IDIAP RESEARCH REPORT


# EXPLOITING LONG-TERM OBSERVATIONS FOR TRACK CREATION AND DELETION IN ONLINE MULTI-FACE TRACKING

Stefan Duffner  Jean-Marc Odobez

Idiap-RR-01-2011

JANUARY 2011

**Centre du Parc**, Rue Marconi 19, P.O. Box 592, CH - 1920 Martigny
T +41 27 721 77 11  F +41 27 721 77 12  info@idiap.ch  www.idiap.ch

# Exploiting Long-Term Observations for Track Creation and Deletion in Online Multi-Face Tracking

Stefan Duffner and Jean-Marc Odobez

***Abstract*— In many visual multi-object tracking applications, the question when to add or remove a target is not trivial due to, for example, erroneous outputs of object detectors or observation models that cannot describe the full variability of the objects to track. In this paper, we present a real-time, online multi-face tracking algorithm that effectively deals with missing or uncertain detections in a principled way. The tracking 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 relying on head detections, likelihood models, and long term observations as well as object track characteristics has been designed to take the decision on when to add or remove a target from the tracker. The proposed method applied on three challenging datasets of more than 9 hours shows a significant performance increase compared to a traditional approach relying on head detection and likelihood models only.**

## I. Introduction

The real-time detection of objects is an important component in many computer vision applications, *e.g.* human-computer interaction, video-surveillance, and augmented reality. Moreover, faces play a crucial role in human communication. Thus, the automatic visual detection and tracking of faces is of particular interest in video-conferencing applications or in the analysis of social interaction.

The most straightforward approach to solve this problem is to employ a face detector [13]. However, despite much progress performed in recent years on multi-view face detection, and the use of these detectors in "simple" scenarios where people predominantly look towards the camera (video conferencing, HCI), this is not sufficient, and 30 to 40% of faces are missed as demonstrated in our results. There are indeed many situations where faces are not detected, which is especially due to variability in face appearance or lighting conditions. Above all, it is the consequence of less common head poses that people naturally take *e.g.* to look at other people in the same room, or to look down (at objects on a table, or if they are tired or bored) which often involves large head tilts. Unfortunately, the missed detections do not happen at random time, since for the above reasons, the difficult head postures can last for long periods (up to one minute in some of our recordings). In practice, 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 methods for visual tracking of multiple faces have been proposed in the literature (*e.g.* [10], [14], [7], [2]). Most of them work on improving tracking performance by proposing new features or new multi-cue fusion mechanisms, and results are demonstrated mostly on short sequences. Few of them address the issue of track initialisation (especially when doing performance evaluation). It is often assumed that a face detector is used for that purpose, but how to rely on a face detector? If a *high* confidence threshold is used, there is a higher risk of missing an early track initialisation. If a *low* threshold is chosen, false track alarms are likely to occur.

Even fewer works address track termination[1]. Indeed, how do we know at each point in time that a tracker is doing fine or that there is a failure? This an important issue in practice, since a false failure detection (*e.g.* due to the absence of detected faces) may mean losing a person track for a long period until the detector finds the face again. Most algorithms work recursively, and assessing tracking failure is often left to the (sudden) drop of objective or likelihood measures which are not that easy to control in practice [8], [9].

Finally, 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 occupy the same space or behave similarly over long periods. However, most of the existing face tracking methods ignore this long-term information, as they concentrate on videos that are often not longer than a minute. Or otherwise, long term information is mainly used to construct stable appearance models of tracked objects [3], [16], *e.g.* by working at different temporal scales [12]. Similarly, some approaches [1], [4] train an (object-specific) detector online, during tracking, to make it more robust to short-term and long-term appearance changes. Recently, Mikami *et al.* [8] introduced the Memory-based Particle Filter where a history of past states (and appearances [9]) 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.* [6], [11]) 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 essentially 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 the tracking of whole persons (not just the face).

[1] Note that principled methods exist to integrate track creation and termination within the tracking framework, *e.g.* Reversible-Jump Markov Chain Monte Carlo (RJ-MCMC) [5], [15]. 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.

