A Track Creation and Deletion Framework for Long-Term Online Multi-Face Tracking

Stefan Duffner and Jean-Marc Odobez

***Abstract*—To improve visual tracking, a large number of papers study more powerful features, or better cue fusion mechanisms, adaptation or contextual models, for instance. A complementary approach consists in improving the track management, that is, deciding when to add a target or stop its tracking, for example in case of failure. This is an essential component for effective multi-object tracking applications, and is often not trivial. Deciding to stop a track or not is a compromise between avoiding erroneous early stopping while tracking is fine, and erroneous continuation of tracking when there is an actual failure. This decision process, very rarely addressed in the literature, is difficult due to, for example, object detector deficiencies or observation models that are insufficient to describe the full variability of tracked objects and deliver reliable likelihood (tracking) information. This paper addresses the track management issue and presents a real-time, online multi-face tracking algorithm that effectively deals with the above difficulties. The tracking itself is formulated in a multi-object state-space Bayesian filtering framework solved with Markov Chain Monte Carlo. Within this framework, an explicit probabilistic filtering step decides when to add or remove a target from the tracker, where decisions rely on multiple cues such as face detections, likelihood measures, long term observations, and track state characteristics. The method has been applied to three challenging datasets of more than 9 hours in total, and demonstrate a significant performance increase compared to more traditional approaches (MCMC, RJ-MCMC) only relying on head detections and likelihoods for track management.**

I. Introduction

A. Motivation

The detection and tracking of faces in real-time is of utmost interest in many computer vision applications from different domains, e.g. video-conferencing, Human-Robotic or Human-Computer interfaces or in the analysis of social interaction.

For instance, *effective group-to-group* communication gains increasing attention in modern video-conferencing applications, and requires efficient and robust algorithms to determine the position of a varying number of faces at each point in time, which is the topic of ongoing research like in the project "Together Anywhere, Together Anytime" (TA2). There, several persons sit in front of a camera (Fig. 1), communicate with each other and with one or several remote sites, and perform some shared activity on a touch-table in front of them. Face tracking and other cues are used by a virtual operator component to understand the communication situations and select interesting shots to show to the remote sites. One of the challenges for face tracking here is that the participants do not always look into the camera, and their attention might be on the touch table or on another person in the room.

Fig. 1. Example video frames from the considered application; dataset 1 and 2 (top), and 3 (bottom). Faces may be difficult to detect, and occlusions can occur requiring an effective mechanism to remove and reinitialise tracks.

The most straightforward approach for solving the face tracking problem is to employ a face detector (e.g. [1]). However, despite much progress in recent years on multi-view face detection, these methods are mostly employed in scenarios where people predominantly look towards the camera. As we demonstrate in our results, this is not sufficient for more complex scenarios, where faces are missed around $30-40\%$ of the time due to less common head poses. Unfortunately, the difficult head postures can last for relatively long periods of time (up to one minute in some of our videos). This means that face detection algorithms have to be complemented by robust tracking approaches; not only to interpolate detection results or filter out spurious detection, as is often assumed, but also to allow head localisation over extended periods of time.

Numerous multiple faces tracking methods have been proposed (e.g. [2], [3], [4], [5], [6]), mainly focusing on new features, new multi-cue fusion mechanisms, better dynamics or adaptive models for instance [7], [8], [9], [10], and results are demonstrated mostly on *short* sequences [7], [8], [9], [10].

However, very few of them address track initialisation and termination, especially in terms of performance evaluation. A face detector is often used to initialise new tracks, but how to cope with its uncertain output? A *high* confidence threshold may lead to missing an early track initialisation. Conversely, with a *low* threshold false tracks are likely to occur.

Track *termination* can be even more difficult. How do we know at each point in time if a tracker is operating correctly? This is an important issue in practise, especially since an *incorrect* failure detection can lead to losing a person track for a long time until the detector finds the face again.

This paper explicitly addresses these issues and proposes an effective solution to handle them.

Stefan Duffner and Jean-Marc Odobez are with Idiap Research Institute, Rue Marconi 19, Martigny, Switzerland

The research leading to these results has received funding from the European Community's 7th Framework Programme ICT Integrating Projects TA2, (grant agreement no. 214793) and HUMAVIPS (no. 247525).

B. Related Work

Principled methods exist to integrate track creation and termination within the tracking framework, for example Reversible-Jump Markov Chain Monte Carlo (RJ-MCMC) [11], [12]. But to be effective, they require appropriate global scene likelihood models involving a fixed number of observations (independent from the number of objects), and these are difficult to build in multi-face tracking applications. Experimental results in Section VI show that an RJ-MCMC-based face tracker [11] performs worse than the proposed approach, mainly because it relies only on the likelihood to decide on track creation and deletion, and not on other cues like tracker location uncertainty or long term statistics.

Kalal *et al*. [13] present an interesting approach for failure detection in visual object tracking that is based on the idea that a correctly tracked target can be tracked *backwards* in time. Unfortunately, the backward tracking greatly increases the overall computational complexity (by a factor linear in the backward depth). In a particle filter tracking framework, another solution is to directly model a failure state as a random variable within the probabilistic model [14]. However, this increases the complexity of the model and thus the inference, and it is difficult in practise to model the distribution of a failure state or failure parameters. Closer to our work, Dockstader *et al*. [15] proposed to detect failure states in articulated human body tracking using a Hidden Markov Model (HMM). However, their method differs significantly from ours: they only use one type of observation (the state covariance estimate) which in our case proves to be insufficient for assessing tracking failure; their observation are quantised to use a standard discrete multinomial likelihood model, whereas our method learns these likelihoods in a discriminative fashion; and their HMM structure (number of states, connections) is specifically designed for their articulated body tracking application.

In applications that are similar to ours the problem of deciding when to stop tracking a face is usually solved in a recursive manner. This means, assessing tracking failure is often left to the (sudden) drop of objective or likelihood measures which are not easy to control in practise [16], [17].

In many scenarios of interest, the camera is fixed, and due to the application and the room configuration, people in front of the camera tend to behave similarly over long periods of time. However, most of the existing face tracking methods ignore this long-term information, as they concentrate on video clips that are often not longer than a minute. Or if they use long-term information, it is mainly for constructing stable appearance models of tracked objects [18], [19], e.g. by working at different temporal scales [20]. Similarly, some methods [9], [21] train an (object-specific) detector online, during tracking, to make it more robust to short-term and long-term appearance changes. However this increases the computational complexity, because a separate model has to be built for each person, and each such detector has to be applied on the input image. Recently, Mikami *et al*. [16] introduced the Memory-based Particle Filter where a history of past states (and appearances [17]) is maintained and used to sample new particles. However, they only addressed single, near-frontal face tracking, in high resolution videos and only evaluated the method on 30 to 60-second video clips. Finally, other works (e.g. [22], [23], [24]) tackle the problem of long-term *person* tracking by analysing the statistics of features from shorter tracks (tracklets), and by proposing methods to effectively associate them. These algorithms are different from ours as they process the data *off-line*, i.e. the observations at each point in time are known in advance, and they mainly deal with tracking the position of the *full human body* as opposed to just faces. Another approach for multiple pedestrian tracking [25] associates smaller tracklets *on-line* and in a statistical sampling framework but no principled mechanism for starting and ending tracks is proposed.

C. Contributions

In this paper, we propose a novel multi-face tracking algorithm. It relies on a principled Bayesian filter solved with a MCMC sampling scheme that handles object interactions. The main contributions of the paper are the following:

- an explicit probabilistic filtering framework to decide when to add or remove an object from the tracker based on the output of a detector, long-term image features, and features from the tracker itself (e.g. state variance);
- use of *long-term* image observations to cope effectively with missing or uncertain face detections;
- exploiting static observations (based on current image observations) as well as dynamic ones (temporal evolution of certain features), for tracking failure assessment;
- a thorough performance evaluation on more than 9 hours of videos involving 2 to 5 persons per view, with around $22\,000$ annotations, showing the superiority of our approach as compared to a traditional RJ-MCMC approach to handle variable number of object tracks;
- further comparison of a single object tracker version of our algorithm including failure detection with several state-of-the-art single object trackers [7], [8], [10].

We extensively evaluate the impact of different factors of the proposed method for real-world applications and draw conclusions about the level of importance of these factors.

This paper extends our prior work [26] in several aspects: introduction of new features to assess the tracking status, more thorough description of the algorithms and of the parameter learning, more in depth performance analysis with illustrative results, more extensive experiments and comparison with state-of-the-art single- and multi-object tracking algorithms.

Section II describes our multi-face MCMC particle filter framework. Section III presents our approach for track creation and failure detection. Section IV describes how the algorithm keeps track of person identities. Section V introduces our experimental protocol, while Section VI present our results. Finally, in Section VII we draw our conclusions.

II. Multi-Face Tracking With Particle Filter

We tackle the problem of multi-face tracking in a recursive Bayesian framework. Assuming we have the observations $\mathbf{Y}_{1:t}$

from time 1 to t, we want to estimate the posterior probability distribution over the state $\tilde{\mathbf{X}}_t$ at time t:

$$p(\tilde{\mathbf{X}}_t|\mathbf{Y}_{1:t}) = \frac{1}{C}\, p(\mathbf{Y}_t|\tilde{\mathbf{X}}_t) \times \int_{\tilde{\mathbf{X}}_{t-1}} p(\tilde{\mathbf{X}}_t|\tilde{\mathbf{X}}_{t-1}) p(\tilde{\mathbf{X}}_{t-1}|\mathbf{Y}_{1:t-1})\, \mathrm{d}\tilde{\mathbf{X}}_{t-1}\ , \quad (1)$$

where C is a normalisation constant. As closed-form solutions are usually not available in practise, this estimation is implemented using a particle filter with a Markov Chain Monte Carlo (MCMC) sampling scheme [11]. The main elements of the model are described below.

A. State space

We use a multi-object state space formulation, with our global state defined as $\tilde{\mathbf{X}}_t = (\mathbf{X}_t, \mathbf{k}_t)$, where $\mathbf{X}_t = \{\mathbf{X}_{i,t}\}_{i=1..M}$ and $\mathbf{k}_t = \{k_{i,t}\}_{i=1..M}$. The variable $\mathbf{X}_{i,t}$ denotes the state of face i, which comprises the position, speed, scale and eccentricity (i.e. the ratio between height and width) of the face bounding box. Each $k_{i,t}$ denotes the status of face i at time t, i.e. $k_{i,t} = 1$ if the face is visible at time t, and $k_{i,t} = 0$ otherwise. Finally, M denotes the maximum number of faces visible at a current time step.

