Online Adaptive Hidden Markov Model for Multi-Tracker Fusion

Tomas Vojir[a,*], Jiri Matas[a], Jana Noskova[b]

[a]*The Center for Machine Perception, FEE CTU in Prague*
Karlovo namesti 13, 121 35 Prague 2, Czech Republic
[b]*Faculty of Civil Engineering, CTU in Prague*
Thakurova 7/2077, 166 29 Prague 6, Czech Republic

Abstract

In this paper, we propose a novel method for visual object tracking called HMMTxD. The method fuses observations from complementary out-of-the box trackers and a detector by utilizing a hidden Markov model whose latent states correspond to a binary vector expressing the failure of individual trackers. The Markov model is trained in an unsupervised way, relying on an online learned detector to provide a source of tracker-independent information for a modified Baum- Welch algorithm that updates the model w.r.t. the partially annotated data.

We show the effectiveness of the proposed method on combination of two and three tracking algorithms. The performance of HMMTxD is evaluated on two standard benchmarks (CVPR2013 and VOT) and on a rich collection of 77 publicly available sequences. The HMMTxD outperforms the state-of-the-art, often significantly, on all datasets in almost all criteria.

Keywords:
visual tracking, on-line learning, hidden markov model, object detection

*corresponding author
Email addresses: vojirtom@cmp.felk.cvut.cz (Tomas Vojir), matas@cmp.felk.cvut.cz (Jiri Matas), noskova@fsv.cvut.cz (Jana Noskova)
URL: http://cmp.felk.cvut.cz/~vojirtom/ (Tomas Vojir), http://cmp.felk.cvut.cz/~matas/ (Jiri Matas)

1. Introduction

In the last thirty years, a large number of diverse visual tracking methods has been proposed [1, 2]. The methods differ in the formulation of the problem, assumptions made about the observed motion, in optimization techniques, the features used, in the processing speed, and in the application domain. Some methods focus on specific challenges like tracking of articulated or deformable objects [3, 4, 5], occlusion handling [6], abrupt motion [7] or long-term tracking [8, 9].

Three observations motivate the presented research. First, most trackers perform poorly if run outside the scenario they were designed for. Second, some trackers make different and complementary assumptions and their failures are not highly correlated (called complementary trackers in the paper). And finally, even fairly complex well performing trackers run at frame rate or faster on standard hardware, opening the possibility for multiple trackers to run concurrently and yet in or near real-time.

We propose a novel methodology that exploits a hidden Markov model (HMM) for fusion of non-uniform observables and pose prediction of multiple complementary trackers using an on-line learned high-precision detector. The non-uniform observables, in this sense, means that each tracker can produce its own type of "confidence estimate" which may not be directly comparable between each other.

The HMM, trained in an unsupervised manner, estimates the state of the trackers – failed, operates correctly – and outputs the pose of the tracked object taking into account the past performance and observations of the trackers and the detector. The HMM treats the detector output as correct if it is not in contradiction with its current most probable state in which the majority of trackers are correct. This limits the cases where the HMM would be wrongly updated by a false detection. For the potentially many frames where reliable detector output is not available, it combines the trackers. The detector is trained on the first image and interacts with the learning of the HMM by partially annotating the sequence of HMM states in the time of verified detections. The recall of the detector is not critical but it affects the learning rate of the HMM and the long-term properties of the HMMTxD method, i.e. its ability to reinitialize trackers after occlusions or object disappearance.

Related work. The most closely related approaches include Santner et al. [10], where three tracking methods with different rates of appearance adaptation are combined to prevent drift due to incorrect model updates. The approach uses simple, hard-coded rules for tracker selection. Kalal et al. [9] combine a tracking-

by-detection method with a short-term tracker that generates so called P-N events to learn new object appearance. The output is defined either by the detector or the tracker based on visual similarity to the learned object model. Both these methods employ pre-defined rules to make decisions about object pose and use one type of measurement, a certain form of similarity between the object and the estimated location. In contrary, HMMTxD learns continuously and causally the performance statistics of individual parts of the systems and fuses multiple "confidence" measurements in the form of probability densities of observables in the HMM. Zhang et al. [11] use a pool of multiple classifiers learned from different time spans and choose the one that maximize an entropy-based cost function. This method addresses the problem of model drifting due to wrong model updates, but the failure modes inherent to the classifier itself remains the same. This is unlike the proposed method which allows to combine diverse tracking methods with different inherent failure modes and with different learning strategies to balance their weaknesses.

Similarly to the proposed method, Wang et al. [12] and Bailer et at. [13] fuse different out-of-the box tracking methods. Bailer et al. combine offline the *outputs* of multiple tracking algorithms. There is no interaction between trackers, which for instance implies that the method avoids failure only if one method correctly tracks the whole sequence. Wang et al. use a factorial hidden Markov model and a Bayesian approach. The state space of their factorial HMM is the set of potential object positions, therefore it is very large. The model contains a probability description of the object motion based on a particle filter. Trackers interact by reinitializing those with low reliability to the pose of the most confident one. The Yuan et al. [14] using HMM in the same setup, but rather than merging multiple tracking method, they focus on modeling the temporal change of the target appearance in the HMM framework by introducing a observational dependencies. In contrast, the HMMTxD method is online with tracker interaction via a high precision object detector that supervises tracker reinitializations which happen on the fly. The appearance modeling is performed inside of each tracker and the HMMTxD capture the relation of the confidence provided by tracker and its performance, validated by the object detector, by the observable distributions. Moreover, the HMMTxD confidence estimation is motion-model free and this prevents biases towards support of trackers with a particular motion model.

Yoon et al. [15] combines multiple trackers in a particle filter framework. This approach models observables and transition behavior of individual trackers, but the trackers are self-adapting which makes it prone to wrong model updates. The adaptation of HMMTxD model is supervised by a detector method set to a spe-

cific mode of operation – near 100% precision – alleviating the incorrect update problem.

The contributions of the paper are: a novel method for fusion of multiple trackers based on HMMs using non-uniform observables, a simple, and so far unused, unsupervised method for HMMs training in the context of tracking, tunable feature-based detector with very low false positive rate, and the creation of a tracking system that shows state-of-the-art performance.

2. Fusing Multiple Trackers

HMMTxD uses a hidden Markov model (HMM) to integrate pose and observational confidence of different trackers and a detector, and updates its own confidence estimates that in turn define the pose that it outputs. In the HMM, each tracker is modeled as working correctly (1) or incorrectly (0). The HMM poses no constraints on the definition of tracker correctness, we adopted target overlap above a threshold. Having at our disposal $\mathbf{n}$ trackers, the set of all possible states is $\{s_1, s_2, \ldots, s_N\} = \{0, 1\}^{\mathbf{n}}, N = 2^{\mathbf{n}}$ and the initial state $s_1 = (1, 1, \ldots, 1)$. Note that the trackers are not assumed to be independent, because an independence of tracker correctness is not a realistic assumption. For example, if the tracking problem is relatively easy, all trackers tend to be correct and in the case of occlusion all tend to be incorrect (see the analysis in [16]). The number of states $2^{\mathbf{n}}$ grows exponentially with the number of trackers. However, we do not consider this a significant issue – due to "real-time" requirements of tracking, the need to combine more than a small number of trackers, say $\mathbf{n} = 4$, is unlikely.

The HMMTxD method overview is illustrated in Fig. 1. Each tracker provides an estimate of the object pose ($\mathbf{B}_i$) and a vector of observables ($\mathbf{x}_i$), which may contain a similarity measure to some model (such as normalized cross-correlation to the initial image patch, distance of template and current histograms at given position, etc.) or any other estimates of the tracker performance. The $\mathbf{x}_i, i = \{1, 2, \ldots, \mathbf{n}\}$ serve as observables to relate the tracker current confidence to the HMM. Each individual observable depends only on one particular tracker and its correctness, hence, they are assumed to be conditionally independent conditioned on the state of the HMM (which encodes the tracker correctness).