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 threefold: i) employ an explicit probabilistic filtering framework to decide when to add or remove an object from the tracker based on a longer-term image features, the output of a face detector, as well as features coming from the face tracker itself (*e.g.* state variance); ii) propose the use of *long-term* image observations in order to cope effectively with missing or uncertain face detections; iii) a thorough performance evaluation on nearly $10$ hours of video conferencing videos involving $2$ to $5$ persons per view, with around $22,000$ annotations. Results demonstrate the validity of our approach.

The paper is organised as follows. The next Section describes our multi-face MCMC particle filter framework. Section III presents our approach for track creation and failure detection. And in section IV, we present our experimental results.

## 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 practice, this estimation is implemented using a particle filter with a Markov Chain Monte Carlo (MCMC) sampling scheme [5]. The main elements of the model are described in more detail in the following sections.

### 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 $\mathbf{k}_t = \{\mathbf{k}_{i,t}\}_{i=1..M}$. The variable $\mathbf{X}_{i,t}$ denotes the state of face $i$, which comprises the position, scale and eccentricity (*i.e.* the ratio between height and width) of the face bounding box. Each $\mathbf{k}_{i,t}$ denotes the status of face $i$ at time $t$, *i.e.* $\mathbf{k}_{i,t} = 1$ if the face is visible at time t, and $\mathbf{k}_{i,t} = 0$ 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:

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

*i.e.* the product of an interaction prior $p_0$ and of the dynamics of each individual faces. More precisely,

$$p(\mathbf{X}_{i,t}|\mathbf{X}_{t-1}, \mathbf{k}_t) = \begin{cases} p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1}) & \text{if } k_{i,t} = 1 \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

Note that this is actually feasible since the creation and deletion of targets are defined outside the filtering step (see next section), *i.e.* $k_{i,t}$ is not updated during this step. The dynamics $p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})$ of visible faces are described by a first-order auto-regressive model for the translation components and a zero-th order model with steady-state for the scale and eccentricity parameters. The steady-state is updated only when a detected face is associated and at a much slower pace compared to the frame-to-frame dynamics.

The interaction prior $p_0$ is defined as

$$p_0(\mathbf{X}_t|\mathbf{k}_t) = \prod_{\{i,j\}\in\mathcal{P}} \phi(\mathbf{X}_{i,t}, \mathbf{X}_{j,t}) \propto \exp\Big\{-\lambda_g \sum_{\{i,j\}\in\mathcal{P}} g(\mathbf{X}_{i,t}, \mathbf{X}_{j,t})\Big\}, \quad (4)$$

preventing targets to become too close to each other. The set $\mathcal{P} = \{\{i,j\} \mid k_{i,t} = 1 \wedge k_{j,t} = 1 \wedge i \neq j\}$ 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 $\lambda_g$ controls the strength of the interaction prior and was 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. Note that 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:

$$p(\mathbf{Y}_t|\tilde{\mathbf{X}}_t) = \prod_{i|k_{i,t}=1} p(\mathbf{Y}_{i,t}|\mathbf{X}_{i,t}). \quad (5)$$

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

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

where $D$ denotes the Euclidean distance[2] and $\lambda_D$ is set to $20$. More precisely, we divided the face into three horizontal bands and compute two normalised histograms in each of the band using two different discretisations: $N_b = 8$ and $N_b = 4$ bins per channel, using the scheme proposed in [10] which decouples coloured pixels (accumulated in $N_b \times N_b$ HS bins) from greyscale pixels (put in $N_b$ separate bins).

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

[2]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). Let $h^d_{i,t}$ denote the histograms computed from the detected face associated with a tracked object $i$. Then:

$$h^*_{i,t}(r) = (1-\epsilon)h^*_{i,t-1}(r) + \epsilon h^d_{i,t}(r) \quad \forall r\,, \tag{7}$$

where $\epsilon$ 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 [5] and works as follows.