B. State Dynamics

The overall state dynamics is defined as:

$$p(\tilde{\mathbf{X}}_t|\tilde{\mathbf{X}}_{t-1}) \propto p_0(\mathbf{X}_t|\mathbf{k}_t) \prod_{i\in\{1..M\}|k_{i,t}=1} p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})\,, \quad (2)$$

that is the product of an interaction prior p_0 and of the dynamics of each individual face that is visible at iteration t like in tracking methods for a fixed number of targets [11]. Note that this is actually feasible since the creation and deletion of targets are defined outside the filtering step (see next section). The position and speed components of the visible faces are described by a mixture of a first-order auto-regressive model p_a and a uniform distribution p_u, i.e., if x denotes a position and speed component vector, we have: $p(x_{i,t}|x_{i,t-1}) = \alpha p_a(x_{i,t}|x_{i,t-1}) + (1-\alpha) p_u(x_{i,t}|x_{i,t-1})$, with $p_a(x_{i,t}|x_{i,t-1}) = \mathcal{N}(Ax_{t-1}; 0, \mathbf{\Sigma})$, and $p_u(x_{i,t}|x_{i,t-1})) = c$ with c being a constant allowing for small "jumps" coming from face detection proposals (see Eq. 8). A first order model with steady-state is used for the scale and eccentricity parameters. If x denotes one such component: $(x_t - SS) = \mathcal{N}(a(x_{t-1} - SS); 0, \sigma_{SS})$, where SS denotes the steady-state value. The steady-state values for scale and eccentricity are updated only when a detected face is associated with the face track and at a much slower pace compared to the frame-to-frame dynamics.

The interaction prior p_0 is defined as

$$p_0(\mathbf{X}_t|\mathbf{k}_t) = \prod_{\{i,j\}\in\mathcal{P}} \phi(\mathbf{X}_{i,t}, \mathbf{X}_{j,t}) \propto \exp(-\lambda_g \sum_{\{i,j\}\in\mathcal{P}} g(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})), \quad (3)$$

preventing targets to become too close to each other. The set $\mathcal{P}$ consists of all possible pairs of objects that are visible. The penalty function $g(\mathbf{X}_{i,t}, \mathbf{X}_{j,t}) = \frac{2a(B_i \cap B_j)}{a(B_i)+a(B_j)}$ is the intersection area as a fraction of the average area of the two bounding boxes B_i and B_j defined by $\mathbf{X}_{i,t}$ and $\mathbf{X}_{j,t}$, where $a(.)$ denotes the area operator. The factor λ_g controls the strength of the interaction prior (set to 5 in our experiments).

C. Observation Likelihood

As a trade-off between robustness and computational complexity, we employ a relatively simple but effective observation likelihood for tracking. Another model could be used as well.

Given our scenario, we assume that the face observations $\mathbf{Y}_{i,t}$ are conditionally independent given the state, leading to an observation likelihood defined as the product of the visible individual faces likelihoods:

$$p(\mathbf{Y}_t|\tilde{\mathbf{X}}_t) = \prod_{i|k_{i,t}=1} p(\mathbf{Y}_{i,t}|\mathbf{X}_{i,t}). \quad (4)$$

Note that we did not include a partial (or full) overlap model in the likelihood component, nor any other contextual tracking techniques [27]. Strong overlaps are prevented explicitly by the interaction term (Eq. 3) in the dynamics. This approach is appropriate for our scenarios (teleconference, HCI/HRI), where continuous partial face occlusions happen only rarely. More often, faces are occluded by other body parts that are not followed by the tracker, like a person's hand, or another person's body crossing in front. Even a joint likelihood model would not handle these cases. Thus, for longer full occlusions, our strategy is to have the algorithm remove the track of the occluded face, and restart it afterwards as soon as possible.

The observation model for a face i is based on $R = 6$ HSV colour histograms $\mathbf{Y}_{i,t} = [h(r, \mathbf{X}_{i,t})]$ $(r = 1..R)$, that are computed on the face region described by the state $\mathbf{X}_{i,t}$. They are compared to histogram models $h^*_{i,t}(r)$, to define the observation likelihood for a tracked face as follows:

$$p(\mathbf{Y}_{i,t}|\mathbf{X}_{i,t}) \propto \exp(-\lambda_D \sum_{r=1}^{6} \left(D^2[h^*_{i,t}(r), h(r, \mathbf{X}_{i,t})]\right) - D_0)\,, \quad (5)$$

where D denotes the Euclidean distance[1], $\lambda_D = 20$, and D_0 is a constant offset defining the distance at which the likelihood in Eq. (5) gives 1.0. More precisely, we divided the face into three horizontal bands and in each band computed two normalised histograms with two different levels of quantisation. Specifically, we used the scheme proposed in [2] which decouples coloured pixels (put into $N_b \times N_b$ HS bins) from grey-scale pixels (N_b separate bins) and applied it with two different quantisation levels, $N_b = 8$ and $N_b = 4$ bins per channel. This choice of semi-global multi-level histograms results from a compromise between speed, robustness to appearance variations across people as well as head pose variations for individuals, and a well conditioned likelihood, i.e. peaky enough to accept a well identified optimum, but with a smooth basin of attraction towards this optimum, adapted to low sampling strategies.

The histogram models of one face are initialised when a new target is added to the tracker. Furthermore, to improve the tracker's robustness to improper initialisation and changing

[1] A Bhattacharyya distance could have been used as well.

lighting conditions, they are updated whenever a detected face is associated with the given face track (see below):

$$h^*_{i,t}(r) = (1-\epsilon)h^*_{i,t-1}(r) + \epsilon h^d_{i,t}(r) \quad \forall r\,, \tag{6}$$

where $h^d_{i,t}$ denotes the histograms from the detected face region, and ϵ is the update factor (set to 0.2 in our experiments).

D. Tracking algorithm

At each time instant, the tracking algorithm proceeds in two main stages: first, recursively estimate the states of the currently visible faces relying on the model described above and solved using a MCMC sampling scheme. Second, make a decision on adding a new face or on deleting currently tracked faces. This second stage is described in Section III. The MCMC sampling scheme allows for efficient sampling in this high-dimensional state space of interacting targets, and follows the method described in [11].

Let N be the total number of particles and N_{bi} the number of "burn-in" particles. At each tracking iteration, we do:

1) initialise the MCMC sampler at time t with the sample $\tilde{\mathbf{X}}_t^{(0)}$ obtained by randomly selecting a particle from the set $\{\tilde{\mathbf{X}}_{t-1}^{(s)}, s = (N_{bi}+1)\dots N\}$ at time $t-1$ and sample the state of every visible target i using the dynamics $p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})$ (deleted targets are ignored);
2) sample iteratively N particles from the posterior distribution of (1) using the Metropolis-Hastings algorithm:
   a) sample a new particle $\tilde{\mathbf{X}}_t{}'$ from a proposal distribution $q(\tilde{\mathbf{X}}_t{}'|\tilde{\mathbf{X}}_t{}^{(s)})$ (described below);
   b) compute the acceptance ratio:

$$a = \min\left(1, \frac{p(\tilde{\mathbf{X}}_t{}'|\mathbf{Y}_{1:t})\, q(\tilde{\mathbf{X}}_t{}^{(s)}|\tilde{\mathbf{X}}_t{}')}{p(\tilde{\mathbf{X}}_t{}^{(s)}|\mathbf{Y}_{1:t})\, q(\tilde{\mathbf{X}}_t{}'|\tilde{\mathbf{X}}_t{}^{(s)})}\right) \tag{7}$$

   c) accept the particle (i.e. define $\tilde{\mathbf{X}}_t{}^{(s+1)} = \tilde{\mathbf{X}}_t{}'$) with probability a. Otherwise, add the old particle (i.e. set $\tilde{\mathbf{X}}_t{}^{(s+1)} = \tilde{\mathbf{X}}_t{}^{(s)}$)

After time step t, the particle set $\{\tilde{\mathbf{X}}_t^{(s)}\}_{s=N_{bi}+1}^{N}$ represents an estimation of the posterior $p(\tilde{\mathbf{X}}_t|\mathbf{Y}_{1:t})$.

The proposal function $q(\cdot)$ allows for selecting good candidates for the particle set. Efficiency in MCMC sampling is obtained by modifying object states one at a time. More precisely, a new sample is selected by letting $\tilde{\mathbf{X}}_t{}' = \tilde{\mathbf{X}}_t{}^{(s)}$, randomly select a face i amongst the visible ones, and then sample the proposed state $\mathbf{X}'_{i,t}$ of face i from:

$$q(\mathbf{X}'_{i,t}|\tilde{\mathbf{X}}_t) = \Big[(1-\alpha)\frac{1}{N-N_{bi}}\sum_r p(\mathbf{X}'_{i,t}|\mathbf{X}^{(s)}_{i,t-1}) + \alpha\, p(\mathbf{X}'_{i,t}|\mathbf{X}^d_t)\Big] \tag{8}$$

that is a mixture of the state dynamics (ensuring temporal smoothness) and the output of a face detector (avoiding tracker drift) controlled by the factor α,. where $\mathbf{X}^d_t$ denotes the state of the closest detection coming from a face detector [1] and associated with face i. Again, targets removed at the previous step are ignored, while recently added targets are simply sampled around their initial position.

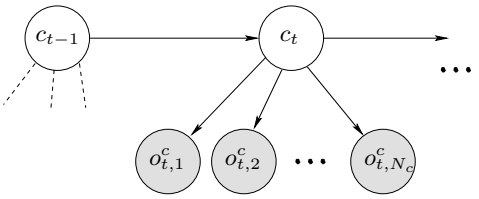


Fig. 2. The HMM used at each image position for tracker target creation. The variable c_t indicates a face centred at a particular image position. The probability of c_t is estimated recursively using the observations $o^c_{t,1}..o^c_{t,N_c}$.

III. Target creation and removal

The way objects are added and removed from the tracker is a key feature of the proposed algorithm. In our application scenario, the goal is to avoid false alarms as much as possible. This means, the tracker should be able to detect as quickly as possible if there is a tracking failure. On the other hand, it should not stop tracking when there is no failure since it may take a long time until the face is detected again.

We propose to use two different Hidden Markov Models (HMM) for that purpose, as described in the following sections. One is used for object creation and the other for object removal. Each of them receives different types of observations.