In general, there are no constraints on observable values, however, in the proposed HMM the observable values are required to be normalized to the $(0, 1)$ interval. The observables are modeled as beta-distributed random variables (Eq. 1) and its parameters are estimated online. The beta distribution was chosen for its versatility, where practically any kind of unimodal random variable on $(0, 1)$ can

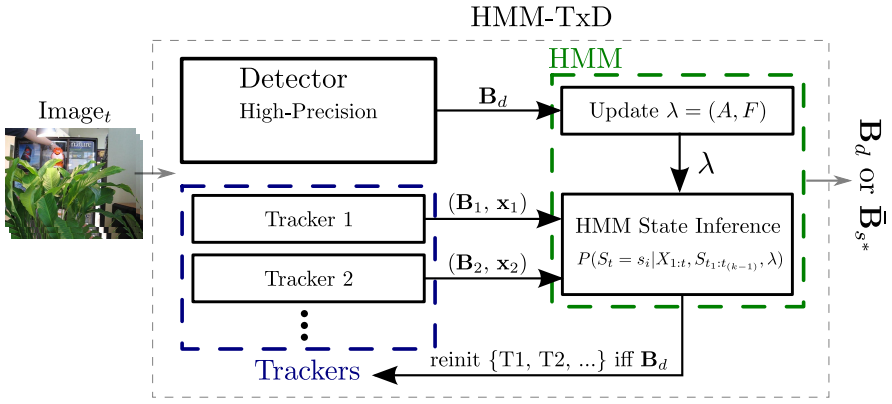


Figure 1: The structure of the HMMTxD. For each frame, the detector and trackers are run. Each tracker outputs a new object pose and observables $(\mathbf{B}_i, \mathbf{x}_i)$ and the detector outputs either the verified object pose $\mathbf{B}_d$ or nothing. If detector fires, HMM is updated and trackers are reinitialized and the final output is the $\mathbf{B}_d$, otherwise, HMM estimate the most probable state s^* and outputs an average bounding box $\bar{\mathbf{B}}_{s^*}$ of trackers that are correct in the estimated state s^*.

be modeled by the beta distribution, i.e. for any choice of any lower and upper quantiles, a beta distribution exists satisfying the given quantile constraint [17].

Learning the parameters of the beta distributions online is crucial for the adaptability to particular tracking scenes, where the observable values from a different trackers may be biased due to scene properties, or to adapt to a different types of observables of trackers and their correlations to the "real" tracker performance. For example, taking correlation with the initial target patch as an observable for one tracker and color histogram distance to a initial target for a second tracker, the correlation between their values and the performance of the tracker may differ depending on object rigidity and color distribution of object and background.

The HMM is parameterized by the pair $\lambda = (A, F)$, where A are the probabilities of state transition and F are the beta distributions of observables with shape parameters $p, q > 0$ and density defined for $x \in (0, 1)$

$$f(x|p, q) = \frac{x^{p-1}(1-x)^{q-1}}{\int_0^1 u^{p-1}(1-u)^{q-1}du}. \tag{1}$$

Since the goal is real-time tracking without any specific pre-processing, learning of HMM parameters has to be done online. Towards this goal, the object detector, which is set to operating mode with low false positive rate, is utilized to partially annotate the sequence of hidden states. In contrast to classical HMM, where only a sequence of observations $\mathbb{X} = \{X_t\}_{t=1}^T, X_t = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x_n})$ is available,

we are in a semi-supervised setting and have a time sequence $0 = t_0 < t_1 < t_2 \ldots < t_K \leq T$ of observed states of a Markov chain $\mathbb{S} = \{S_{t_k} = s_{i_k}, \{t_k\}_{k=1}^{K}\}$, and Markov chain starting again in state s_1, all trackers correct, at any time $\{t_k + 1, 0 \leq k \leq K\}$, since there are reinitialized to common object pose. This information is provided by the detector, where $\{t_k\}_{k=1}^{K}$ is a sequence of detection times. The HMM parameters are learned by a modified Baum-Welch algorithm run on the observations $\mathbb{X}$ and the annotated sequence of states $\mathbb{S}$. The partial annotation and HMM parameter estimation update is done strictly online.

The output of the HMMTxD is an average bounding box of correct trackers of the current most probable state s_t^*. For $t_{(k-1)} < t < t_k, 1 \leq k \leq K$ the forward-backward procedure [18] for HMM is used to calculate probability of each state at time t (see Eq. A.1-A.7) and the state $s_t^* \in \{0, 1\}^{\mathbf{n}} \setminus (0, 0, \ldots, 0)$ is the state for which

$$P(S_t = s_i | X_1, \ldots, X_t, S_{t_1}, \ldots, S_{t_{(k-1)}}, \lambda) \tag{2}$$

is maximal. This equation is computed using Eq. A.5 and maximized w.r.t $i, 1 \leq i \leq N$. For $t_K < t \leq T$ the Eq. 2 holds with $t_{(k-1)} = t_K$. This ensures that the algorithm outputs a pose for each frame which is required by most benchmark protocols. Illustration of the tracking process and HMM insight is shown in Fig. 2. Theoretically the parameters of HMM could be updated after each frame. However, in our implementation, learning takes place only at frames where the detector positively detects the object, i.e. the sequence of states starting and ending with observed state inferred by the detector[1]. The detector is used only if the detection pose is not in contradiction with the pose of the current most probable state in which the majority of trackers are correct. This ensure that even when the detector makes a mistake, the HMM is not wrongly updated. When we are in the state that one or none of the trackers are correct, the detector get precedence.

3. Learning the Hidden Markov Model

For learning of the parameters λ of the HMM a MLE inference is employed, however maximizing the likelihood function $P(\mathbb{X}, \mathbb{S}|\lambda)$ is a complicated task that cannot be solved analytically. In the proposed method, the Baum-Welch algorithm [19] is adapted. The Baum-Welch algorithm is a widespread iterative pro-

[1]If pure online fusion is not required, future observations can also be used to determine the probability of each state.

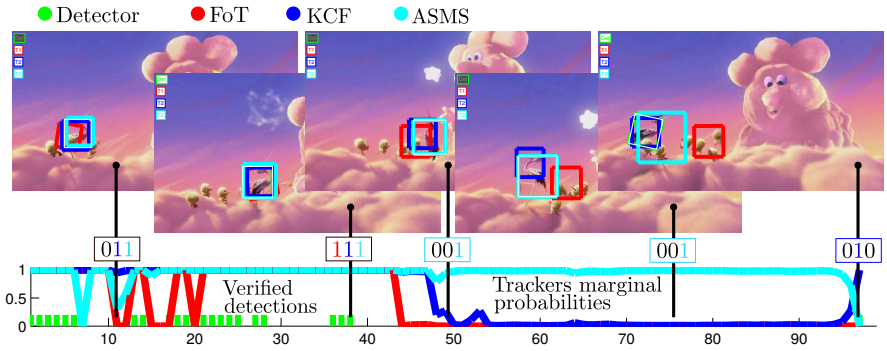


Figure 2: Illustration of HMM state and trackers probability estimation during tracking. The bottom graph shows the marginal probabilities for each tracker being correct and the detection times (green spikes). Above the graph the inferred states s_t^* with color encoded correct trackers (1) are displayed. The final output is defined by the state s_t^* and the bounding box is highlighted by white color. Best viewed zoomed in color.

cedure for estimating parameters of HMM where each iteration increases the likelihood function but, in general, the convergence to the global maximum is not guaranteed. The Baum-Welch algorithm is in fact an application of the EM (Expectation-Maximization) algorithm [20].

3.1. Classical Baum-Welch Algorithm

Let us assume the HMM with N possible states $\{s_1, s_2, \ldots, s_N\}$, the matrix of state transition probabilities $A = \{a_{ij}\}_{i,j=1}^{N}$, the vector of initial state probabilities $\pi = (1, 0, 0, \ldots, 0)$, the initial state $s_1 = (1, 1, \ldots, 1)$, a sequence of observations $\mathbb{X} = \{X_t\}_{t=1}^{T}$, $X_t \in R^m$ and $F = \{f_i(x)\}_{i=1}^{N}$ the system of conditional probability densities of observations conditioned on $S_t = s_i$

$$f_i(x) = f(x|S_t = s_i) \text{ for } 1 \leq i \leq N, 1 \leq t \leq T, x \in R^m \tag{3}$$

where S_t are random variables representing the state at time t, and $\lambda = (A, F)$ is denoting the parameter set of the model.