Let $N$ be the total number of particles and $N_b$ the number of "burn-in" particles. At each tracking iteration, do the following steps:

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 $\{\tilde{\mathbf{X}}_{t-1}^{(s)}, s = (N_b+1)\dots N\}$ at time $t-1$ and sample the state of every visible target $i$ in $\tilde{\mathbf{X}}_t^{(0)}$ using the dynamics $p(\mathbf{X}_{i,t}|\mathbf{X}_{i,t-1})$;
2) sample iteratively $N$ particles from the posterior distribution of (1) using the Metropolis-Hastings algorithm according to:
   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:

$$a = \min\left(1, \frac{p(\tilde{\mathbf{X}}_t{}'|\mathbf{Y}_{1:t})\, q(\tilde{\mathbf{X}}_t{}^{(s)}|\tilde{\mathbf{X}}_t{}')}{p(\tilde{\mathbf{X}}_t{}^{(s)}|\mathbf{Y}_{1:t})\, q(\tilde{\mathbf{X}}_t{}'|\tilde{\mathbf{X}}_t{}^{(s)})}\right) \tag{8}$$

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

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

The proposal function $q()$ allows to select good candidates for the particle set. Efficiency in the MCMC is obtained by modifying object states one at a time. More precisely, the 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 sample the proposed state $\mathbf{X}'_{i,t}$ of face $i$ from:

$$q(\mathbf{X}'_{i,t}|\tilde{\mathbf{X}}_t) = \Big[(1-\alpha)\frac{1}{N-N_b}\sum_r p(\mathbf{X}'_{i,t}|\mathbf{X}^{(s)}_{i,t-1}) + \alpha\, p(\mathbf{X}'_{i,t}|\mathbf{X}^d_t)\Big] \tag{9}$$

where $\mathbf{X}^d_t$ denotes the state of the closest detection coming from a face detector [13] and associated with face $i$. That is, the proposal is defined as a mixture with $\alpha = 0.3$ controlling the proportion of samples generated by each mixture component. It relies on the dynamics from past particle to propose good state candidates assuming temporal smoothness, and on the output of a face detector which is useful for handling tracker drift.

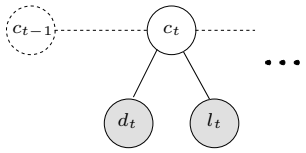


Fig. 1. The HMM for tracker target creation, used at each image pixel. The variable $c_t$ indicates if there is a face at a particular position. The probability of $c_t$ is estimated recursively using the observations $d_t$ and $l_t$.

## III. Target creation and removal

The way objects are added and removed from the tracker is a key feature of the proposed algorithm. In previous work [5], [15], target creation and removal are directly integrated into the probabilistic tracking framework. However, this requires global scene likelihood models which are difficult to obtain in this type of application (see footnote 1 on page 1). Our goal is to achieve a high precision during tracking, *i.e.* we would like to avoid as much as possible false alarms. This means that the tracker should be able to detect as quickly as possible if there is a tracking failure; simultaneously, it should not stop tracking when there is no failure, since the algorithm may have to wait for a long time before the face is detected again. Surprisingly, this problem has received little attention in the past.

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, and they receives different types of observations.

A face detector (for both frontal and profile views) is called every 10 frames (*i.e.* roughly once per second, as our algorithm is able to process around 10 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 frame did not greatly improve the tracking performance and considerably slowed down the algorithm. A detection gets associated with a target if their distance is smaller than two times their average width. Naturally, only un-associated detections are considered for the initialisation of a new target.

### A. Creation

For deciding when to add new targets to the face tracker, we propose a simple HMM that estimates the state 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. 1 illustrates this. (In the following, we drop the indexes $i$ and $j$ for clarity.) The HMM uses two different types of observed image features: one based on the output of the face detector, $d_t$, and a long-term "memory" of the states (*i.e.* positions) of tracked faces, $l_t$. The posterior probability $p(c_t|l_{1:t}, d_{1:t})$ is recursively estimated and then used to validate or reject