A face detector (for both frontal and profile views) is called every 10 frames (i.e. around twice per second, as our algorithm is able to process around 20-23 frames/s in real-time). The HMMs are updated only at these instants, but rely on observations computed on all frames since the last update. According to our experiments, applying the detector to every video frame did not significantly improve the tracking performance and considerably slowed down the algorithm.

Before the creation and removal step, each detection is associated to a track provided the following conditions hold:

1) the detection is not associated with any other target,
2) it has the smallest distance to the tracked target,
3) the distance between detection and target is smaller than two times the average width of their bounding boxes,
4) the two bounding boxes overlap.

Although a more generic way would be to use training data to learn the association rules and parameters as done in [28], for instance, the above conditions work well for our data in the large majority of cases.

In the following, we describe the HMMs for target creation and removal. Note that naturally, only un-associated detections are considered for the initialisation of a new target.

A. Creation

When initialising a new target we have two objectives: first, minimise erroneous initialisations due to false detections, and second, initialise correct targets as early as possible.

For deciding when to add new targets to the face tracker, we propose a simple HMM that estimates the probability of a hidden, discrete variable $c_t(i,j)$ indicating at each image position (i,j) if there is a face or not at this position. Fig. 2 illustrates the model. In the following, we drop the (i,j) indices for clarity. Let us denote by $\mathbf{O}^c_t = [o^c_{t,1}, \dots, o^c_{t,N_c}]$ the set of N_c observations at each time step t, and by $\mathbf{O}^c_{1:t} = [\mathbf{O}^c_1, \dots, \mathbf{O}^c_t]$ the sequence of observations from time

1 to time t. Assuming the transition matrix is defined as: $p(c_t|c_{t-1}) = 1$ iff $c_t = c_{t-1}$ and 0 otherwise, the posterior probability of the state c_t can be recursively estimated as:

$$p(c_t = s|\mathbf{O}^c_{1:t}) = \frac{p(\mathbf{O}^c_t|c_t = s)\, p(c_{t-1} = s|\mathbf{O}^c_{1:t-1})}{\sum_{s'} p(\mathbf{O}^c_t|c_t = s')\, p(c_{t-1} = s'|\mathbf{O}^c_{1:t-1})}, \tag{9}$$

where

$$p(\mathbf{O}^c_t|c_t) = \prod_{i=1}^{N_c} p(o^c_{t,i}|c_t)\ . \tag{10}$$

1) **Track creation:** for each detected face that is not associated with any current face target, we decide whether a track is created or not. To this end, if (i, j) denotes the centre position of the face detection, the ratio:

$$r^c_t(i,j) = \frac{p\left(c_t(i,j) = 1|\mathbf{O}^c_{1:t}(i,j)\right)}{p\left(c_t(i,j) = 0|\mathbf{O}^c_{1:t}(i,j)\right)} \tag{11}$$

is computed. If $r^c_t(i,j) > 1$, then a new track is initialised at (i, j). Otherwise, no track is created from the given detection.

2) **Observations and likelihood models:** we propose to use two different types of observations: $o^c_{\cdot,1}$, the output of the face detector and $o^c_{\cdot,2}$, a long-term "memory" of the states (i.e. positions) of tracked faces $\mathbf{X}_t$.

The first observation is defined as follows. At time t and image position (i, j) we set:

$$o^c_{t,1} = \begin{cases} 1 & \text{if } (i,j) \text{ is covered by one of the} \\ & \text{bounding boxes of the detected faces,} \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

The likelihood of the first observation is then defined as

$$\begin{aligned} &p(o^c_{t,1} = 0|c_t = 0) = 1 - fa, \quad p(o^c_{t,1} = 1|c_t = 0) = fa, \\ &p(o^c_{t,1} = 0|c_t = 1) = md, \qquad p(o^c_{t,1} = 1|c_t = 1) = 1 - md\ , \end{aligned} \tag{13}$$

where fa is the empirical false alarm rate and md the missed detection rate of the detector. According our detection results from several datasets, we set $fa = 0.0001$ and $md = 0.4$.

The second observation $o^c_{t,2}$ is based on the history of past image positions of tracked faces, which we will call "tracking memory" in the following. At each iteration of the tracker, the tracking memory is updated slowly according to the mean of the current state distribution $\bar{\mathbf{X}}_t$:

$$o^c_{t,2} = (1 - \beta)o^c_{t-1,2} + \beta I_t\ , \tag{14}$$

where $\beta = 0.001$ and

$$I_t(i,j) = \begin{cases} 1 & \text{if } (i,j) \text{ is covered by one of the} \\ & \text{bounding boxes described by } \bar{\mathbf{X}}_t, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Fig. 3 shows an example of the tracking memory during a run of the face tracker. Given the current value of β, if a region is covered during one minute the observations $o^c_{t,2}$ will reach a value of 0.5 approximately (starting from 0).

Intuitively, we would like to initialise targets more quickly in regions where a person has been "seen" previously. Thus, we model $p(o^c_{t,2}|c_t)$ with a pair of sigmoid functions:

$$p(o^c_{t,2}|c_t = 1, \Theta) = \frac{1}{\pi}\arctan(\delta_l(o^c_{t,2} - \mu_l)) + \frac{1}{2} \tag{16}$$

$$p(o^c_{t,2}|c_t = 0, \Theta) = 1 - p(o^c_{t,2}|c_t = 1)\ , \tag{17}$$

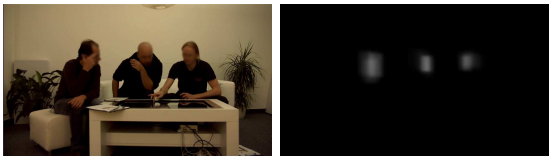


Fig. 3. Example image (left) with an illustration of the corresponding tracking memory (right) during tracking. Qualitatively speaking, track creation will be faster (almost immediate) when a new face detection is observed in the "white" regions whereas repetitive detection will be needed to initiate a track in a "black" region. Similarly, when an object track moves to black regions, its failure probability will become higher. See text for details.

where the parameters $\Theta_l = (\mu_l, \delta_l)$, denote the offset and the slope of the sigmoid (see Fig. 4). Intuitively, the offset μ_l denotes the threshold value beyond which an observation $o^c_{t,2}$ is more likely to occur within the bounding box of a detected face than in a non-face area, whereas the slope controls how fast the likelihood change is around this threshold.

3) **Parameter learning:** the parameters $\Theta_l = (\delta_l, \mu_l)$ of the sigmoid functions in equations 16 and 17 have been trained offline with a set of $N^{\pm}$ observations o_i. These observations are tracking memory values that have been collected from real tracking sequences and are composed of N^+ positive instances measured at image positions of correct face detections, and N^- negative instances measured at image positions of false detections. To train the model, we maximise the posterior probability of the labels for the given observations o:

$$\Theta^* = \underset{\Theta}{\operatorname{argmax}} \prod_{i=1}^{N^{\pm}} p(c = C_i|o_i, \Theta), \tag{18}$$

where $C_i \in \{0, 1\}$ denotes the class label of o_i, and $p(c = c_i|o_i, \Theta) \propto p(o_i|c = c_i, \Theta)$ is given by Eq. 16 and 17, and we assumed an equal prior on both classes. In practise, we find a good approximation of Θ^* by doing a grid search in a reasonable range over the parameter space Θ. Figure 4 shows an example of a pair of learnt sigmoid functions and the respective decision boundary (in this case for target removal).

If observations are greater than μ, the ratio $\frac{p(o_i|c=1)}{p(o_i|c=0)} > 1$, that means a face is more likely to be present. Otherwise, it is more likely that no face is present.

B. Removal

During tracking, we want to assess at each point in time if the algorithm is still correctly following a face or if it has lost track. The algorithm can lose track, for example, when it gets distracted by a similar background region or when a person leaves the scene. More concretely, the objective is to interrupt the tracking as soon as possible if a failure occurs, and to continue tracking otherwise, even when a face has not been detected and associated with the track for a long time.

In a way similar to target initialisation, we propose to use for each tracked face i an HMM estimating at each time step t the hidden status variable $k_{i,t}$ indicating correct tracking ($k_{i,t} = 1$) or tracking failure ($k_{i,t} = 0$). We will drop the face index i in the following. Fig. 5 illustrates the proposed model.

Let us denote by $\mathbf{O}^r_t = [o^r_{t,1}, \ldots, o^r_{t,N2}]$ the set of N_r observations at each time step t, and by $\mathbf{O}^r_{1:t} = [\mathbf{O}^r_1, \ldots, \mathbf{O}^r_t]$

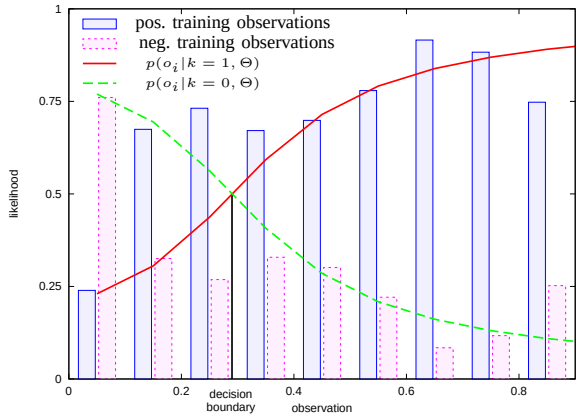


Fig. 4. Example of an observation likelihood model (here o^r_3) described by a pair of sigmoid functions with learnt parameters $\Theta = \{\delta, \mu\}$. The parameters of the positive sigmoid function (solid red curve) have been optimised to model best the positive (blue solid boxes) and negative (purple dotted boxes) training observations (here illustrated by histograms) according to Eq. 18. The offset μ (here at $x = 0.29$), where the two sigmoid paired curves cross, defines a soft decision boundary. More precisely, in this example, an observation above this threshold favours target removal, as $p(o_i|k=1) > p(o_i|k=0)$, and vice versa. The parameter δ controls the slope of the functions, and thus the strength and (un)certainty of the decision. In this case, the slope is not very steep, reflecting the fact that small observations can be observed even when track is not lost.