Let us denote

$$Q(\lambda, \lambda') = \sum_{\mathfrak{s} \in \mathfrak{S}} P(\mathfrak{s}|\mathbb{X}, \lambda) \log[P(\mathfrak{s}, \mathbb{X}|\lambda')], \tag{4}$$

where $\mathfrak{S} = \{s_1, s_2, \ldots, s_N\}^T$ is a set of all possible T-tuples of states and $\mathfrak{s} \in \mathfrak{S}, \mathfrak{s} = (\mathfrak{s}_1, \ldots, \mathfrak{s}_t, \ldots, \mathfrak{s}_T)$ is one sequence of states. According to Theorem 2.1. in [19]

$$Q(\lambda, \lambda') \geq Q(\lambda, \lambda) \Rightarrow P(\mathbb{X}|\lambda') \geq P(\mathbb{X}|\lambda) \tag{5}$$

and the equality holds if and only if $P(\mathfrak{s}|\mathbb{X},\lambda) = P(\mathfrak{s}|\mathbb{X},\lambda')$ for $\forall \mathfrak{s} \in \mathfrak{S}$. The classical Baum-Welch algorithm repeats the following steps until convergence:

1. Compute $\lambda^* = \arg\max_\lambda Q(\lambda_n, \lambda)$
2. Set $\lambda_{n+1} = \lambda^*$.

3.2. Modified Baum-Welch Algorithm

We propose the modified Baum-Welch algorithm that exploits the partially annotated sequence of states, where the known states are inferred from the detector output. Let $0 = t_0 < t_1 < t_2 \ldots < t_K \leq T$ be a sequence of detection times, $\mathbb{S} = \{S_{t_k} = s_{i_k}, \{t_k\}_{k=1}^K\}$ be observed states of Markov chain, marked by the detector, and $S_{t_k+1} = s_1$ for $0 \leq k \leq K$. So the sequence of observations of the HMM is divided into $K+1$ independent subsequences, each with a fixed initial state s_1, the first K subsequences with a known terminal state defined by the detector and the last subsequence with an unknown terminal state.

The following equations are obtained by employing the modification to the Baum-Welch algorithm,

$$\log[P(\mathfrak{s},\mathbb{X},\mathbb{S}|\lambda)] = \sum_{t=1}^{T-1} \log a_{\mathfrak{s}_t\mathfrak{s}_{t+1}} + \sum_{t=1}^{T} \log f_{\mathfrak{s}_t}(X_t), \tag{6}$$

$$\begin{aligned} Q(\lambda_n,\lambda) = &\sum_{\mathfrak{s}\in\mathfrak{S}} P(\mathfrak{s}|\mathbb{X},\mathbb{S},\lambda_n) \sum_{t=1}^{T-1} \log a_{\mathfrak{s}_t\mathfrak{s}_{t+1}} + \\ &\sum_{\mathfrak{s}\in\mathfrak{S}} P(\mathfrak{s}|\mathbb{X},\mathbb{S},\lambda_n) \sum_{t=1}^{T} \log f_{\mathfrak{s}_t}(X_t). \end{aligned} \tag{7}$$

The maximization of the $Q(\lambda_n,\lambda)$ can be separated to maximization w.r.t. transition probability matrix $A = \{a_{ij}\}_{i,j=1}^N$ by maximizing the first term and w.r.t. observable densities $F = \{f_i(x)\}_{i=1}^N$ by maximizing the second term.

The maximization of Eq. 7 w.r.t. A constrained by $\sum_{j=1}^N a_{ij} = 1$ for $1 \leq i \leq N$ is obtained by re-estimating the parameters $\hat{A} = \{\hat{a}_{ij}\}_{i,j=1}^N$ as follows:

$$\begin{aligned} \hat{a}_{ij} &= \frac{\text{expected number of transitions from state } s_i \text{ to state } s_j}{\text{expected number of transitions from state } s_i} \\ &= \frac{\sum_{(t=1 \text{ and } t\neq t_k, 1\leq k\leq K)}^{T-1} P(S_t = s_i, S_{t+1} = s_j|\mathbb{X},\mathbb{S},\lambda)}{\sum_{(t=1 \text{ and } t\neq t_k, 1\leq k\leq K)}^{T-1} P(S_t = s_i|\mathbb{X},\mathbb{S},\lambda)}. \end{aligned} \tag{8}$$

This equation is computed using modified forward and backward variables of the Baum-Welch algorithm to reflect the partially annotated states. For the exact derivation of formulas for computation of $\hat{a}_{ij}$ see the Appendix A.

3.2.1. Learning Observable Distributions

The maximization of Eq. 7 w.r.t. $F = \{f_i(x)\}_{i=1}^{N}$ depends on assumptions on the system of probability densities F. It is usually assumed (e.g. in [18, 19]) that F is a system of probability distributions of the same type and differ only in their parameters.

In the HMMTxD the m-dimensional observed random variables $X_t = (X_t^1, X_t^2, \ldots, X_t^m) \in R^m$ are assumed conditionally independent and to have the beta-distribution, so $f_i(x), 1 \leq i \leq N$ are products of m one-dimensional beta distributions with parameters of shape $\{({p_i}^j, {q_i}^j)\}_{j=1}^{m}, 1 \leq i \leq N$. In this case maximization of the second term of the Eq. 7 is an iterative procedure using inverse digamma function which is very computationally expensive [17].

We propose to estimate the shape parameters of the beta distributions with a generalized method of moments. The classical method of moments is based on the fact that sample moments of independent observations converge to its theoretical ones due to the law of large numbers for independent random variables. In the HMMTxD observations $\mathbb{X} = \{X_t\}_{t=1}^{T}$ are not independent. The generalized method of moments is based on the fact that $\{X_t - E(X_t|X_1, X_2, \ldots, X_{t-1})\}_{t=1}^{T}$ is a sequence of martingale differences for which the law of large numbers also holds. Using the generalized method of moments gives estimates of the parameters of shape

$$\hat{p}_i^j = \hat{\mu}_i^j \left(\frac{\hat{\mu}_i^j(1-\hat{\mu}_i^j)}{(\hat{\sigma}_i^j)^2} - 1 \right) \tag{9}$$

and

$$\hat{q}_i^j = (1-\hat{\mu}_i^j) \left(\frac{\hat{\mu}_i^j(1-\hat{\mu}_i^j)}{(\hat{\sigma}_i^j)^2} - 1 \right) \tag{10}$$

where

$$\hat{\mu}_i^j = \frac{\sum_{t=1}^{T} X_t^j P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}{\sum_{t=1}^{T} P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)} \tag{11}$$

and

$$(\hat{\sigma}_i^j)^2 = \frac{\sum_{t=1}^{T} (X_t^j - \hat{\mu}_i^j)^2 P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}{\sum_{t=1}^{T} P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}. \tag{12}$$

Let us denote the system of probability densities with re-estimated parameters as $\hat{F} = \{\hat{f}_i(x)\}_{i=1}^{N}$. The generalized method of moments is described in detail in the Appendix B.

3.2.2. Algorithm Overview

The complete modified Baum-Welch algorithm is summarized in Alg. 1, where after each iteration $P(\mathbb{X}, \mathbb{S}|\lambda_{n+1}) \geq P(\mathbb{X}, \mathbb{S}|\lambda_n)$ and we repeat these steps until convergence. Note that $\hat{A}_n$ is a maximum likelihood estimate of A therefore always increases $P(\mathbb{X}, \mathbb{S}|\lambda_n)$ (shown in [18]) but $\hat{F}_n$ is estimated by the method of moments so the test on likelihood increase is required ("if statement" in the Alg. 1). In fact, this algorithm structure match to the generalized EM algorithm (GEM) introduced in [20].

Algorithm 1: Algorithm for HMM parameters learning

Input: $\mathbb{X}, \mathbb{S}, \lambda_n = (A_n, F_n)$
Output: $\lambda_{n+1} = (A_{n+1}, F_{n+1})$
repeat
  Compute likelihood $P(\mathbb{X}, \mathbb{S}|\lambda_n)$
  Estimate $\hat{A}_n$ by Eq. 8 and $\hat{F}_n$ by Eq. 9, 10
  if $P(\mathbb{X}, \mathbb{S}|\hat{A}_n, \hat{F}_n) < P(\mathbb{X}, \mathbb{S}|A_n, F_n)$ **then**
    $\lambda_{n+1} = (\hat{A}_n, F_n)$
  else
    $\lambda_{n+1} = (\hat{A}_n, \hat{F}_n)$
  $\lambda_n = \lambda_{n+1} = (A_{n+1}, F_{n+1})$
until *convergence* $\vee$ *max number of iteration*

4. Feature-Based Detector

The requirements for the detector are: adjustable operation mode (e.g. set for high precision but possibly low recall), (near) real-time performance and the ability to model pose transformations up to at least similarity (translation, rotation, isotropic scaling). Basically, any detector-like approach can be used and it may vary based on application. We choose to adapt a feature-based detector which has been shown to perform well in image retrieval, object detection and object tracking [8] tasks.