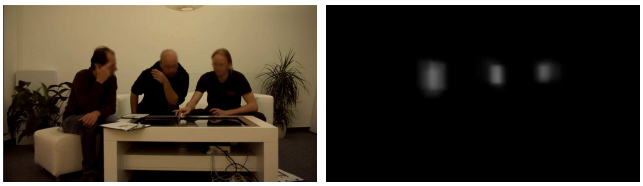

Fig. 2. Example image with corresponding tracking memory during tracking

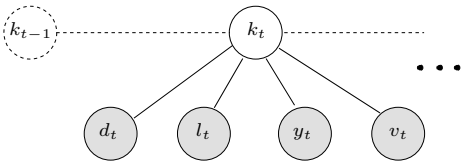


Fig. 3. The HMM for tracker target removal, used for each tracked face. The variable $k_t$ indicates for a given face if it is still tracked correctly or if a failure occurred. The probability of $k_t$ is estimated recursively using the observations $d_t, l_t, y_t$, and $v_t$.

detections coming from a face detector, helping to initialise new tracking targets.

The first observation measured at iteration $t$ and each image position $(i, j)$ is based on the output of the face detector:

$$d_t(i,j) = \begin{cases} 1 & \text{if } (i,j) \text{ is covered by one of the bounding} \\ & \text{boxes coming from the face detector,} \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

The likelihood $p(d_t|c_t)$ can be approximated with

$$p(d_t = 0|c_t = 0) = 1 - fa, \quad p(d_t = 1|c_t = 0) = fa,$$
$$p(d_t = 0|c_t = 1) = md, \quad p(d_t = 1|c_t = 1) = 1 - md\ ,$$

where $fa$ is the empirical false alarm rate and $md$ the missed detection rate of the face detector. Here we set $fa = 0.0001$ and $md = 0.4$.

The second likelihood $p(l_t|c_t)$ is based on a history of past image positions of tracked faces $l_t$, 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 $\bar{\mathbf{X}}_t$:

$$l_t = (1 - \beta)l_{t-1} + \beta I_t\ , \tag{11}$$

where $\beta = 0.001$ and

$$I_t(i,j) = \begin{cases} 1 & \text{if } (i,j) \text{ is covered by one of the} \\ & \text{bounding boxes described by } \bar{\mathbf{X}}, \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Fig. 2 shows an example of the tracking memory during a run of the face tracker. Intuitively, we would like to initialise targets more quickly in regions where a person has been "seen" previously. We approximated $p(l_t|c_t)$ with a pair of sigmoid functions

$$p(l_t|c_t = 1, \Theta) = a\ \arctan(\delta_l l_t - \mu_l)) + \frac{1}{2}. \tag{13}$$
$$p(l_t|c_t = 0, \Theta) = 1 - p(l_t|c_t = 1) \tag{14}$$

where $a$ is set to $\frac{1}{\pi}$, and the parameters $\Theta_l = (\delta_l, \mu_l)$, *i.e.* the slope and the offset of the sigmoid, have been trained offline with a set of observations collected from real tracking sequences.

Given the observations at time $t$ and the previous estimate of the posterior probability $p(c_{t-1}|d_{1:t-1}, l_{1:t-1})$, we compute the new posterior:

$$p(c_t|d_{1:t}, l_{1:t}) = \frac{p(d_t|c_t)p(l_t|c_t)\ p(c_{t-1}|d_{1:t-1}, l_{1:t-1})}{\sum_{c'_t} p(d_t|c'_t)p(l_t|c'_t)\ p(c_{t-1}|d_{1:t-1}, l_{1:t-1})} \tag{15}$$

Note that here we define the transition probability $p(c_t|c_{t-1}) = 1$ iff $c_t = c_{t-1}$ and 0 otherwise. Thus we don't include the term in 15.

Now, for each face detection that is not associated to any tracker target we take its centre position $(i, j)$ and compute the probability ratio:

$$r^c_t(i,j) = \frac{p(c_t(i,j) = 1|d_{1:t}(i,j), l_{1:t}(i,j))}{p(c_t(i,j) = 0|d_{1:t}(i,j), l_{1:t}(i,j))}\ . \tag{16}$$

If $r^c_t(i, j) > 1$, a new track is initialised from the detection.

### *B. Removal*

In a similar manner, for each tracked face $i$, we employ a HMM for estimating the hidden status variable $k_{i,t}$ indicating that the face is visible or not. We will drop the index $i$ in the following. Fig. 3 illustrates this model. In addition to the tracking memory $l_t$ and the output from the face detector $d_t$, two other observations are used: $y_t$, the observation likelihood of the mean state, and $v_t$, the maximum of the variances in $x$ and $y$ direction of the particle distribution of the respective face. We assume that if the target is still tracked correctly the observation likelihood $y_t$ should be high and the variance $v_t$ of the particles should be low.

Here, $p(d_t|k_t)$ is the same as $p(d_t|c_t)$ for creating a target, except that it is now calculated per object and not per image location. The other three observation likelihoods $p(l_t|k_t), p(y_t|k_t), p(v_t|k_t)$ have been modelled with three different sigmoid functions for $k = 1$ and with their inverse $1 - p(\cdot|k = 1)$ for $k = 0$ (*c.f.* 13-14). The parameters of these sigmoids have been learnt from data gathered from real tracker runs on annotated videos.

Let $o_t = \{o_{i,t}|_{i=1}^4\} = \{d_t, l_t, y_t, v_t\}$ be the set of observations at time $t$. Analogous to 15, we can estimate the posterior probability of $p(k_t|o_t)$ recursively:

$$p(k_t|o_{1:t}) = \frac{\prod_i p(o_{i,t}|k_t)\ p(k_t|k_{t-1})\ p(k_{t-1}|o_{1:t-1})}{\sum_{k'_t} [\prod_i p(o_{i,t}|k'_t)\ p(k'_t|k_{t-1})\ p(k_{t-1}|o_{1:t-1})]}\ , \tag{17}$$

The state transition probability $p(k_t|k_{t-1})$ is 0.999 for staying in the same state and 0.001 for changing state.

Finally, the tracking of a target is considered to have failed if the probability ratio $r^r_t < 1$, where

$$r^r_t = \frac{p(k_t = 1|o_t)}{p(k_t = 0|o_t)}\ . \tag{18}$$

| set | duration | # videos | # persons in view | # different persons | # annotated frames |
|---|---|---|---|---|---|
| 1 | 4 h | 6 | 2 | 5 | 5 000 |
| 2 | 2.5 h | 32 | 1-5 | 27 | 12 000 |
| 3 | 2.5 h | 2 | 2-4 | 7 | 4 800 |

TABLE I

STATISTICS ON THE THREE EVALUATION DATA SETS.

Fig. 4. Example frames from datasets 1 (top left), 2 (top right), and 3 (bottom row). Faces are blurred for privacy reasons.

## IV. EXPERIMENTAL RESULTS

### A. Data

Experiments have been conducted on a total amount of more than 9 hours of video data that has been extensively annotated. We used three sets of video files recorded in different environments (see Table I). According to our scenario of interest, the recorded people have been sitting around a table and filmed by a central camera. They are playing online games with people in a remote location using a laptop or touchscreen, *i.e.* they are often looking downwards, and in the videos their faces are often not detected by a standard face detector [13]. For efficiency reasons, the videos are processed at a resolution of $640 \times 360$ pixels, and the original frame rate has been changed to 12.5 fps.

Figure 4 shows example images from the three datasets. In dataset 1, the lighting conditions are good. However, as in the other datasets, there are long periods where faces are not detected. In dataset 2, the overall complexity of the videos is higher because of more difficult lighting conditions and because it contains more people including children, so the scene is more dynamic. Also, occlusions are occurring more frequently. The videos of dataset 3[3] are rather challenging for face tracking as people sit close to each other. Also, the lighting condition and image quality is worse in the second video of this set. Finally, the number of participants is varying throughout the videos.

### B. Tracking evaluation

*1)* **Performance measures:** In a given video frame, 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:

$$F = \frac{2a(B_i \cap B_j)}{a(B_i) + a(B_j)}\ , \tag{19}$$

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

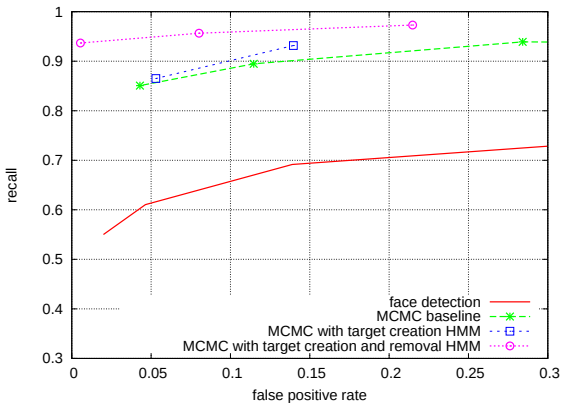


Fig. 5. False positive rate vs. recall for Dataset 1

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. We further define the recall and false positive rate as:

$$R = \frac{\sum_{i=2}^{G} \delta_i d_i}{\sum_{i=2}^{G} \delta_i} \qquad FP = \frac{\sum_{i=2}^{G} \delta_i f_i}{\sum_{i=2}^{G} \delta_i}\ , \tag{20}$$

where $G$ is the number of annotated frames, $d_i$ the proportion of correctly tracked/detected faces in frame $i$ (according to 19), $f_i$ is the number of false positive outputs divided by the number of ground truth objects in frame $i$, and $\delta_i$ is the duration between frame $i$ and $i-1$.

*2)* **Algorithms:** We compared the following algorithms:

- a standard face detector [13] with models for frontal and profile view
- an MCMC baseline tracker, *i.e.* the tracking algorithm described in section II. Every (un-associated) face detection is initialised as a new tracking target, up to a maximum number of $5$ targets in a given frame. We also tried to initialise a target only after several successive detections but this didn't have a big impact on the precision measures. A tracked target gets removed if it has no associated detections for 100 frames ($\equiv$ 8 seconds) or if the likelihood drops below $10\%$ of the running average of its likelihood.
- the proposed MCMC tracker with the HMM for target creation (see section III-A). Target removal has been done as for the baseline.
- the proposed MCMC tracker with HMMs for target creation and removal (see sections III-A and III-B).

*3)* **Results:** We plotted the recall and false positive rate 20 for the different algorithms with varying face detector threshold. Fig. 5-7 show the results. Clearly, for low detector thresholds the false positive (FP) rate of the face detector (red solid lines) 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 doesn't use the HMMs for target creation and removal it achieves a

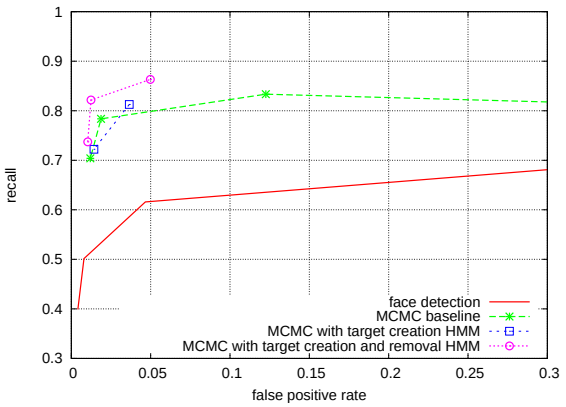


Fig. 6. False positive rate vs. recall for Dataset 2

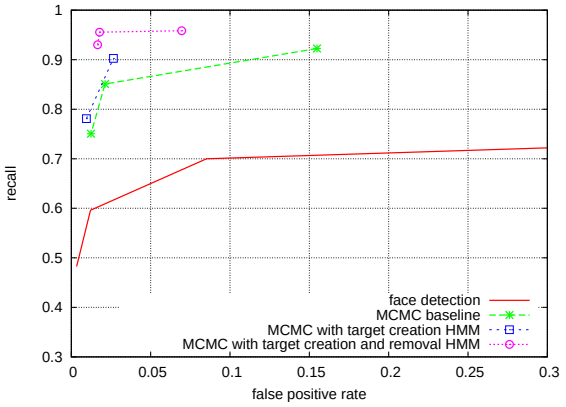


Fig. 7. False positive rate vs. recall for Dataset 3

good performance. When using the HMM for target creation (blue dashed lines) there is a slight increase in performance. The improvement is only marginal here because the face detector produces very few false positives. Finally, the purple dotted lines show the results of the proposed algorithm with both HMMs. The performance in terms of recall and FP rate is clearly better than for the baseline system without HMMs. This can also be seen in Table II, where we compared the performance of the different algorithms with a given face detector threshold (*i.e.* 3). Note also the the total number of interruptions of the tracker is decreased.

Note that the proposed algorithm runs in real-time, *i.e.* around 10-15 frames/s at a resolution of $640 \times 360$ pixels on an Intel PC at 3.16 GHz.

## V. Conclusions

We presented a 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 respectively *remove* a target to the tracker.

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

| | | face detection | tracking w/o HMMs | HMM 1 | HMM 1 +HMM 2 |
|---|---|---|---|---|---|
| 1 | recall<br>FP rate<br># interruptions | 61.0%<br>4.64% | 89.5%<br>11.47%<br>350 | 86.5%<br>5.3%<br>305 | 95.6%<br>8.02%<br>141 |
| 2 | recall<br>FP rate<br># interruptions | 61.6%<br>4.66%<br>– | 78.4%<br>1.8%<br>967 | 81.3%<br>3.7%<br>794 | 82.2%<br>1.2%<br>647 |
| 3 | recall<br>FP rate<br># interruptions | 59.6%<br>1.21%<br>– | 85.1%<br>2.12%<br>403 | 78.1%<br>0.95%<br>407 | 95.5%<br>1.78%<br>96 |

TABLE II

Performance comparison on the three datasets (with face detector threshold 3).


## Acknowledgement

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no. ICT-2007-214793


## References


[1] B. Babenko, M. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Proc. of CVPR*, pages 983–990, 6 2009.

[2] S. M. Bhandarkar and X. Luo. Integrated detection and tracking of multiple faces using particle filtering and optical ow-based elastic matching. *CVIU*, 113:708–725, 2009.

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

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

[5] Z. Khan, T. Balch, and F. Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. *IEEE Trans. on PAMI*, 27(11):1805–1918, 11 2005.

[6] Y. Li, C. Huang, and R. Nevatia. Learning to associate: HybridBoosted multiple object tracking. In *Proc. of CVPR*, pages 2953–2960, 6 2009.

[7] E. Maggio, M. Taj, and A. Cavallaro. Efficient multi-target visual tracking using random finite sets. *IEEE on Circuits and Systems for Video Technology*, 18(8):1016–1027, 2008.

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

[9] D. Mikami, K. Otsuka, and J. Yamato. Memory-based particle filter for tracking objects with large variation in pose and appearance. In *Proc. of ECCV*, volume 3, pages 215–228, 2010.

[10] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *Proc. of ECCV*, volume 1, pages 661–675, Copenhagen, May-June 2002.

[11] 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*, volume 1, pages 605–619, 2010.

[12] Darrell T., G. Gordon, Harville M., and J. Woodfill. Integrated person tracking using stereo, color, and pattern detection. *International Journal of Computer Vision*, pages 175–185, 2000.

[13] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. of CVPR*, volume 1, pages 511–518, 2001.

[14] Y. Wu, T. Yu, and G. Hua. Tracking appearances with occlusions. In *Proc. of CVPR*, volume 1, pages 789–795, 2003.

[15] Jian Yao and Jean-Marc Odobez. Multi-camera multi-person 3D space tracking with MCMC in surveillance scenarios. In *ECCV, workshop on Multi Camera and Multi-modal Sensor Fusion Algorithms and Applications (ECCV-M2SFA2)*, Marseille, France, October 2008.

[16] C. Zhang and Y. Rui. Robust visual tracking via pixel classification and integration. In *Proc. of ICPR*, pages 37–42, Hong Kong, 9 2006.