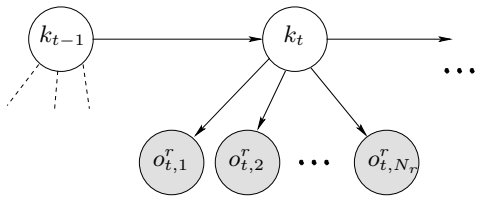


Fig. 5. The HMM for target removal, used for each tracked face. The variable k_t indicates if a given face is still tracked correctly or if a failure occurred. The probability of k_t is estimated recursively using the observations $o^r_{t,1}..o^r_{t,N_r}$.

the sequence of observations from time 1 to time t. The posterior probability of k_t can be recursively estimated as:

$$p(k_t|\mathbf{O}^r_{1:t}) = \frac{\sum_{k'_{t-1}} p(\mathbf{O}^r_t|k_t)\, p(k_t|k'_{t-1})\, p(k'_{t-1}|\mathbf{O}^r_{1:t-1})}{\sum_{k'_t,k'_{t-1}} p(\mathbf{O}^r_t|k'_t)\, p(k'_t|k'_{t-1})\, p(k'_{t-1}|\mathbf{O}^r_{1:t-1})}, \tag{19}$$

where

$$p(\mathbf{O}^r_t|k_t) = \prod_{i=1}^{N_r} p(o^r_{t,i}|k_t)\ . \tag{20}$$

The state transition probability $p(k_t|k_{t-1})$ is set to 0.999 for staying in the same state and 0.001 for changing state, assuming a frame rate of approximately 20 frames per second as in our experiments.

1) **Track ending:** for each tracked face and at each time step, we compute the ratio:

$$r^k_t = \frac{p(k_t = 1|\mathbf{O}^r_{1:t})}{p(k_t = 0|\mathbf{O}^r_{1:t})}\ . \tag{21}$$

If $r^k_t < 1$ for a given face, then the tracking is considered to have failed and the target is removed.

2) **Observations and likelihood models:** we propose to use $N_r = 7$ different types of observations $\mathbf{O}^r_t = [o^r_{t,1}, \ldots, o^r_{t,7}]$ extracted from the image as well as the state of the tracker itself. We can divide them into two categories:

- four *static* observations $(o^r_{t,1}, \ldots, o^r_{t,4})$ that provide indications on the *state of the tracker*, and

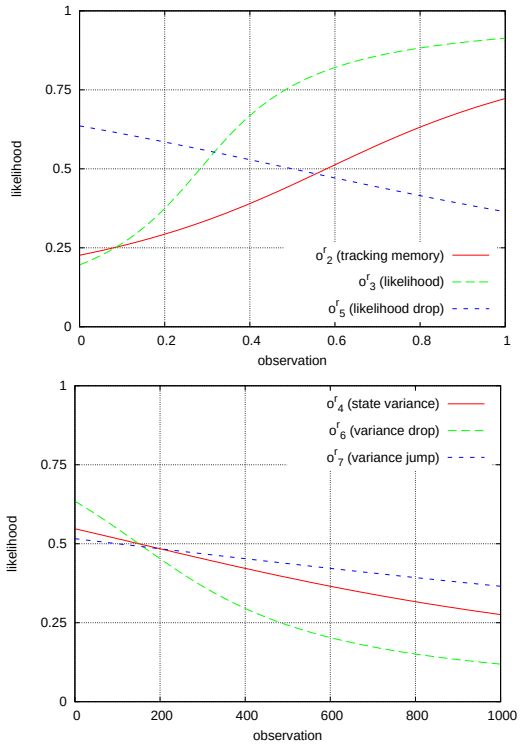


Fig. 6. Trained likelihood functions $p(o^r_2|k_t = 1)$ to $p(o^r_7|k_t = 1)$. The soft decision boundary for a given observation type is the x-value where the curve crosses the line $y = 0.5$. A steeper curve qualitatively means that the corresponding observation is more discriminative for the decision to remove a target or not (e.g. for the likelihood and variance drop observations). Note that although the tracking memory slope is not as steep, it nonetheless plays an important role since spatially, a tracker can quickly move from a position where the observation is greater than 0.58 (i.e. above the soft boundary), that is, a position in a region where tracks have been observed recently, to an image positions where the observation is very close to 0.

- three *dynamic* observations $(o^r_{t,5}, \ldots, o^r_{t,7})$ that provide indications on the *temporal evolution* and variability of certain observations.

Except for one observation, all likelihoods are modelled by pairs of sigmoid functions:

$$p(o^r_{t,i}|k_t = 1, \Theta) = a_i\ \arctan(\delta_i(o^r_{t,i} - \mu_i)) + \frac{1}{2}, \tag{22}$$

$$p(o^r_{t,i}|k_t = 0, \Theta) = 1 - p(o^r_{t,i}|k_t = 1), \tag{23}$$

where, as for target creation observations (section III-A), the amplitude a_i is set to $\frac{1}{\pi}$ (or $-\frac{1}{\pi}$ for some observation types), and the parameters $\Theta_i = (\delta_i, \mu_i)$, i.e. the slope and the offset of the sigmoid, have been trained offline with a set of positive and negative observations as described at the end of Section III-A. The only difference is that the training observations are collected at each time instant during tracking runs and not only when faces are detected. Figure 6 shows the plots of the trained functions for $k = 1$. Below, we describe each observation we have used and comment on the learnt parameters.

Static observations: the first static observation for a given target is based on the output of the face detector:

$$o^r_{t,1} = \begin{cases} 1 & \text{if a detection is associated with the target} \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

The likelihood $p(o^r_{t,1}|k_t)$ is defined in the same way as for $o^c_{t,1}$ in Eq. 13 (that is, $p(o^r_{t,1} = u|k_t = l) = p(o^c_{t,1} = u|c_t = l)$).

The second observation $o^r_{t,2}$ is the tracking memory value at the respective target position (m, n) in the image, as defined in the previous section (Eq. 14 and 15):

$$o^r_{t,2} = o^c_{t,2}(m, n) \ . \tag{25}$$

This ensures that the tracking of a face is more likely to be maintained if a face stays at its previous position (with a high tracking memory value). And conversely, the target should be removed with a higher probability when it moves to image regions that were never occupied by a face before. From the slope of the sigmoid function corresponding to $p(o^r_2|k_t = 1)$ shown in Fig. 6, we can see that these observations provide a relatively discriminant information regarding k_t.

The third observation type is the tracker observation likelihood computed at the mean state value $\bar{\mathbf{X}}_{i,t}$ of target i:

$$o^r_{t,3} = p(\mathbf{Y}_{i,t}|\bar{\mathbf{X}}_{i,t}) \ , \tag{26}$$

as defined by Eq. 5. The likelihood $p(o^r_{t,3}|k_t)$ is again defined by a pair of sigmoids (Eq. 22 and 23). Figure 6 shows that these observations are highly discriminant with respect to k_t, in the sense that a tracker likelihood value below approximately 0.3 is characteristic for a tracking failure.

The fourth observation relates to the variance of the target filtering distribution. More precisely, let $\sigma^2_{i,t,x}$ and $\sigma^2_{i,t,y}$ be the variances of the horizontal and vertical position of target state $\mathbf{X}_{i,t}$. Then we define

$$o^r_{t,4} = \max(\sigma^2_{i,t,x}, \sigma^2_{i,t,y}). \tag{27}$$

A higher variance of the state distribution means a higher uncertainty (and vice versa), and the track should be stopped more quickly. The rather flat function in Fig. 6 shows that these observations are less discriminant on their own.

Dynamic observations: the three remaining observations are based on the temporal variation of different features. They rely on the detection of rapid increases or decreases over time of particle variance and observation likelihood. To this end, we assume that the values of these features are normally distributed during tracking, and we use the Page-Hinckley test [29] to detect jumps or drops of these (one-dimensional, Gaussian) "signals" with respect to their means. This test works as follows: let ω_t be the signal for which we want to detect an abrupt *decrease*. Then, the following values are computed at each iteration t:

$$M_{\omega,t} = M_{\omega,t-1} + \left(\omega_t - (\bar{\omega}_t - \frac{j_\omega}{2})\right) \tag{28}$$

$$m_{\omega,t} = \max(m_{\omega,t-1}, M_{\omega,t}) \tag{29}$$

$$\hat{m}_{\omega,t} = m_{\omega,t} - M_{\omega,t} \ , \tag{30}$$

where $M_{\omega,0} = 0$, j_ω is a constant that determines the tolerated change of value ω, and $\bar{\omega}_t$ is the running average of ω. $M_{\omega,t}$ accumulates the values going above the expected lower bound $(\bar{\omega} - j_\omega)$. The value $m_{\omega,t}$ memorises the maximum value of this cumulative sum, and the difference between these last two values $\hat{m}_{\omega,t}$ (Eq. 30) is an indication of an abrupt decrease of the value ω. On the other hand, if ω_t decreases only *gradually*, then the running average $\bar{\omega}_t$ will follow this decrease. The cumulative sum $M_{\omega,t}$ will constantly increase,

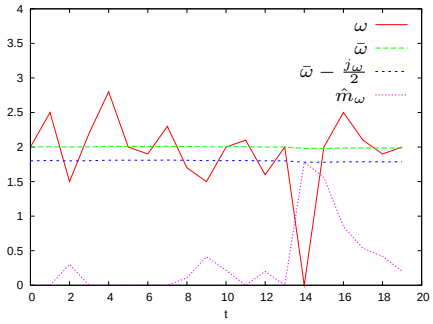


Fig. 7. Illustration of the Page-Hinckley test to detect abrupt decreases of a signal. The solid red line shows the temporal evolution of some signal, the purple dotted line shows the computed result of the Page-Hinckley test that we use as observation to detect abrupt signal drops. At time $t = 14$ a drop of the signal occurs and is correctly detected (peak of purple dotted line).

leading to $m_{\omega,t} = M_{\omega,t}$ and thus $\hat{m}_{\omega,t} = 0$. Figure 7 illustrates the Page-Hinckley test with some example data. At $t = 14$ a signal drop occurs leading to a high value of $\hat{m}_\omega$.

Similarly, for detecting an abrupt *increase* of ω we compute:

$$U_{\omega,t} = U_{\omega,t-1} + \left(\omega_t - (\bar{\omega}_t + \frac{j_\omega}{2})\right) \tag{31}$$

$$u_{\omega,t} = \min(u_{\omega,t-1}, U_{\omega,t}) \tag{32}$$

$$\hat{u}_{\omega,t} = U_{\omega,t} - u_{\omega,t} \ , \tag{33}$$

where $U_{\omega,0} = 0$. In its original form, the Page-Hinckley test produces a binary output. It is one if $\hat{m}_{\omega,t}$ or $\hat{u}_{\omega,t}$ is above a predefined threshold and zero otherwise. Here, we propose to directly use the values $\hat{m}_{\omega,t}$ or $\hat{u}_{\omega,t}$ as observations.