There are many possible combinations of features and their descriptors with different advantages and drawbacks. We exploit multiple feature types: specifically, Hessian keypoints with the SIFT [21] descriptor, ORB [22] with BRISK and ORB with FREAK [23]. Each feature type is handled separately, up to the point where point correspondences are established. A weight is assigned to each feature type w^g and is set to be inversely proportional to the number of features on the reference template, to balance the disparity in individual feature numbers.

The detector works as follows. In the initialization step, features are extracted from the inside and the outside of the region specifying the tracked object. Descriptors of the features outside of the region are stored as the background model.

Usually, the input region is not 100% occupied by the target; therefore, fast color segmentation [24] attempts to delineate the object more precisely than the axis-aligned bounding box to remove the features that are most likely not on the target. The step is not critical for the function of the detector, since the bounding box is a fall-back option. We assume that at least 50% of the bounding box is filled with pixels that belong to the target, if the segmentation fails (returns a region containing less than 50% of area of the bounding box), all features in the initial bounding box are used.

Additionally, for each target feature, we use a normal distribution $\mathcal{N}(\mu^f, \sigma^f)$ to model the similarity of the feature to other features. The parameters μ^f and σ^f are estimated in the first frame by randomly sampling 100 features, other than f, and computing distances to the feature f, from which the mean and variation are computed. This allows defining the quality of correspondence matches in a probabilistic manner for each feature, thus getting rid of global static threshold for the acceptable correspondence distance.

In the detection phase, features are detected and described in the whole image. For each feature g_i from the image the nearest neighbour (in Euclidean space or in Hamming distance metric space, depending on the feature type) feature b^* from the background model and the nearest neighbour feature f^* from the foreground model are computed. A tentative correspondence is formed if the feature match passes the second nearest neighbour test and a probability that the correspondence distance belongs to the outlier distribution is lower than a predefined significance set to 0.1%. So

$$\frac{d(g_i, f^*)}{d(g_i, b^*)} < 0.8 \ \wedge \ \mathcal{F}(d(g_i, f^*)|\mu^{f^*}, \sigma^{f^*}) < 0.1\% \tag{13}$$

where $\mathcal{F}(d|\mu^{f^*}, \sigma^{f^*})$ is a c.d.f. of the normal distribution with parameters μ^{f^*}

and σ^{f^*} of a distance distribution of features not corresponding to f^*. The 0.1% significance corresponds to the $\mu - 3\sigma$ threshold. Finally, RANSAC estimates the target current pose using a sum of weighted inliers as a cost function for model support

$$\text{cost} = \sum_i w^{g_i} * [g_i == \text{inlier}], \tag{14}$$

which takes into account the different numbers of features per feature type on the target.

The decision whether the detected pose is considered correct depends on the number of weighted inliers that supports the RANSAC-selected transformation and it controls the trade-of between precision and recall of the method. This threshold is automatically computed in the first frame of the sequence as $\max(5, \min(0.03 * \text{max\_number\_of\_features\_in\_target\_bbox}, 10))$. The threshold interval (5,10) and the feature multiplier (0.03) were set experimentally to have the false positive rate close to zero for the most of the testing sequences. Furthermore, majority voting is used to verify that the detection is not in contradiction to the estimated HMM state, i.e. if we are in the state where two or more (majority) trackers are correct and the detector is not consistent with them, the detection is not used. This mitigates the false positive detections, therefore HMM updates, when the trackers works correctly.

The true and false positives for 77 sequences are shown in Fig. 3, where the detector works on almost all sequences with zero false positive rate (0.46% average false positive rate on the dataset) and 30% recall rate. The failure cases of this feature-based detector are mostly caused by the imprecise initial bounding box, which contains large portion of structured background (i.e. background where the detector finds features) and due to the presence of similar object in the scene, e.g. sequences *hand2, basketball, singer2*.

5. HMMTxD Implementation

To demonstrate the performance of the proposed framework, a pair and a triplet of published short-term trackers were plugged into the framework to show the performance gain by combination of a different number of trackers. As Bailer et al. [13] pointed out, not all trackers when combined can improve the overall performance (i.e. adding tracking method with similar failure mode will not benefit).

We therefore choose methods that have a different designs and work with different assumptions (e.g. rigid global motion vs. color mean-shift estimation

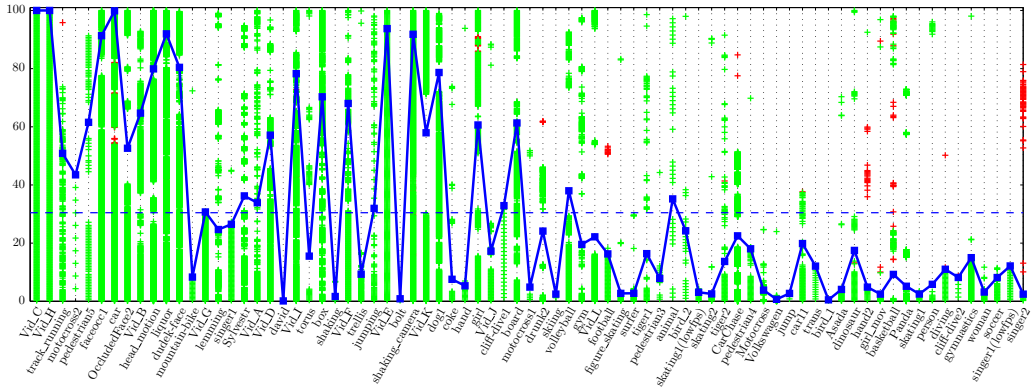


Figure 3: Frames with the detections for 77 sequences dataset. The green marks show the true positive detection and red marks are false positive. The blue line shows the recall of the detector and blue dashed line shows the average recall over all sequences. The length of each sequence is normalized to range $(0, 100)$.

vs. maximum correlation response). These trackers are the Flock of Trackers (FoT) [25], scale adaptive mean-shift tracker (ASMS) [26] and kernelized correlation filters (KCF) [27]. This choice shows that superior performance can be achieved by using simple, fast trackers (above 100fps) that may not represent the state-of-the-art. The trackers can be arbitrarily replaced depending on the user application or requirements.

Trackers

The Flock of Trackers (FoT) [25] evenly covers the object with patches and establishes frame-to-frame correspondence by the Lucas-Kanade method [28]. The global motion of the target is estimated by RANSAC.

The second tracker is a scale adaptive mean-shift tracker (ASMS) [26] where the object pose is estimated by minimizing the distance between RGB histograms of the reference and the candidate bounding box. The KCF [27] tracker learns a correlation filter by ridge regression to have high response to target object and low response on background. The correlation is done in the Fourier domain which is very efficient.

These three trackers have been selected since they are complementary by design. FoT enforces a global motion constrain and works best for rigid object with texture. On the other hand, ASMS does not enforce object rigidity and is well suited for articulated or deformable objects assuming their color distribution is

discriminative w.r.t. the background. KCF can be viewed as a tracking-by detection approach using sliding window like scanning.

For each tracker position, two global observable measurements are computed, namely the Hellinger distance between the target template histogram and the histogram of the current position and normalized cross-correlation score of the current patch and the target model patch. These target models are initialized in the first frame and then updated exponentially with factor of 0.5 during each positive detection of the detector part. Additionally, each tracker produces its own estimate of performance. For FoT it is the number of predicted correspondences (for details please see [25]) that support the global model. For ASMS it is the Hellinger distance between its histogram model and current neighbourhood background (i.e. color similarity of the object and background) and for KCF it is a correlation response of the tracking procedure.

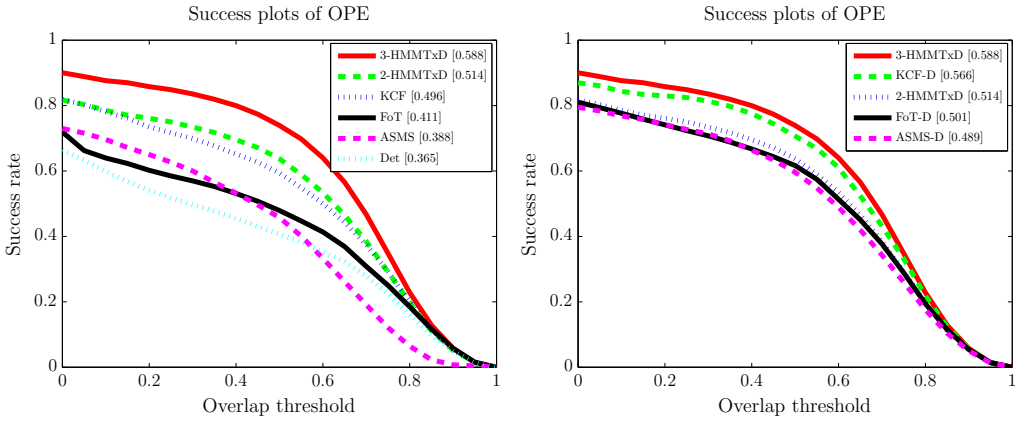


Figure 4: CVPR2013 OPE benchmark comparison of individual trackers and their combination in the proposed HMMTxD. The 2-HMMTxD denotes the combination of FoT and ASMS trackers and 3-HMMTxD is a combination of FoT, ASMS and KCF trackers. Det stands for the proposed detector. The right plot show simple combination of individual trackers with the proposed detector. Suffix "-D" refers to the combination with detector.

6. Experiments

The HMMTxD was compared with state-of-the-art methods on two standard benchmarks and on a dataset TV77[2] containing 77 public video sequences col-

[2] http://cmp.felk.cvut.cz/~vojirtom/dataset/index.html

lected from tracking-related publications. The dataset exhibits wider diversity of content and variability of conditions than the benchmarks.

Parameters of the method were fixed for all the experiments. In the HMM, the initial beta distribution shape parameters (p, q) were set to $(2, 1)$ for correct state (1) and $(1, 2)$ for fail state (0) for all observations and the transition matrix was set to prefer staying in the current state. The transition matrix has 0.98 on diagonal, 0 in fist column, 0.001 in last column, $1e-10$ in last row and 0.05 otherwise. The matrix is normalized so that rows sum to one. States in the matrix are binary encoded starting from the left column which corresponds to the state $s_1 = (1, ..., 1)$. The number of iteration for Baum-Welch alg. was set to 3.

The processing speed on the VOT2015 dataset is (in frames per second) minimum 1.03, maximum 33.72 and average 10.83 measured on a standard notebook with Intel Core-i7 processor. This speed is mostly affected by the number of features detected in the images which correlates to the resolution of the image (in the dataset the range is from 320x180 to 1280x720).

First, we compare the performance of individual parts of the HMMTxD framework (i.e. KCF, ASMS, FoT trackers) and their combination via HMM as proposed in this paper. Two variants of HMMTxD are evaluated – 2-HMMTxD refers to combination of FoT and ASMS trackers and the 3-HMMTxD to combination of all mentioned trackers. We also show the benefit of the proposed detector when simply combined with the individual trackers in such way that if detector fires the tracker is reinitialized. The Figure 4 shows the benefit gained from the detector and further consistent improvement achieved by the combination of the trackers. More detailed per sequence analysis on the TV77 dataset (Fig. 5 and Fig. 6) shows more clearly the efficiency of learning tracker performance online. In almost all sequences the HMMTxD is able to identify and learn which trackers works correctly and achieve the performance of at least the best tracker or higher (e.g. *motocross1, skating1(low), Volkswagen, singer1, pedestrian3, surfer*). Most notable failure cases are caused by the detector failure, e.g. in sequences *singer2, woman, skating1, basketball, girl_mov*.

In all other experiments, the abbreviation HMMTxD refers to the combination of all 3 trackers.

Evaluation on the CVPR2013 Benchmark [29] that contains 50 video sequences. Results on the benchmark have been published for about 30 trackers. The benchmark defines three types of experiments: (i) one-pass evaluation (OPE) – a tracker initialized in the first frame is run to the end of the sequence, (ii) temporal robustness evaluation (TRE) – the tracker is initialized and starts at a random frame, and (iii) spatial robustness evaluation (SRE) – the initialization is perturbed

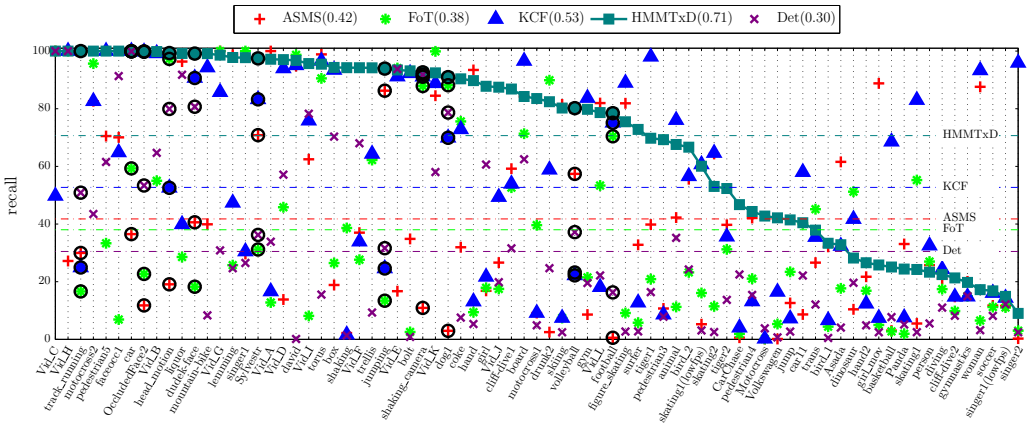


Figure 5: Per sequence analysis of the single trackers (i.e. KCF, ASMS, FoT) and the proposed HMMTxD. The average recall is shown by the dashed lines (precise number is in the legend). Black circles mark grayscale sequences. The sequences are ordered by HMMTxD performance.

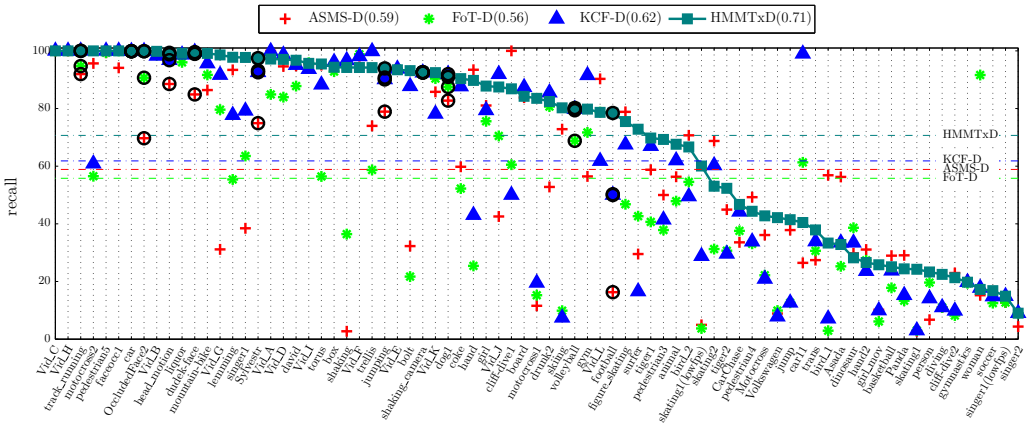


Figure 6: Per sequence analysis of the single trackers combined with the detector (i.e. KCF-D, ASMS-D, FoT-D) and the proposed HMMTxD. The average recall is shown by the dashed lines (precise number is in the legend). Black circles mark grayscale sequences. The sequences are ordered by HMMTxD performance.

spatially. Performance is measured by precision (spatial accuracy, i.e. center distance of ground truth and reported bounding box) and success rate (the number of frames where overlap with the ground truth was higher than a threshold). The results are visualized in Fig. 7 where only results of the 10 top performing trackers are plotted. Together with the tracker from this benchmark, we also added the MEEM [11] tracker, which is a recent state-of-the-art tracker. The proposed

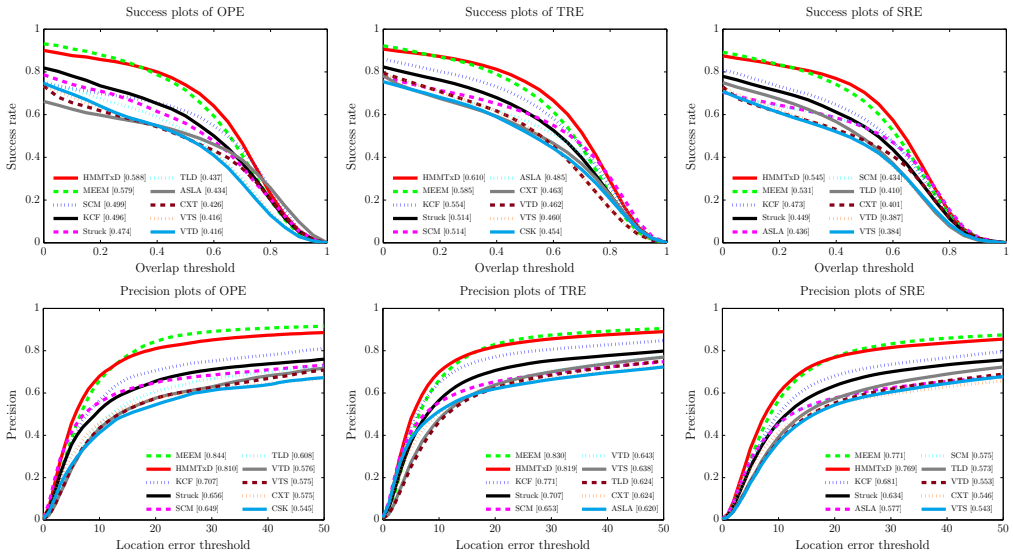


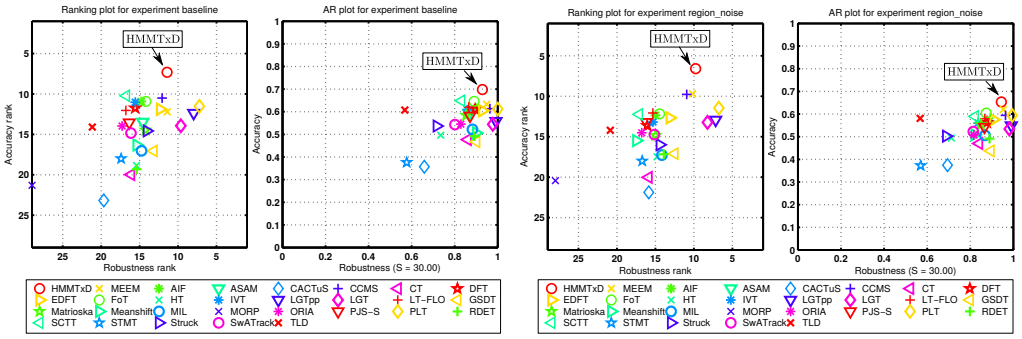
Figure 7: Evaluation of HMMTxD on the CVPR2013 Benchmark [29]. The top row shows the success rate as a function of the overlap threshold. The bottom row shows the precision as a function of the localization error threshold. The number in the legend is AUC, the area under ROC-curve, which summarizes the overall performance of the tracker for each experiment.

HMMTxD outperforms all trackers in the success rate in all three experiments. Its precision is comparable to MEEM [11] the top performing tracker in terms of precision. HMMTxD outperforms significantly the OPE results reported in Wang et al. [12], where 5 top performing trackers from this particular benchmark were used for combination (other experiments were not reported in the paper).

VOT2013 benchmark [30] evaluates trackers on a collection containing 16 sequences carefully selected from a large pool by a semi- automatic clustering method. For comparison, results of 27 tracking methods are available and the added MEEM tracker was evaluated by us using default setting from the publicly available source code. The performance is measured by accuracy, average overlap with the ground truth, and robustness, the number of re-initialization of the tracker so that it is able to track the whole sequence. Average rank of trackers is used as an overall performance indicator.

In this benchmark, the proposed HMMTxD achieves clearly the best accuracy (Fig. 8). With less than one re-initialization per sequence it performs slightly worse in terms of robustness due to two reasons.

Firstly, the HMM recognizes a tracker problem with a delay and switching to other tracker (here even one frame where the overlap with ground truth is



(a) baseline experiment (b) region-noise experiment

Figure 8: Evaluation of HMMTxD on the VOT 2013 Benchmark [30]. HMMTxD result is shown as the red circle. The left plot shows the ranking in accuracy (vertical axis) and robustness (horizontal axis) and the right plot shows the raw average values of accuracy and robustness (normalized to the $(0, 1)$ interval). For both plots the top right corner is the best performance.

zero leads to penalization) and secondly the VOT evaluation protocol, which require re-initialization after failure and to forget all previously learned models (the VOT2013 refer to this as causal tracking), therefore the learned performance of the trackers is forgotten and has to be learned from scratch.

The results for the baseline and region-noise experiments are shown in Fig. 8. Note that the ranking of the methods differs from the original publication since two new methods (HMMTxD and MEEM) were added and the relative ranking of the methods changed. The top three performing trackers and their average ranks are HMMTxD (8.77), PLT (9.24), LGTpp [31] (10.11). MEEM tracker ends up at the fifth place with average rank 10.87. The rankings were obtained by the toolkit provided by the VOT in default settings for baseline and region noise experiments.

The second best performing method on the VOT2013 is the unpublished PLT for which just a short description is available in [30]. PLT is a variation of structural SVM that uses multiple features (color, gradients). STRUCK [32] and MEEM [11] are similar method to the PLT based on SVM classification. We compared these method with HMMTxD on the diverse 77 videos along with the TLD [9] which has a similar design as HMMTxD. HMMTxD outperforms all these methods by a large margin on average recall – measured as number of frames where the tracker overlap with ground truth is higher than 0.5 averaged over all sequences. Results are shown in Fig. 9. Qualitative comparison of these state-of-the-art methods is shown in Fig. 10. Even for sequences with lower recall (e.g. *bird_1, skating2*), the HMMTxD is able to follow the object of interest.

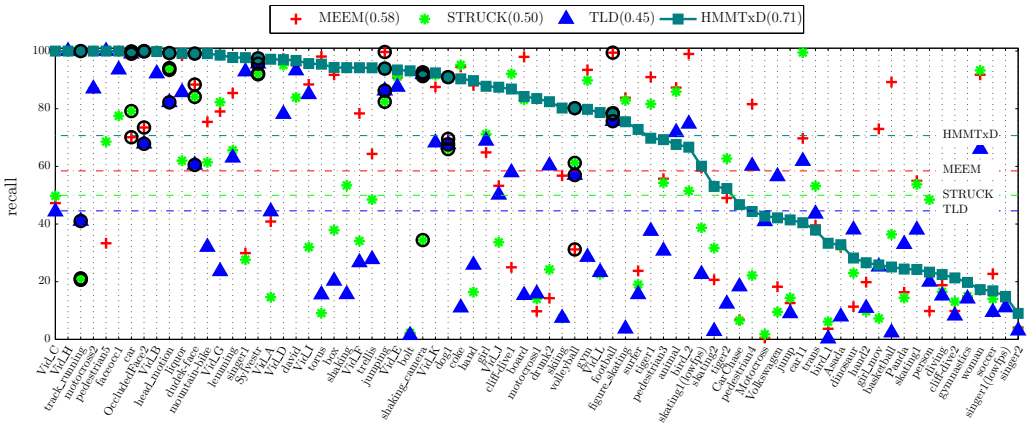


Figure 9: Evaluation of state-of-the-art trackers on the TV77 dataset in terms of recall, i.e. number of correctly tracked frames. The average recall is shown by the dashed lines (precise number is in the legend). Black circles mark grayscale sequences. The sequences are ordered by HMMTxD performance.

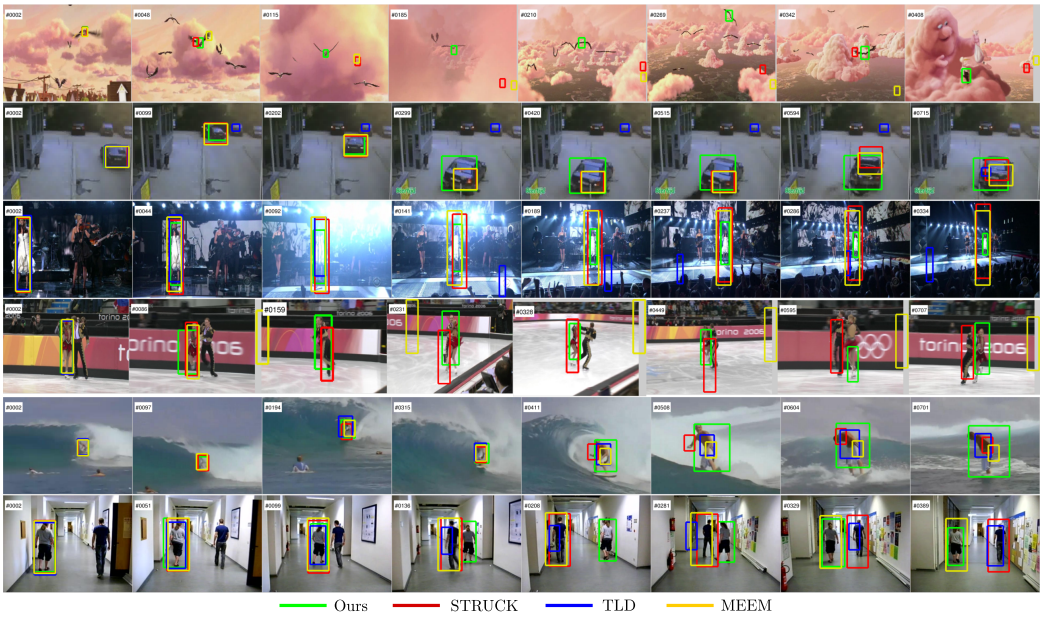


Figure 10: Qualitative comparison of the state-of-the-art trackers on challenging sequences from the TV77 dataset (from top *bird_1, drunk2, singer1, skating2, surfer, Vid_J*).

7. Conclusions

A novel method called HMMTxD for fusion of multiple trackers has been proposed. The method utilizes an on-line trained HMM to estimate the states of the individual trackers and to fuse a different types of observables provided by the trackers. The HMMTxD outperforms its constituent parts (FoT, ASMS, KCF, Detector and its combinations) by a large margin and shows the efficiency of the HMM with combination of three trackers.

HMMTxD outperforms all methods included in the CVPR2013 benchmark and perform favorably against most recent state-of-the-art tracker. The HMMTxD also outperforms all method of the VOT2013 benchmark in accuracy, while maintaining very good robustness, and ranking in the first place in overall ranking. Experiments conducted on a diverse dataset TV77 show that the HMMTxD outperforms state-of-the-art MEEM, STRUCK and TLD methods, which are similar in design, by a large margin. The processing speed of the HMMTxD is $5 - 10$ frames per second on average, which is comparable with other complex tracking methods.

Acknowledgements

The research was supported by the Czech Science Foundation Project GACR P103/12/G084 and by the Technology Agency of the Czech Republic project TE01020415 (V3C – Visual Computing Competence Center).

Appendix A. Forward-Backward Procedure for Modified Baum-Welch Algorithm

Let us assume the HMM with N possible states $\{s_1, s_2, \ldots, s_N\}$, the matrix of state transition probabilities $A = \{a_{ij}\}_{i,j=1}^{N}$, the vector of initial state probabilities $\pi = (1, 0, 0, \ldots, 0)$, the initial state $s_1 = (1, 1, \ldots, 1)$, a sequence of observations $\mathbb{X} = \{X_t\}_{t=1}^{T}$, $X_t \in R^m$ and $F = \{f_i(x)\}_{i=1}^{N}$ the system of conditional probability densities of observations conditioned on $S_t = s_i$.

Let $0 = t_0 < t_1 < t_2 \ldots < t_K \leq T$ be a sequence of detection times, $\mathbb{S} = \{S_{t_k} = s_{i_k}, \{t_k\}_{k=1}^{K}\}$ be observed states of Markov chain, marked by the detector, and $S_{t_k+1} = s_1$ for $0 \leq k \leq K$.

The forward variable for the Baum-Welch algorithm is defined as follows. Let $1 \leq i \leq N, 1 \leq k \leq K, t_{(k-1)} < t \leq t_k$ and

$$\alpha_t(i) = P(X_{t_{(k-1)}+1}, \ldots, X_t, S_t = s_i|\lambda) \text{ then} \tag{A.1}$$

$$\alpha_{t_{(k-1)}+1}(1) = f_1(X_{t_{(k-1)}+1}), \tag{A.2}$$

$$\alpha_{t_{(k-1)}+1}(i) = 0 \text{ for } i \neq 1 \tag{A.3}$$

and for $t_{(k-1)} < t < t_k$

$$\alpha_{(t+1)}(i) = \sum_{j=1}^{N} \alpha_t(j) a_{ji} f_i(X_{t+1}), \tag{A.4}$$

$$P(S_t = s_i | X_1, \ldots X_t, S_{t_1}, S_{t_2}, \ldots, S_{t_{(k-1)}}, \lambda) = \frac{\alpha_t(i)}{\sum_{j=1}^{N} \alpha_t(j)}. \tag{A.5}$$

For $t_K < t < T$ the forward variable is in principle the same as above with $t_{(k-1)} = t_K$. So

$$P(X_{t_K+1}, \ldots, X_T | \lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{A.6}$$

$$P(\mathbb{X}, \mathbb{S} | \lambda) = \prod_{k=1}^{K} \alpha_{t_k}(i_k) * \sum_{i=1}^{N} \alpha_T(i) \quad \text{where } S_{t_k} = s_{i_k}. \tag{A.7}$$

The backward variable for $t_{(k-1)} < t < t_k$ is

$$\beta_t(i) = P(X_{t+1}, \ldots, X_{t_k}, S_{t_k} | S_t = s_i, \lambda), \tag{A.8}$$

where $\beta_{t_k}(i_k) = 1$ and $\beta_{t_k}(i) = 0$ for $i \neq i_k$ and

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} f_j(X_{t+1}) \beta_{t+1}(j). \tag{A.9}$$

For $t_K < t < T$ the backward variable is in principle the same as above where $\beta_T(i) = 1$ for $1 \leq i \leq N$.

Given the forward and backward variables, we get the following probabilities, that are used to update parameters of HMM. For $0 < t < T$ and $t \neq t_k, 1 \leq k \leq K$

$$P(S_t = s_i, S_{t+1} = s_j | \mathbb{X}, \mathbb{S}, \lambda) = \tag{A.10}$$

$$\frac{\alpha_t(i) a_{ij} f_j(X_{t+1}) \beta_{(t+1)}(j)}{\sum_{k=1}^{N} \sum_{l=1}^{N} \alpha_t(k) a_{kl} f_l(X_{t+1}) \beta_{t+1}(l)} \tag{A.11}$$

and for $0 < t \leq T$

$$P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)}. \tag{A.12}$$

The final equation for the update of transition probabilities A of HMM is as follows.

$$\hat{a}_{ij} = \frac{\text{expected number of transitions from state } s_i \text{ to state } s_j}{\text{expected number of transitions from state } s_i} \tag{A.13}$$

$$= \frac{\sum_{(t=1 \text{ and } t \neq t_k, 1 \leq k \leq K)}^{T-1} P(S_t = s_i, S_{t+1} = s_j|\mathbb{X}, \mathbb{S}, \lambda)}{\sum_{(t=1 \text{ and } t \neq t_k, 1 \leq k \leq K)}^{T-1} P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}. \tag{A.14}$$

Appendix B. Generalized Method of Moments

For a simplification let us assume HMM with one-dimensional observed random variables $\{X_t\}_{t=1}^{+\infty}, X_t \in R$. The sequence $\{X_t - E(X_t|X_1, X_2, \ldots, X_{t-1})\}_{t=1}^{+\infty}$ is a martingale difference series where

$$E(X_t|X_1, X_2, \ldots, X_{t-1}) = \sum_{i=1}^{N} E(X_t|X_1, X_2, \ldots, X_{t-1}, S_t = i)P(S_t = i) \tag{B.1}$$

$$= \sum_{i=1}^{N} E(X_t|S_t = i)P(S_t = i). \tag{B.2}$$

Under the assumption that $\{X_t\}_{t=1}^{+\infty}$ are uniformly bounded random variables i.e. $|X_t| < c, c \in (0, +\infty)$ for all $t \geq 1$, the strong law of large numbers for a sum of martingale differences can be used(see Theorem 2.19 in [33]). So

$$\lim_{T \to +\infty} \frac{1}{T} \sum_{t=1}^{T} [X_t - \sum_{i=1}^{N} E(X_t|S_t = i)P(S_t = i)] = 0 \text{ almost surely.} \tag{B.3}$$

Let us denote $\mu_i = E(X_t|S_t = i)$ for $1 \leq t \leq T$ and $\hat{\mu}_i$ the estimate of μ_i based on the modified method of moments. The estimate $\hat{\mu}_i$ is a solution of a following equation w.r.t. μ_i

$$\frac{1}{T}\sum_{t=1}^{T} X_t = \frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{N} \mu_i P(S_t = i). \tag{B.4}$$

Having one equation for N unknown variables $\mu_i, 1 \leq i \leq N$ it is necessary to add some constrains to get a unique solution. We propose to minimize

$$\sum_{t=1}^{T}\sum_{i=1}^{N} (X_t - \mu_i)^2 P(S_t = i), \tag{B.5}$$

w.r.t. $\mu_i, 1 \leq i \leq N$ giving

$$\hat{\mu}_i = \frac{\sum_{t=1}^{T} X_t P(S_t = s_i)}{\sum_{t=1}^{T} P(S_t = s_i)} \tag{B.6}$$

which satisfy the moment equation (B.4). The same way of reasoning can be used for higher moments of $\{X_t\}_{t=1}^{T}$. For example using $\{(X_t)^2\}_{t=1}^{T}$ we get estimates $\hat{\sigma}_i^2$ for $\sigma_i^2 = var(X_t|S_t = i)$ for $1 \leq t \leq T$,

$$\hat{\sigma}_i^2 = \frac{\sum_{t=1}^{T} (X_t - \hat{\mu}_i)^2 P(S_t = s_i)}{\sum_{t=1}^{T} P(S_t = s_i)}. \tag{B.7}$$

In the HMMTxD m-dimensional observed random variables $X_t = (X_t^1, X_t^2, \ldots, X_t^m)$ are assumed, each of them having beta- distribution and being conditionally independent. There are well-known relations for a mean value EX and a variance $varX$ of a random variable X having beta distribution and its shape parameters (p, q)

$$p = EX\left(\frac{EX(1-EX)}{varX} - 1\right) \tag{B.8}$$

and

$$q = (1-EX)\left(\frac{EX(1-EX)}{varX} - 1\right). \tag{B.9}$$

Using the modified method of moments gives

$$\hat{\mu}_i^j = \frac{\sum_{t=1}^{T} X_t^j P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}{\sum_{t=1}^{T} P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)} \tag{B.10}$$

and

$$(\hat{\sigma}_i^j)^2 = \frac{\sum_{t=1}^{T} (X_t^j - \hat{\mu}_i^j)^2 P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}{\sum_{t=1}^{T} P(S_t = s_i|\mathbb{X}, \mathbb{S}, \lambda)}. \tag{B.11}$$

Then

$$\hat{p}_i^j = \hat{\mu}_i^j \left(\frac{\hat{\mu}_i^j(1-\hat{\mu}_i^j)}{(\hat{\sigma}_i^j)^2} - 1 \right) \tag{B.12}$$

and

$$\hat{q}_i^j = (1-\hat{\mu}_i^j) \left(\frac{\hat{\mu}_i^j(1-\hat{\mu}_i^j)}{(\hat{\sigma}_i^j)^2} - 1 \right). \tag{B.13}$$

If we assume in our model $\lambda = (A, F)$ that for some $\{(i_r, j_r) \in \{1, 2, \ldots, N\} \times \{1, 2, \ldots, m\} : p_{i_r}^{j_r} = p, q_{i_r}^{j_r} = q\}_{r=1}^R$ then

$$\hat{p} = \hat{\mu} \left(\frac{\hat{\mu}(1-\hat{\mu})}{\hat{\sigma}^2} - 1 \right) \tag{B.14}$$

and

$$\hat{q} = (1-\hat{\mu}) \left(\frac{\hat{\mu}(1-\hat{\mu})}{\hat{\sigma}^2} - 1 \right) \tag{B.15}$$

where

$$\hat{\mu} = \frac{\sum_{r=1}^R \sum_{t=1}^T X_t^{j_r} P(S_t = s_{i_r} | \mathbb{X}, \mathbb{S}, \lambda)}{\sum_{r=1}^R \sum_{t=1}^T P(S_t = s_{i_r} | \mathbb{X}, \mathbb{S}, \lambda)} \tag{B.16}$$

and

$$\hat{\sigma}^2 = \frac{\sum_{r=1}^R \sum_{t=1}^T (X_t^{j_r} - \hat{\mu})^2 P(S_t = s_{i_r} | \mathbb{X}, \mathbb{S}, \lambda)}{\sum_{r=1}^R \sum_{t=1}^T P(S_t = s_{i_r} | \mathbb{X}, \mathbb{S}, \lambda)}. \tag{B.17}$$

References

[1] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, ACM Computing Surveys, 2006. 2

[2] A. Smeulder, D. Chu, R. Cucchiara, S. Calderara, A. Deghan, M. Shah, Visual tracking: an experimental survey, IEEE Transactions on Pattern Analysis and Machine Intelligence (2013). 2

[3] J. Kwon, K. M. Lee, Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling., in: Computer Vision and Pattern Recognition, 2009, pp. 1208–1215. 2

[4] M. Godec, P. M. Roth, H. Bischof, Hough-based tracking of non-rigid objects, in: International Conference on Computer Vision, 2011. 2

[5] L. Cehovin, M. Kristan, A. Leonardis, Robust visual tracking using an adaptive coupled-layer visual model, IEEE Transactions on Pattern Analysis and Machine Intelligence 35 (4) (2013) 941–953. 2

[6] H. Grabner, J. Matas, L. Van Gool, P. Cattin, Tracking the invisible: Learning where the object might be, in: Computer Vision and Pattern Recognition, 2010, pp. 1285–1292. 2

[7] X. Zhou, Y. Lu, Abrupt motion tracking via adaptive stochastic approximation Monte Carlo sampling, in: Computer Vision and Pattern Recognition, 2010, pp. 1847–1854. 2

[8] F. Pernici, A. D. Bimbo, Object tracking by oversampling local features, IEEE Transactions on Pattern Analysis and Machine Intelligence 99 (PrePrints) (2013) 1. 2, 10

[9] Z. Kalal, K. Mikolajczyk, J. Matas, Tracking-learning-detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (7) (2012) 1409–1422. 2, 18

[10] J. Santner, C. Leistner, A. Saffari, T. Pock, H. Bischof, PROST Parallel Robust Online Simple Tracking, in: Computer Vision and Pattern Recognition, San Francisco, CA, USA, 2010. 2

[11] J. Zhang, S. Ma, S. Sclaroff, MEEM: robust tracking via multiple experts using entropy minimization, in: Proc. of the European Conference on Computer Vision, 2014. 3, 16, 17, 18

[12] N. Wang, D. yan Yeung, Ensemble-based tracking: Aggregating crowdsourced structured time series data, in: T. Jebara, E. P. Xing (Eds.), Proceedings of the 31st International Conference on Machine Learning, 2014, pp. 1107–1115. 3, 17

[13] C. Bailer, A. Pagani, D. Stricker, A superior tracking approach: Building a strong tracker through fusion, in: European Conference on Computer Vision, Lecture Notes in Computer Science, 2014. 3, 12

[14] Y. Yuan, H. Yang, Y. Fang, W. Lin, Visual object tracking by structure complexity coefficients, Multimedia, IEEE Transactions on 17 (8) (2015) 1125–1136. 3

[15] J. H. Yoon, D. Y. Kim, K.-J. Yoon, