k = L_i \cup l_{new}$$

merge:
$$L_k = L_i \cup L_j \cup L_{merge} \cup l_{new}$$
$$L_{merge} = \{a \cup b : a \subset L_i, b \subset L_j, a \cap b = \emptyset\}$$

split:
$$L_k = \mathcal{P}(L_i) \cup l_{newk} \quad L_j = \mathcal{P}(L_i) \cup l_{newj}$$

where $\mathcal{P}(L_i)$ is the partition of the set $L_i$, excluding the empty set. Label propagation in merge-splits is performed by combining the merge and split rules.

In all these examples, $l_{new}$ stands for a new label, corresponding to a new object entering the scene. Table 1 shows the admissible labels for the example considered in this section. For example, nodes 1, 2 correspond to new pedestrians entering the scene, node 3 is a merge of two pedestrians, node 4 is a group occlusion and nodes 5, 6 correspond to a group split.

4.3 Node conditional distributions

There are three types of nodes in the BN network: hidden nodes, observation nodes and restriction nodes. We have to automatically define the conditional probability tables $p(x_i|a_i)$ for each type of node. Let us consider the hidden nodes first since they represent the variables we want to estimate.

4.3.1 Hidden Nodes

Hidden nodes can either have simple labels, representing a single pedestrian, or group labels. The label of a given node is inherited from the parent nodes using one of three mechanisms: occlusion, group merge or group split. In addition we also assume that each node can have a new label corresponding to new objects entering the scene.
To define a probability distribution of the node variable \( x_i \) we must specify the probability of each of the four previous mechanisms (new, occlusion, merge, split) which will be denoted by \( P_{\text{new}}, P_{\text{occl}}, P_{\text{merge}}, P_{\text{split}} \). These probabilities can be set as constant or they may vary according to the stroke geometry and position.

Since there are six basic topologies, we have six types of conditional distributions \( p(x_k|a_k) \). For example, in the case of occlusion (first topology) the node \( x_i \) is either equal to its parent label or it is a new label. Therefore,

\[
P(x_k|x_i) = \begin{cases} 
1 - P_{\text{new}} & x_k = x_i \\
P_{\text{new}} & x_k = l_{\text{new}} 
\end{cases}
\]  

In the other cases, the probability distribution is slightly more complex. For example, if \( x_k \) has a merge topology with parents \( x_i, x_j \), then we must consider 4 hypotheses: i) \( x_k \) receives the label of \( x_i \), ii) \( x_k \) receives the label of \( x_j \), iii) \( x_k \) is a merge of both labels or iv) it is a new label. This leads to

\[
P(x_k|x_i, x_j) = \begin{cases} 
P_{\text{merge}} & x_k = x_i \cup x_j \\
P_{\text{occl}} & x_k = x_i \\
P_{\text{occlj}} & x_k = x_j \\
P_{\text{new}} & x_k = l_{\text{new}} 
\end{cases}
\]

In the case of splits, similar arguments lead to

\[
P(x_k|x_i) = \begin{cases} 
P_{\text{split}}(2^{N_i} - 2) & x_k \subset \mathcal{P}(x_i) \setminus x_i \\
P_{\text{occl}} & x_k = x_i \\
P_{\text{new}} & x_k = l_{\text{new}} 
\end{cases}
\]

where \( N_i \) is the number of objects in the group label \( x_i \). If \( x_k \subset \mathcal{P}(x_i) \setminus x_i \), \( x_k \) is a subset of the group \( x_i \) (we assume that all the \( 2^{N_i} - 2 \) subsets are equiprobable). If \( x_k = x_i \) then the node \( k \) has all the labels of the parent node. In this case, the group does not separate and has simply suffered a temporary occlusion. If \( x_k = l_{\text{new}} \) the stroke \( k \) corresponds to a new pedestrian and is not related to the parent node. When \( x_i \) is a simple label (represents a single pedestrian) this expression is not valid since group splitting should not be allowed, in this case.

The conditional distributions of the other topologies follow similar guidelines and they are defined in appendix. The above parameters \( P_{\text{new}}, P_{\text{occl}}, P_{\text{merge}}, P_{\text{split}} \) are heuristically defined or learned from the data. They can be independently specified provided that their sum is equal to 1 for each type of topology. In this paper, the parameters were heuristically chosen but they depend on the stroke geometry e.g., the occlusion probability is higher if the occlusion time is smaller.

### 4.3.2 Restriction Nodes

Every time there is a split, a binary node in included to create a dependency between the two children: the same object cannot simultaneously belong to two groups at the same instant of time.

The conditional probability table of the restriction node \( r_{ij} \) is created in such a way that \( r_{ij} = 1 \) if there is no labelling conflict (common labels) and \( r_{ij} = 0 \) otherwise. Therefore

\[
P(r_{ij} = 1|x_i, x_j) = \begin{cases} 
1 & x_i \cap x_j = \emptyset \\
0 & \text{otherwise}
\end{cases}
\]
4.3.3 Observation Nodes

Each stroke detected in the video signal is characterized by a set of features e.g., average color histogram, average speed, region shape, key colors obtained by clustering or by the mean shift [5] or invariant features such as Scale Invariant Feature Transform (SIFT) [11].

Let $y_k$ be the set of features extracted in stroke $s_k$. It is assumed that $y_k$ is a random variable which depends on the label $x_k$. Therefore, we must be able to define the conditional probability $p(y_k|x_k)$. This distribution depends on the specific choice we make about the features.

In this paper, we have used a simple set of features consisting of the $M$ ($M=5$) dominant colors obtained by clustering the color components of the pixels of all the active regions associated to a given stroke $s_k$. Each color is represented by two features: the normalized r, g color coefficients. Therefore, $y_k$ is a $2M$ dimensional feature vector. The computation of the dominant colors is performed using the k-means algorithm.

Every time a new pedestrian enters the scene, the set of dominant colors is computed and used to characterize the pedestrian during its evolution in the camera field of view. Therefore, each simple label is defined by $M$ dominant colors $Z_k \in \mathbb{R}^{2M}$. A group of labels is characterized by the dominant colors of all the objects inside the group. We assume that a stroke $s_k$ is well represented by a tentative label $x_k$ if the dominant colors of $s_k$ match the dominant colors of a tentative label $x_k$. Figure 6 shows the color scatter diagram of a given stroke and its dominant colors, computed by clustering in the $rg$ space. It is also shown the dominant colors of a tentative label. The matched colors are represented by circles.

Let $y_{ki}$ be the $i-th$ dominant color of the video stroke $k$ and let $z_{x_kj}$ be the $j-th$ dominant color of the label $x_k$. We will assume that

$$p(y_k|x_k) = \prod_{i=1}^{M} p(y_{ki}|x_k)$$

(13)
where $p(y_k|x_k)$ has a high value $\alpha$ in the case of matched colors and a low value $\beta$ in the case of unmatched ones

$$p(y_k|x_k) = \begin{cases} \alpha & \text{if } \min_j \|y_{ki} - z_{x_kj}\| < \delta \\ \beta & \text{otherwise} \end{cases}$$

where $\delta$ is a matching threshold.

### 4.4 Inference

Inference methods are used to compute the most probable configuration of the hidden nodes (labels) given the observed nodes $\{y_i\}$ and restriction nodes $\{r_{ij}\}$. This is equivalent to the solution of (2). Fortunately, there are several well known methods to compute the most probable configuration e.g., message passing method for polytrees [20] or the junction tree algorithm [19]. The complexity of the inference procedure grows if there are multiple paths. In this paper, inference was performed using the Bayes Net Matlab toolbox, developed by K. Murphy [21].

The system described before is able to track moving objects in video sequences, managing to deal with complex situations (occlusions and groups). However, it has two major drawbacks: it is an off-line algorithm (the network complexity and delay grow to infinity as time goes by) and the topological restrictions (two parents and two children) cannot cope with complex situations involving several objects.

These issues are addressed in the following sections and the proposed algorithm is then experimentally evaluated in real video sequences.

### 5 On-line Tracking

The algorithm described in section 4 is tailored to off-line analysis of video sequences in batch mode. We have to wait until the end of the sequence before doing inference. Furthermore, it cannot be applied to long video sequences as the network complexity and the computational time grow to infinity as time goes by.

A tracking system should provide labeling results in real time, with a small delay. Therefore it is not possible to analyse the video sequence in a batch mode i.e., performing inference after detecting the object trajectories in the whole video sequence.

To avoid these difficulties two strategies are suggested: periodic inference and network simplification. The first strategy consists of incrementally building the network and performing the inference every $T$ seconds. Denoting by $x(0: kT), y(0: kT)$ the variables of the Bayesian network associated to strokes detected in the interval $[0, kT]$, then the object labeling can be periodically performed by solving

$$\hat{x}(0: kT) = \arg \max_{x(0: kT)} p(x(0: kT), y(0: kT))$$

The network grows as before but the labeling delay is reduced to less than $T$ seconds. The solution of (15) can be obtained by standard techniques as before (e.g., junction tree algorithm [19]).

In practice we wish to have an instantaneous labeling of all the objects i.e., we do not wish to wait $T$ seconds for a new global inference. Can we obtain an instantaneous labeling of the trajectories with a Bayesian network?

To obtain on-line labeling a suboptimal approach can be devised which combines the optimal decision obtained at the instant $kT$ with the new information collected after the instant $kT$.

Let $x_i$ be a hidden node associated to a trajectory active in the interval $[kT, t]$. Exact inference is performed by computing $P(x_i|y(0: t))$. In order to derive a suboptimal estimate we shall assume that the information
available to estimate \( x_i \) is the combination of \( y(0 : kT) \) (past) and \( y_i(kT : t) \) (recent) i.e., we consider the recent observations of stroke \( s_i \) and neglect the observations of the other strokes after the last inference instant \( kT \).

Applying the Bayes law,

\[
P(x_i|y(0 : t)) = \alpha p(y_i(kT : t)|x_i) P(x_i|y(0 : kT))
\]

(16)

where \( \alpha = 1/P(y_i(kT : t)|y(0 : kT)) \) is a normalization constant, \( P(x_i|y(0 : kT)) \) is a global prior, computed before in the inference step at time \( kT \) using all the information about the interaction among nodes and \( p(y_i(kT : t)|x_i) \) represents new local information. The choice of the best label \( \hat{x}_i \) is performed by selecting the highest a posteriori probability \( P(x_i|y(0 : t)) \). When \( x_i \) is a new variable i.e., when a new stroke is detected after \( kT \), we assume a uniform prior: no label is preferred based on past information.

The above strategies convert the batch algorithm into a on-line algorithm i.e., they solve the first problem. However, the network size increases as before. To overcome this difficulty a simplification is needed. The main idea used in this work is to bound the memory of the system by freezing a subset of the network nodes with their most probable values.

Old (hidden and observed) nodes influence the labeling assignment of current nodes. However, this influence decreases and tends to zero as time goes by: recent variables are more important than old ones. So, we need to use techniques to forget the past. In this paper, we allow a maximum of \( N \) active nodes and freeze all the other nodes by assigning them the most probable label obtained in previous inferences. In this way, the complexity of the network remains bounded and can be adapted to the computational resources available for tracking.

Several strategies can be used to select the nodes to be frozen (dead nodes). A simple approach is used for this purpose: oldest nodes are eliminated and the most recent \( N \) nodes are kept active.

Fig. 7 shows the evolution of the Bayesian network at three instants, for a PETS 2001 sequence. Although the number of nodes grows linearly with time, only the most recent ones are active and updated by the inference algorithm, therefore keeping the computational burden under control.

### 6 Extensions

Until now we have restricted the number of sons and parents of each node to maximum of 2. This is too restrictive to cope with many practical situations. The key question is: how can we deal with more complex situations as shown in Fig 8a?*

The extension of merge and split topologies defined in section 3 to deal with arbitrary number of parents or children is straightforward. The most difficult problem concerns the merge-splits since there is a combinatorial explosion of different merge-split topologies. In the sequel we will first define the extensions of merge and split nodes to the general case and then address the merge-split problem.

Figures 8b,c show the merge and split topologies with multiple parents and children. The simplest case is the split since the label propagation rules and conditional probability distribution defined in (8,11) remain valid. The merge has different rules however since we must consider the association of \( 2^p - n - 1 \) subsets of parent nodes i.e. we should consider the association of any pair of nodes \( (x_i, x_j) \) any triplet \( (x_i, x_j, x_k) \), etc. The set of admissible labels \( L \) can be recursively computed as follows:

\[
L = L^p \cup L_{new}
\]

(17)

*This restriction applies to hidden nodes only
Figure 7: Evolution of the Bayesian network at three time instants. Gray circles represent frozen past nodes and white circles represent active nodes to be labeled by the inference process.

Figure 8: General network: a) merge-split b) merge; c) split)
Figure 9: Modified Bayesian net with mediating nodes $x_{mi}$

$$L^i = \text{Merge}(L^{i-1}, L_i) \quad i = 1, \ldots, p \quad L^0 = \emptyset$$

where $L_i$ is the set of admissible labels of $x_i$. The Merge(,.) is defined in (7). The set of admissible labels $L$ can be split into three subsets $L_{\text{cont}}, L_{\text{merge}}, L_{\text{new}} = l_{\text{new}}$ associated with stroke continuation, merge and new. The conditional distribution must consider all these three types of labels as follows

$$P(x|x_1, \ldots, x_p) = \begin{cases} P_{\text{occli}} & x = x_j \\ P_{\text{merge}} & x \in L_{\text{merge}} \\ P_{\text{new}} & x = l_{\text{new}} \end{cases}$$

Concerning the merge-splits, it is not possible to explicitly consider all the different topologies when the number of children or parents is higher than 2. A different approach is followed. We will detect all such nodes, $x_k$ in the BN and introduce mediating nodes between $x_k$ and its parents with more children (see Fig. 9). In this way splits and merges become separate and can be dealt with the previous rules. Mediating nodes allow to solve the split-merge problem in an elegant way.

7 Tracking Experiments

The proposed algorithm was used to track multiple pedestrians in video sequences. This section shows tracking results obtained in three video sequences: an indoor sequence extracted from PETS 2004 database ("Meet Split 3rd Guy") [23] and two outdoor sequences, one from the PETS 2001 database (test set example 1 [22]) and one sequence recorded at a university campus. All these sequences have multiple interacting objects (typically less than 10) with group associations and occlusions. The sampling rate is 25 fps.

Inference was performed every 15 seconds using the on-line algorithm\(^4\) and a maximum number of ancestor nodes $N = 10$ which prevents the network from growing. The Bayesian network was automatically built during the tracking operation without human intervention and the inference results were obtained using Murphy Bayes Net toolbox for Matlab [21].

\(^4\)the update rate depends on the application; 15sec is an acceptable delay in some surveillance tasks
7.1 Example 1 - PETS 2001

Figure 10 shows the performance of the tracker in the PETS 2001 sequence sampled at 25 fps, during the first 120 sec. Fig 10a shows the evolution of all active regions detected in the video stream. This figure displays one of the coordinates of the mass center (column) as a function of time. Every time there is an occlusion or when two or more objects overlap it is no longer possible to associate the detected regions with past ones. In such cases the trajectories are interrupted.

Fig. 10b shows the labeling results obtained with the on line algorithm described in the paper with a maximum of 2 parents and 2 children per node. The algorithm manages to disambiguate most of the occlusions well. Only two labeling errors are observed in a total of 34 strokes (3000 frames). The errors are in label 6 and in a switch of labels 3, 8 at \( t = 110 \) sec. The output of the on-line algorithm was compared with the results of the batch version of the tracker. The same labeling was obtained in both cases with important computational savings (CPU times\(^5\): 258 sec (batch) and 12.8 sec (on line)). In this example, the computation time of the labeling algorithm is about 10% of the sequence duration. The statistics of the sequence complexity (number of objects, number of groups, number of tracks,duration) and tracker performance (labeling errors, computational time) are shown in Table 2.

Figure 11 shows an example of the labeling interpretation obtained with the proposed algorithm. This example shows a group merging and splitting involving two vehicles while the other objects are being tracked by the system. This situation is correctly solved.

\(^5\)these tests were performed on a P4 at 2.8 GHz
Table 2: Experimental results: number of strokes (NS); number of objects (NO); number of groups (NG); duration (D sec.); number of labeling errors (NE); computational time (CT sec.)

<table>
<thead>
<tr>
<th>Seq.</th>
<th>NS</th>
<th>NO</th>
<th>NG</th>
<th>D</th>
<th>NE</th>
<th>CT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS 2001</td>
<td>34</td>
<td>8</td>
<td>5</td>
<td>120</td>
<td>2</td>
<td>12.8</td>
</tr>
<tr>
<td>Campus</td>
<td>62</td>
<td>11</td>
<td>10</td>
<td>54.8</td>
<td>2</td>
<td>42.8</td>
</tr>
<tr>
<td>PETS 2004</td>
<td>67</td>
<td>7</td>
<td>4</td>
<td>36</td>
<td>5</td>
<td>26.6</td>
</tr>
</tbody>
</table>

Figure 11: PETS 2001: labeling results at time instants (in sec.) a) 32.8; b) 42.4; c) 49.2; d) 49.9; e) 57.6 and f) 60.8.
7.2 Example 2 - University Campus

This sequence contains the trajectories of 10 pedestrians walking in an university campus for 54.8 s. The number of groups (10) is much larger than in the previous example. Figure 12 shows the detected strokes as well as the labeling results obtained by the Bayesian network. The BN tracker managed to solve most of the occlusions and group merging and splitting well. Only two labeling errors were obtained (see table 2).

Figure 13 illustrates the performance of the tracker in a typical situation in which two persons meet, forming a group and then separate (pedestrians with labels 2 and 3).

Figure 7 shows the Bayesian network architecture at three time instants (visible nodes are not represented). Although the number of nodes grows quickly with time, only the most recent ones are updated by the inference algorithm, therefore keeping the computational burden under control. The gray nodes were classified as frozen by the pruning algorithm and their labels and are not allowed to change.

7.3 Example 3 - PETS 2004

Figure 14 shows the performance of the tracker in the indoor sequence 'Meet Split 3rd Guy' (PETS 2004) allowing general topologies i.e., more than 2 parents and 2 children per node. This is a difficult example, useful to illustrate the performance of the tracker in the presence of occlusions, group merging and splitting. The low level operations needed for object detection produce detection errors (mainly false positives) due to drastic illumination changes and the presence of static objects (persons) which remain undetected during some intervals of time and generate small active regions due to small movements. Even under these conditions the BN tracker manages to produce
good results.

Fig. 14a shows the evolution of all active regions detected in the video stream as in previous examples. Fig. 4b shows the labeling results obtained with the on-line algorithm described in the paper. The BN tracker manages to disambiguate most of the occlusions well (only the yellow stroke is misclassified).

Figure 14b shows examples of the tracker performance in group merging and splitting. This sequence has three moving objects (3,4,6) and three static objects. The tracker manages to correctly track the three moving objects most of the time as shown in Fig. 15. Three persons walk in separately (Fig. 15a), they merge in groups of two (Figs. 15b,c,e) and they split after a while (Figs. 15d,f). All these events are correctly interpreted by the tracker. Namely, the correct label is assigned after the two splits of Figs. 15d,f.

The tracker has some difficulty to deal with the static objects (labels 1,2,5) since they are not correctly detected by the low level algorithms (background subtraction). These objects remain in the same place during the whole sequence. They should be considered as background. However, there are small movements which are detected and appear in Fig. 14.

The performance of the BN tracker in this example is summarized in table 2.

We have tried to repeat the experimental tests without freezing past nodes but the amount of memory needed increases exponentially and exhausts the computer resources. This is an important issue to be addressed in the future.
Figure 14: PETS 2004: a) detected strokes; b) most probable labeling obtained with the on line algorithm.

Figure 15: PETS 2004: labeling results at time instants (in sec.) a) 9.6; b) 12.3; c) 14.4; d) 15.0; e) 15.5 and f) 17.3.
8 Conclusions

This paper presents a system for long term tracking of multiple pedestrians in the presence of occlusions and group merging and splitting. The system tries to follow all moving objects present in the scene by performing a low level detection of spatio-temporal strokes followed by a labeling procedure which attempts to assign consistent labels to all the strokes associated to the same pedestrian. The interaction among pedestrians is modeled using a Bayesian network which is automatically built during the surveillance task. This allows to formulate the labeling problem as an inference task which integrates all the available information extracted from the video stream, allowing to update the interpretation of the detected tracks every time new information is available. For example, when a group of objects splits, there is not enough information to perform a reliable identification of each object. However the proposed system is able to improve its output, when additional information is extracted from the video signal.

The proposed system is able to deal with occlusions of pedestrians by static objects or by other pedestrians forming groups. The system estimates the identifier (label) of each isolated object (or group) after the end of the occlusion.

The labeling model used in this paper (Bayesian network) is not time driven. It is event driven: it tries to assign labels to the object trajectories (strokes) detected by simple low level operations. Therefore, only a single variable is used to identify each stroke even if the stroke lasts for tens or hundreds of frames. The Bayesian network does not try to describe the evolution of labels from frame to frame. It describes data conflicts: occlusions, group merging and splitting.

The performance of the system does not depend on the occlusion duration in the case of groups. In the case of occlusion by static objects the performance is also independent of the occlusion duration, provided that it does not exceed a threshold defined by the user.

Future work should consider several issues. Complexity is the most important problem to be addressed in the future. Since the system complexity depends on the video stream, it is important to devise strategies to monitor and control the network complexity. A first step towards this direction is proposed here by network pruning and the use of frozen nodes associated with past information. However, this is not enough since the number of labels associated to each node also increases exponentially. Future work should consider label pruning as well e.g., by discarding the less probable labels, keeping however the model consistency.

The image features should also be improved. The use of dominant colors is a crude representation of the object appearance. More sophisticated features (e.g., local and invariant features [11, 15]) should be studied in this context as tool to improve the systems performance.

The proposed algorithm can be extended to other types of video objects (e.g., cars, sport players) by using a motion model to exploit predictable movements.

References


Appendix

This appendix defines the expressions for the conditional probability tables of merge-splits.

**Merge-split 1:**

\[
P(x_k|x_i, x_j) = \begin{cases} 
  P_{\text{split}}/(2^{N_j} - 2) & x_k \subset P(x_j) \setminus x_j \\
  P_{\text{merge}}/(2^{N_j} - 1) & x_k \subset M_{ij} \\
  P_{\text{occl} i} & x_k = x_i \\
  P_{\text{occl} j} & x_k = x_j \\
  P_{\text{new}} & x_k = l_{\text{new}} 
\end{cases} \tag{20}
\]

where \( M_{ij} \) contains group labels including label \( x_i \) and a subgroup of \( x_j \) without common objects. Specifically,

\[
M_{ij} = \{ a \cup b : a = x_i, b \subset P(x_j), a \cap b = \emptyset \} \tag{21}
\]

**Merge-split 2:**

\[
P(x_k|x_i, x_j) = \begin{cases} 
  P_{\text{split}, i}/(2^{N_i} - 2) & x_k \subset P(x_i) \setminus x_i \\
  P_{\text{split}, j}/(2^{N_j} - 2) & x_k \subset P(x_j) \setminus x_j \\
  P_{\text{merge}}/[(2^{N_i} - 1)(2^{N_j} - 1)] & x_k \subset M_{ij}^* \\
  P_{\text{occl} i} & x_k = x_i \\
  P_{\text{occl} j} & x_k = x_j \\
  P_{\text{new}} & x_k = l_{\text{new}} 
\end{cases} \tag{22}
\]

where \( M_{ij}^* \) contains the group labels of all subset of \( x_i \) merged with a subgroup of \( x_j \) without common objects. Specifically,

\[
M_{ij}^* = \{ a \cup b : a \subset P(x_i), b \subset P(x_j), a \cap b = \emptyset \} \tag{23}
\]

Merge-split 3 is a special case of merge-split 2 when two of the parent nodes are the same. The conditional distribution of the merge-split 3 is therefore given by (22).