Thus, using equations 28-33, we define:

$$o^r_{t,5} = \hat{m}_{o^r_t,3} \qquad o^r_{t,6} = \hat{m}_{o^r_t,4} \qquad o^r_{t,7} = \hat{u}_{o^r_t,4} \ . \tag{34}$$

Observations $o^r_{t,5}$ indicate drops of the likelihood $p(\mathbf{Y}_t|\bar{\mathbf{X}}_t)$ of a given face (see 26). And $o^r_{t,6}$, $o^r_{t,7}$ indicate abrupt decreases and increases of the variance of the state distribution defined in 27. The likelihood functions $p(o^r_{t,5}|k_t), p(o^r_{t,6}|k_t)$, and $p(o^r_{t,7}|k_t)$ are defined by pairs of sigmoids (Eq. 22, 23) with parameters trained offline. The resulting sigmoids for $o^r_{t,5}$ and $o^r_{t,6}$ are relatively flat, thus not so discriminant on their own (see Fig. 6). The observation $o^r_{t,7}$, that is the rapid increase in position variance, is more discriminant.

IV. Person Identification

The algorithm further tries to keep track of the identities of different persons and associates each track with a person, i.e. for each new target track it decides if it belongs to a previously seen person or if it is an new person. In this work, we built person models which are longer-term descriptions of person appearance acquired from observations during the tracking process. Here, a simple colour-based model, similar to [20] has been used. More specifically, the model $P_{j,t}$ of a person j is composed of two colour histograms: one describing the face region, $h^f_{j,t}$, and one for the shirt, $h^s_{j,t}$. The structure of the histograms is similar to the one used for the observation likelihood in the tracking algorithm (II-C), i.e. two different quantisation levels and decoupled colour and grey-scale bins.

If a target is added to the tracker and there is no stored person model that is un-associated, a new model is initialised

immediately and associated to the target. Otherwise, the face and shirt histograms ($h^f_{i,t}$, $h^s_{i,t}$) of the new target i are computed recursively over r successive frames and stored in $P^*_{i,t}$. After this period, we calculate the likelihood of each stored model $P_{j,t}$ given an unidentified candidate $P^*_{i,t}$:

$$p(P_{j,t}|P^*_{i,t}) = \exp\left(-\lambda(w_f D^2[h^f_{j,t}, h^f_{i,t}] + w_s D^2[h^s_{j,t}, h^s_{i,t}])\right), \quad (35)$$

where D is the Euclidean distance, and the weights are $w_f = 1, w_s = 2$. A given person i is then identified by simply determining the model $P_{m,t}$ with the maximum likelihood:

$$m = \underset{j}{\operatorname{argmax}}\, p(P_{j,t}|P^*_{i,t}) \ , \quad (36)$$

provided that $p(P_{m,t}|P^*_{i,t})$ is above a threshold θ (we chose 0.1 here). If not, a new person model is created and added to the stored list. All associated person models are updated at each iteration with a small factor $\alpha^p = 0.01$. The candidate models are updated with factor $\alpha^* = 0.1$.

V. Experimental protocol

A. Databases

Approach. Many tracking algorithms are tested on short sequences (maximum one to two minutes) [7], [8], [9], [10] where the creation/removal problem is not addressed and therefore their test sequences are not appropriate for our purpose. Further, these sequences are often specifically made up (e.g. moving a puppet or a coke can attached with a wire) [7], [10] and do not correspond to any application scenario.

In this paper, we adopt another approach, and ground our experiments on a real use-case, with long recordings involving natural individual and interactive behaviours, and without people caring about the camera or the tracking task. Although the data may not look particularly challenging, it is indeed full of cases that can affect tracking due to the variety of situations, of poses, of pose changes and dynamics, quick body shifts, partial occlusions by hands (when people touch their face or their hair, drink or eat), full occlusion by others, or combinations of the above. In addition, *long* sequences can be challenging for algorithm using adaptation components (like our colour histogram models). Indeed, the difficulty with adaptation does not always lie in the most dynamic situations (where the objects can be distinguished from the background), but also in other quiet and apparently simple moments, where partial drift in the representation can actually occur and ultimately lead to failure.

Data. Experiments have been conducted on more than 9 hours of video data that have been annotated extensively. We used three sets of videos recorded in different environments (see Fig. 1). According to our scenario, recorded people have been sitting at a table and filmed by a central camera (roughly 2-3 metres away). They have been playing online games with people in a remote location using a laptop or touch-screen. As a result, they are often looking downwards and their faces are often not detected by a standard detector [1].

Figure 1 shows example images from the three datasets. In dataset 1, the lighting conditions are overall good. In dataset 2, the overall complexity of the videos is higher because of more difficult lighting conditions (e.g. cross shadow between people), it contains more people including children, so the scene is more dynamic. Also, occlusions occur more frequently. The videos of dataset 3 are rather challenging for face tracking as people sit close to each other[2]. Also, the lighting condition and image quality is worse in the second video of this set. Finally, the number of visible persons is varying throughout the videos due to people leaving temporarily the scene.

B. Annotation

Almost 22 000 videos frames have been annotated. The time spacing between two annotated video frames δ_t varies from $0.04s$ to $12s$ according to the dynamics of the scene. That is, δ_t is smaller for periods with large movement of the persons and vice versa. In total, there are more than 60 000 head position annotations, together with an identifier, i.e. a number that is assigned to each person. The position and size of a head is described by a bounding box.

C. Evaluation protocol

Since the paper contribution is mainly about track failure detection (and track creation), one could for instance imagine to label all failures of a tracking algorithm and evaluate whether the failure detection algorithm performs well or not. However, since each tracker may fail at different moments, for different tracks, depending on history (for instance if a track was removed or not has impact on the current model due for instance to the colour model adaptation), or on parameters, such an approach is not feasible in practise. We thus evaluate our approach in terms of tracking performance, which is our final task of interest. To this end, the principal performance measures are precision and recall (over time) of the face tracking result, as we want to track faces as long as possible (to obtain a high recall) and stop tracking as soon as a failure occurs (to increase the precision). In the following, we describe the used performance measures, the different algorithms that we compared, and their parameters.

1) **Performance measures:** in a given video frame, first, every tracked face (or face detection) is associated with a ground truth face from the annotation. The association rules are described in section III. A face detection or tracker output is counted as correct if the F-measure with the ground truth is greater than 0.1. The F-measure is defined as:

$$F = \frac{2a(B_i \cap B_j)}{a(B_i) + a(B_j)} \ , \quad (37)$$

where B_i is the ground truth rectangle (i.e. a bounding box of the entire head) and B_j is the rectangle output from face detection or tracking. In other words, the F-measure is the ratio between the intersection and the average area of the two rectangles.

We further define the recall and false positive rate for an entire video as:

$$R = \frac{\sum_{i=2}^{G} \delta_i d_i}{\sum_{i=2}^{G} \delta_i} \ , \qquad FP = \frac{\sum_{i=2}^{G} \delta_i f_i}{\sum_{i=2}^{G} \delta_i} \ , \quad (38)$$

[2]Dataset 3 is available at http://www.idiap.ch/dataset/ta2

where G is the number of annotated frames, d_i the proportion of correctly tracked/detected faces in frame i (i.e. those for which $F > 0.1$), f_i is the number of false positive outputs divided by the number of ground truth objects in frame i, and δ_i is the time difference between frame i and $i-1$.

We also measure the total number of interruptions for a given dataset. An interruption is defined as the event when a track is falsely ended, i.e. the face (ground truth) is still present but the respective target is removed from the tracker.

Finally, to measure the accuracy of identification as described section IV, we computed the *object purity* [30] for each ground truth object:

$$OP = \frac{\sum_{i=2}^{G} \delta_i q_i}{\sum_{i=2}^{G} \delta_i} , \tag{39}$$

where G is again the number of annotated frames, and q_i is the proportion of correctly identified faces in frame i, as explained in the following. The identity assigned to a ground truth object at time i is given by the algorithm described in IV and more specifically Eq. 36. Once the tracking has been run on a complete video, we can compute the above rate by associating to each object the face track that has the longest overlap with the object (according to the F-measure).

2) **Algorithms:** to evaluate our approach, we compared our results against a standard face detector [1] including models for frontal and profile views, with two competitive baselines (RJ-MCMC and MCMC baseline), and also conducted experiments by switching off different parts of the model to evaluate the benefit of the different approach components. More precisely, the algorithms are as follows:

- *RJ-MCMC*: a tracker based on the Reversible-Jump Monte Carlo Markov Chain algorithm [11], [12]. In addition to the *Update* move, which follows the MCMC description in section II-D, four other moves have been implemented to handle the creation and removal of targets: *Add, Delete, Stay*, and *Leave*. For more details, we refer the reader to [11], [12].
- *MCMC baseline:* an MCMC-based tracker, i.e. the algorithm described in section II. For target creation and removal, the following strategy has been used: every (unassociated) face detection is initialised as a new target. We also tried to initialise a target only after *several* successive detections but this didn't have a large impact on the precision. A tracked target was removed if it had no associated detections for 100 frames (8 seconds) or if the likelihood dropped below 10% of its running average.
- *MCMC/HMM1:* the proposed MCMC tracker with the HMM for target creation (see section III-A). Target removal has been done as for the baseline.
- *MCMC/HMM1+2a:* the proposed MCMC tracker with HMMs for target creation and removal (see sections III-A and III-B). Observations o_1^r, o_3^r, and o_4^r have been used, that is the ones based on face detections, on the likelihood, and on the state distribution variance.
- *MCMC/HMM1+2b:* like "MCMC/HMM1+2a" but using the additional observations o_2^r, i.e. the tracking memory. That means this algorithm uses all the *static* observations.
- *MCMC/HMM1+2c:* the proposed MCMC tracker with HMMs for target creation and removal using all the observations: *static* and *dynamic*.

3) **Parameters:** all the trackers use 500 particles with a burn-in proportion of 25%. For efficiency, the videos are processed at a resolution of 640×360 pixels, and the original frame rate has been changed to 12.5 fps. The face detector threshold has been varied from 2 to 4 to obtain the different precision-recall curves. The value of the other parameters has been mentioned directly in the text. In practise, the algorithm did not exhibit a high sensitivity to changing their setting.

VI. Results

A. Qualitative results.

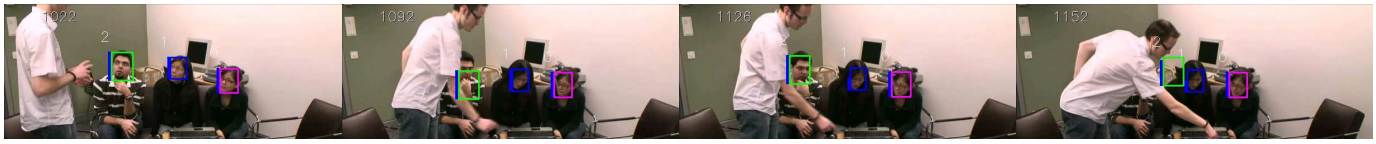
Our first experiment illustrates the proposed track creation and removal components. Figure 8 shows some snapshots of the results. The different colours of the rectangles represent different identities. In frame 1152 (Fig. 8(d)), the person seated on the left starts to be occluded for the first time. After several partial and then full occlusion moments, the object is finally removed at time 1284. The track is re-initialised in frame 1404, shortly after the face re-appears, and again removed in frame 1684 (Fig. 8(h)) after another longer occlusion.

Figure 9 shows for this video segment (of Fig. 8) the evolution of the track removal observation likelihoods $p(o_{t,i}^r|k_t)$ and posterior probability $p(k_t = 1|\mathbf{O}_t^r)$ of keeping the track of the left sitting person. The graph gives some intuition on how the proposed algorithm works and why. Absence of detections, a low tracking memory or likelihood value as well as rapid changes in variance are important cues for detecting potential tracking failures. However, it is the fusion over time of the contributions of all cues that allows to make reliable decisions.

The tracking results in Fig. 10 illustrate another situation and the role of the interaction prior p_0 (Eq. 3) that prevents strong overlap between trackers. In this case, one face is being occluded by another one. Both models compete for the observations, but the occluded head being visually less likely is kept on the side thanks to the spatial prior, which definitively generates a strong drop in likelihood. Full occlusion is thus implicitly handled by our track creation/removal algorithm, stopping the occluded, i.e. less confident, track, and reinitialising it later on when obtaining a new detection.

B. Quantitative multi-object tracking results

In the second set of experiments, we measure the performance of the proposed tracking algorithm while varying different tracker components. In Fig. 11, we plot the recall and false positive rates for the tested algorithms with a varying face detector threshold (from 2 to 4). The performance using both creation and removal HMMs is higher than for algorithm MCMC/HMM1, only using the HMM for target creation and relying on face detections and likelihood drops to assess tracking failure. Comparing MCMC/HMM1+2b (orange lines) with MCMC/HMM1+2a (dotted magenta lines) shows that adding the tracking memory observation o_2^r clearly increases the recall. Finally, algorithm MCMC/HMM1+2c (black solid



(a) frame 1022 (b) frame 1092 (c) frame 1126 (d) frame 1152

(e) frame 1274 (f) frame 1488 (g) frame 1616 (h) frame 1684

Fig. 8. Tracking result on a video with several occlusions and tracking failures (left sitting person). Different coloured rectangles represent different identities.

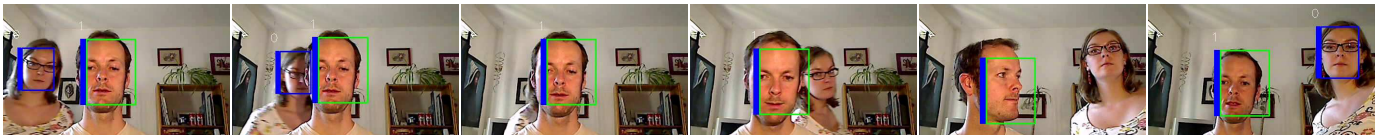


Fig. 10. Snapshots of a short video showing one face being occluded by another one. The girl's track (blue rectangle) is correctly removed by detecting a failure and reinitialised automatically afterwards. Note that identities, represented by rectangles of different colours, are kept consistent.

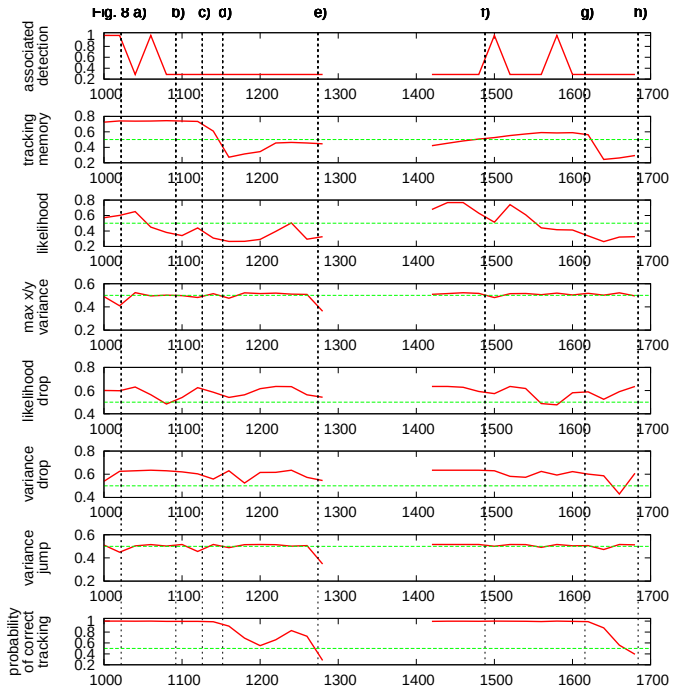


Fig. 9. Evolution of track removal likelihoods $p(o^r_{t,i}|k_t = 1)$ and the posterior probability $p(k_t = 1|\mathbf{O}^r_t)$ of correct tracking for the person sitting left in the short sequence illustrated in Fig. 8. Vertical dashed lines a) to h) correspond to the frames in Fig. 8. A likelihood value above 0.5 (green dashed line) favours keeping the track, and vice-versa. The track is actually removed if the final posterior probability (last row) is below 0.5. Initially, in (a), tracking is correct. At (b), the progressive occlusion results in a decrease of likelihood, but the face becomes visible again (c). In (d), the stronger occlusion pushes the track off the recent tracked face area, as captured by the low tracking memory, and results again in a low tracker likelihood, which jointly almost triggers the failure detection. It is ultimately the continuous absence of face detections and the sudden variance jump in (e) that allow to correctly identify the failure and remove the track. The tracker is correctly reinitialised around frame 1415. Then, a transient occlusion occurs (f) and later a new occlusion generating a low tracking memory and likelihood as well as a rapid state variance drop (usually due to only few particles having a high weight) leading to the correct stop of the track at time 1684.

lines) that incorporates the dynamic observations detecting jump/drop in likelihood or position variance results in further performance improvement in most of the cases. Comparing the results for the three datasets, one can notice that the recall of dataset 2 is lower than for the other sets, probably due to a lower recall of the face detector (see Fig. 12).

We further compared our method (MCMC/HMM1+2c) with other tracking approaches: MCMC baseline and RJ-MCMC [11]. The results are shown in Fig. 12, together with the results of the face detector, as a reference.

Clearly, for low detection thresholds the false positive (FP) rate of the face detector is much too high for many practical applications. For higher thresholds, the detector misses a lot of faces. We can see that for an acceptable FP rate (< 0.1) the recall is rather low (between 0.4 and 0.7). The dashed green lines show the results of the baseline tracker. Although it does not use the HMMs for target creation and removal it already achieves a good performance.

The performance of the proposed algorithm is clearly better than with the RJ-MCMC and the MCMC baseline systems. Since the tracking algorithm for the MCMC baseline is the same as for the proposed method, the performance improvement is clearly due to the target creation and removal mechanisms. The precision of RJ-MCMC is rather low because the creation and removal of targets is only based on the observation likelihood, as in [11]. Note that, unlike our approach, RJ-MCMC adds and removes targets at the particle level. Although this is a principled statistical framework that models at each point in time the current belief on the number of visible targets, it is more difficult to capture longer-term dynamics and features from the state distribution itself. The MCMC baseline (green dashed lines), on the other hand, adds and removes targets based on more efficient, longer-term observations, namely the likelihood with respect to its mean and the face detector output. Thus, its performance is better than the one of RJ-MCMC.

Table I compares the algorithms for a given face detector

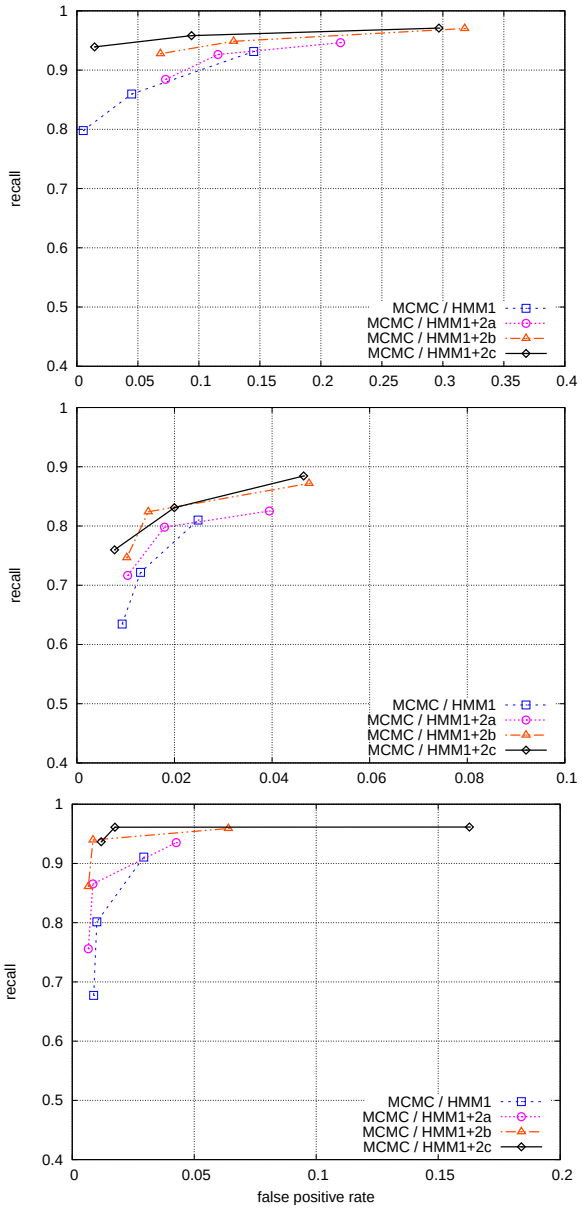


Fig. 11. False positive rate and recall for Datasets 1 to 3 using different observation types

threshold. The proposed method outperforms the others for all three datasets. Also, the total number of tracker interruptions is decreased. This means that the proposed method maintains face tracks longer, even when the face detector provides no output for extended periods of time or when the likelihood is temporarily decreasing.

Figure 13 shows some tracking results of a video from dataset 3 containing 3-4 persons. The people change their seats from time to time, occlusions occur, and head poses can be challenging, as illustrated in the example.

Finally, the average object purity (OP) over all three datasets is also higher with the proposed method, although the identification algorithm is the same for all three compared methods. We presume that this is because targets are generally tracked for a longer time with fewer interruptions. So, in a video, there are fewer re-identifications and thus potential mis-identifications. The main errors are due to people wearing

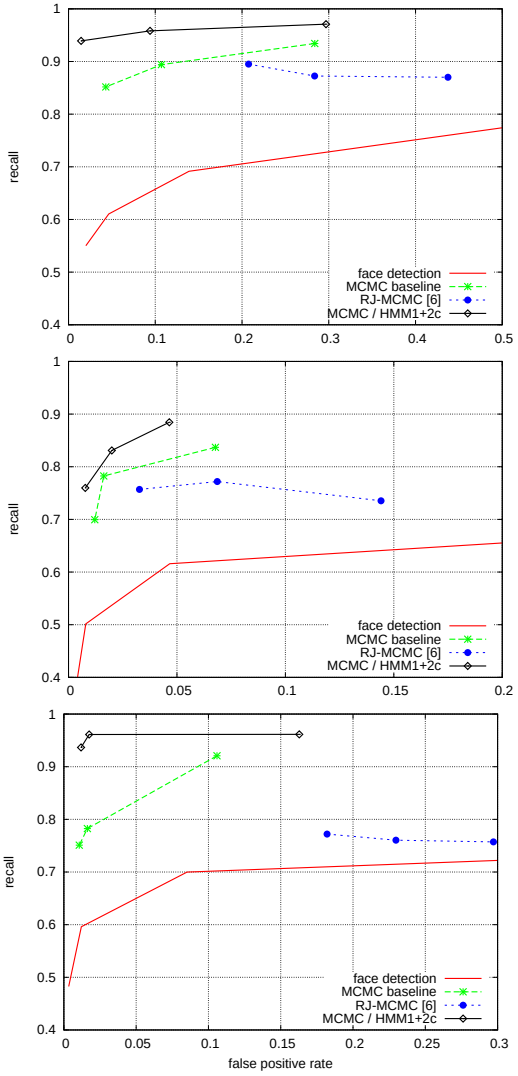


Fig. 12. Performance of different tracking algorithms for datasets 1 (top), 2 (middle), and 3 (bottom) for different face detector thresholds.

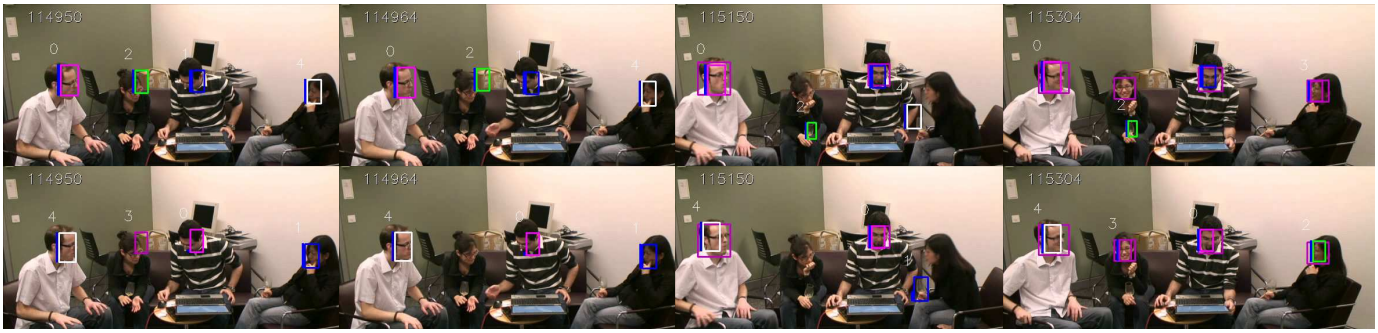
clothes with similar colours.

C. Comparison with state-of-the-art single-object trackers

We also compared our approach with other state-of-the-art algorithms that rely on different appearance modelling strategies. We focus on the longer-term tracking behaviour of these algorithms because previous works [8], [7], [9], [10], [31] do mainly evaluations on videos shorter than one minute. Since available codes are working only for single object tracking, we redesigned our experiments accordingly for this task: for a given person, all trackers were initialised manually in the first frame. Then, each time a failure was identified, the tracker was stopped and re-initialised using the first detection generated by the face detector [1] for the tracked person (called every 10 frames as in all experiments) after the failure instant. Tested algorithms were:

- *FragTrack*: “Fragment-based tracking” [8]. According to the method, a failure is detected if the objective function score is below a certain threshold (varied in different runs between 0.02 and 0.2).

(a) frame 20000 (b) frame 20038 (c) frame 52408 (d) frame 114870



(e) frame 114948 (f) frame 114962 (g) frame 115148 (h) frame 115302

Fig. 13. Snapshots of multiple-object tracking result on dataset 3. For each frame: different coloured rectangles represent different identities. Purple rectangles show the output of the face detector. *Top:* MCMC baseline, *bottom:* proposed approach. With the baseline method, some target are initialised from false detections 13(b), 13(g), and tracks are not maintained when detections are missing 13(c). The proposed approach avoids false initialisations and maintains good tracks longer. In 13(f) tracking failures are detected earlier, and in 13(h), the lost target is re-initialised earlier (second person from the left).

data set		face detection	RJ-MCMC	MCMC baseline	MCMC HMM1+2c
1	recall	55.0%	89.5%	85.2%	93.9%
	FP rate	2.00%	20.75%	4.29%	1.45%
	# interrupt.	–	861	395	112
	average OP	–	41.35%	68.69%	68.98%
2	recall	39.9%	75.7%	69.9%	76.0%
	FP rate	0.41%	3.27%	1.21%	0.77%
	# interrupt.	–	2062	1004	567
	average OP	–	49.09%	66.61%	69.60%
3	recall	48.3%	77.2%	75.1%	93.7%
	FP rate	0.33%	18.2%	1.06%	1.19%
	# interrupt.	–	1299	455	166
	average OP	–	27.96%	34.23%	57.46%

TABLE I
PERFORMANCE COMPARISON ON THE THREE DATASETS (WITH A FIXED FACE DETECTOR THRESHOLD OF 4).

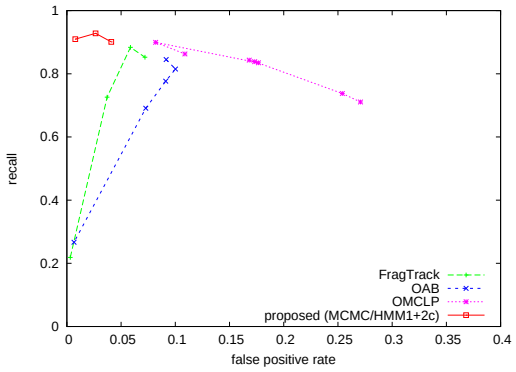


Fig. 14. Comparison with several state-of-the art tracking algorithms.

- *OAB*: "Online Adaboost" [7]. Here, the tracking confidence measure proposed by the authors for failure detection is the number of (internal) detections, that we thresholded with values between 0 and 30.
- *OMCLP*: "Online Multi-Class LPBoost" [10]. As with OAB, a threshold between 0 and 10 on the number of detections was used as failure indication.

As for our approach, we simply used our algorithm (MCMC/HMM1+2c) allowing *one* face track at most.

Data: we used 20 video clips of 5 minutes each (a random subset of our full dataset). As the videos contain several persons, we filtered out detections of the persons that are not supposed to be tracked. As a side effect, most false detections get filtered out as well, so the advantage from our target *creation* algorithm is not evident here, and the difference in performance of our approach can be mostly attributed to the target *removal* (failure detection) algorithm.

Figure 14 shows the resulting curves. Our algorithm achieves the best performance in terms of precision and recall, even if the tested state-of-the-art trackers have a more advanced appearance model than the proposed approach. Figure 15 illustrates the results on one of the videos (only a subregion of the whole video is shown). For each tracker, the best threshold for detecting lost tracks is selected according to the results shown in Fig. 14.

Overall, these results show that (i) even these trackers fail on our seemingly easy scenario, due to pose changes, quick shifts, partial occlusions by hands, etc; and (ii) the question of

when to start and stop tracking is pertinent as well for these trackers and becomes especially relevant with longer videos.

D. Discussion

We have shown that precision and recall of long-term tracking is considerably increased by using the proposed approach for target creation and removal. Our tracker outperforms also state-of-the-art single-object and multi-object tracking algorithms, despite the fact that some of them use more discriminative (multi-modal) appearance models. Indeed, the robustness of our tracking algorithm could probably be improved using these appearance models (provided they work in real time for multiple faces). Also, existing algorithms could largely benefit from our track creation and removal framework, when applied on a longer time scale.

Other scenarios. Recently, we successfully tested our algorithm on video sequences acquired by a humanoid robot Nao in a Human-Robot Interaction scenario involving several people interacting with the robot. In this setting, the video also contains fast camera motion due to the robot rotating his head to address different persons. Although this is an adverse situation (since people's position is changing quickly in the image domain) for the memory tracking cue, successful tracking and track management is achieved.

Furthermore, in this scenario, the track removal component was exploited with another tracking algorithm that does joint head position and pose tracking using a different appearance model (multi-level HOG [32]). The single main modification was to change and relearn the decision threshold (μ) of the sigmoid associated with the new tracker likelihood o_3^r using a small training dataset. Results showed that that the parameters learnt from real data were not so sensitive to different environments.

Other failure informative cues. We currently use seven observations for failure detection. Others could be used as well, but may not always lead to better results. For instance we experimented with other long-term observations based on change detection in the tracker area, with the expectation that it could detect trackers initialised or lost in the non-changing background. Results showed, however, that it is not a discriminative cue, as faces can be static for a long time, and failed tracks more often end up on other (moving) body parts than in the background.

Approach limitations and future work. The approach has several limitations. First, although being surprisingly robust overall, we believe that our base tracker could be improved using additional appearance cues (e.g. local parts). Reducing the raw number of failures would ultimately lead to better performance. Secondly, due to the temporal filtering of the HMM accumulating observations over longer periods of time, failure decisions can be slightly "delayed" (by 1-3 seconds in general). By using other cues based on shape or correlation between tracked image regions, we could expect the method to obtain a better tracking status diagnostic and to take decisions quicker with equal or better reliability. This could reduce the impact of erroneous face detections, which can lead to false track initialisations if they occur consistently and repeatedly, or which can also cause late removal of lost tracks. The discriminative training of a classifier jointly taking into account the different observations could also help in this regard. Finally, even if we are able to accurately and quickly detect failures, the result is still dependent on the time it takes to obtain a new face detection to reinitialise the tracking. Thus, having less tracking failures and better multi-view detectors is obviously a way to increase the performance.

Computational time. The proposed algorithm runs in real-time, i.e. around 20-23 frames/s (including video decoding) at a resolution of 640×360 pixels on an Intel PC at 3.16 GHz. Around 9% of the processing is spent on frame decoding and conversion, 27% on face detection (although run only once every 10 processed frames), 39% on tracker likelihood computation, 9% on the MCMC sampling steps, 7% on target creation (because it is pixel-based but this could be significantly reduced by processing grid-points only), and less than 1% on target removal.

VII. Conclusions

We presented an on-line multi-face tracking algorithm that effectively deals with situations where detections are rare or uncertain. To achieve this, long-term observations from the image and the tracker itself are collected and processed in a principled way using two separate HMMs, deciding on when to *add* and *remove* a target to the tracker.

We evaluated our approach on more than 9 hours of videos with extensive annotation, and the results show that the proposed algorithm increases the performance considerably with respect to state-of-the-art tracking methods not using long-term observations and HMMs.

References

[1] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. of CVPR*, 2001.

[2] P. Perez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proc. of ECCV*, vol. 1, Copenhagen, May–June 2002, pp. 661–675.

[3] Y. Wu, T. Yu, and G. Hua, "Tracking appearances with occlusions," in *Proc. of CVPR*, vol. 1, 2003, pp. 789–795.

[4] E. Maggio, M. Taj, and A. Cavallaro, "Efficient multi-target visual tracking using random finite sets," *IEEE on Circuits and Systems for Video Technology*, vol. 18, no. 8, pp. 1016–1027, 2008.

[5] S. M. Bhandarkar and X. Luo, "Integrated detection and tracking of multiple faces using particle filtering and optical flow-based elastic matching," *CVIU*, vol. 113, pp. 708–725, 2009.

[6] J. Fan, W. Xu, Y. Wu, and Y. Gong, "Human tracking using convolutional neural networks," *IEEE Trans. on Neural Networks*, vol. 21, no. 10, pp. 1610–1623, 2010.

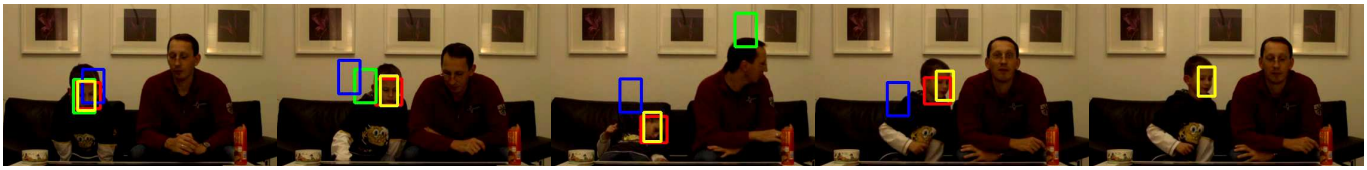
[7] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. of CVPR*, Jun. 2006, pp. 260–267.

[8] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. of CVPR*, Jun. 2006, pp. 798–805.

[9] B. Babenko, M. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. of CVPR*, Jun. 2009, pp. 983–990.

[10] A. Saffari, M. Godec, T. Pock, C. Leistner, and H. Bischof, "Online multi-class LPBoost," in *Proc. of CVPR*, Jun. 2010, pp. 3570–3577.

[11] Z. Khan, T. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," *IEEE Trans. on PAMI*, vol. 27, no. 11, pp. 1805–1918, Nov. 2005.



(a) frame 500 (b) frame 4500 (c) frame 6630 (d) frame 7498 (e) frame 7500

Fig. 15. Snapshots of a single-object tracking result. Red: FragTrack, green: OAB, blue: OMCLP, yellow: proposed approach. Tracks of OAB and OMCLP are maintained too long despite being lost (b)-(d); FragTrack removes the target "too early", (e).

[12] J. Yao and J.-M. Odobez, "Multi-camera multi-person 3D space tracking with MCMC in surveillance scenarios," in *European Conference on Computer Vision, workshop on Multi Camera and Multi-modal Sensor Fusion Algorithms and Applications (ECCV-M2SFA2)*, Marseille, France, Oct. 2008.

[13] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *Proc. of ICPR*, Istanbul, Turkey, 2010.

[14] C. Plagemann, D. Fox, and W. Burgard, "Efficient failure detection on mobile robots using particle filters with gaussian process proposals," in *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI)*, Hyderabad, India, 2007.

[15] S. L. Dockstader, N. S. Imennov, and A. M. Tekalp, "Markov-based failure prediction for human motion analysis," in *Proc. of CVPR*, vol. 2, Nice, France, 2003, pp. 1283–1288.

[16] D. Mikami, K. Otsuka, and J. Yamato, "Memory-based particle filter for face pose tracking robust under complex dynamics," in *Proc. of CVPR*, 2009, pp. 999–1006.

[17] ——, "Memory-based particle filter for tracking objects with large variation in pose and appearance," in *Proc. of ECCV*, vol. 3, 2010, pp. 215–228.

[18] A. D. Jepson, J. D. Fleet, and T. F. El-Margaghi, "Robust online appearance models for visual tracking," *IEEE Trans. on PAMI*, vol. 25, pp. 1296–1311, 2003.

[19] C. Zhang and Y. Rui, "Robust visual tracking via pixel classification and integration," in *Proc. of ICPR*, Hong Kong, Sep. 2006, pp. 37–42.

[20] D. T., G. Gordon, H. M., and J. Woodfill, "Integrated person tracking using stereo, color, and pattern detection," *IJCV*, pp. 175–185, 2000.

[21] Z. Kalal, J. Matas, and K. Mikolajczyk, "Online learning of robust object detectors during unstable tracking," in *Proc. of ICCV(CV workshop)*, 2009, pp. 1417–1424.

[22] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: HybridBoosted multiple object tracking," in *Proc. of CVPR*, Jun. 2009, pp. 2953–2960.

[23] B. Song, T.-Y. Jeng, E. Staudt, and A. K. Roy-Chowdhury, "A stochastic graph evolution framework for robust multi-target tracking," in *Proc. of ECCV*, vol. 1, 2010, pp. 605–619.

[24] C.-H. Kuo and R. Nevatia, "How does person identity recognition help multi-person tracking?" in *Proc. of CVPR*, 2011, pp. 1217–1224.

[25] B. Benfold and I. Reid, "Stable multi-target tracking in real-time surveillance video," in *Proc. of CVPR*, Jun. 2011, pp. 3457–3464.

[26] S. Duffner and J.-M. Odobez, "Exploiting long-term observations for track creation and deletion in online multi-face tracking," in *IEEE Conference on Automatic Face and Gesture Recognition*, Mar. 2011.

[27] M. Yang, Y. Wu, and G. Hua, "Context-aware visual tracking," *IEEE Trans. on PAMI*, vol. 31, pp. 1195–1209, 2009.

[28] M. Richardson and P. Domingos, "Markov logic networks," *Machine Learning*, vol. 62, pp. 107–136, 2006.

[29] M. Basseville, "Detecting changes in signals and systems – a survey," *Automatica*, vol. 24, pp. 309–326, 1988.

[30] K. Smith, D. Gatica-Perez, J. Odobez, and S. Ba, "Evaluating multi-object tracking," in *IEEE Conference on Computer Vision and Pattern Recognition - Workshop on Empirical Evaluation Methods in Computer Vision*, Jun. 2005, p. 36.

[31] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "PROST Parallel Robust Online Simple Tracking," in *Proc. of CVPR*, San Francisco, CA, USA, 2010.

[32] E. Ricci and J.-M. Odobez, "Learning large margin likelihood for realtime head pose tracking," in *Proc. of ICIP*, Oct. 2009.

Stefan Duffner received a Bachelor's degree in Computer Science from the University of Applied Sciences Konstanz, Germany in 2002 and a Master's degree in Applied Computer Science from the University of Freiburg, Germany in 2004. He performed his dissertation research at Orange Labs in Rennes, France, on face image analysis with statistical machine learning methods, and in 2008, he obtained a Ph.D. degree in Computer Science from the University of Freiburg. He then worked for 4 years as a post-doctoral researcher at the Idiap Research Institute in Martigny, Switzerland, in the field of computer vision and mainly face tracking. As of today, Stefan Duffner is an associate professor in the IMAGINE team of the LIRIS research lab at the National Institute of Applied Sciences (INSA) of Lyon, France.

Jean-Marc Odobez received an engineering degree from the Ecole Nationale Suprieure de Tlcommunications de Bretagne (ENSTBr), Brest, France, in 1990, and the PhD degree in signal processing from Rennes University, France, in 1994. He performed his dissertation research at IRISA/INRIA Rennes on dynamic scene analysis using statistical models. He then spent one year as a postdoctoral fellow at the GRASP Laboratory, University of Pennsylvania, working on visually guided robotic navigation problems. From 1996 until September 2001, he was an associate professor in computer science at the Universit du Maine, Le Mans, France. He is now a senior researcher at both the IDIAP Research Institute and EPFL, Switzerland, where he directs the Human Activity analysis team. His main areas of research are computer vision and machine learning techniques applied to multimedia content analysis as well as tracking and human activity and behavior recognition. He is the author or coauthor of more than 100 papers in international journals and conferences in his field. He is or was the principle investigator of 10 European and Swiss projects. He holds two patents on video motion analysis. He is the cofounder of the Swiss Klewel SA company active in the intelligent capture, indexing, and Web casting of multimedia conference and seminar events. He is a member of the IEEE, and associate editor of the Machine Vision and Application journal.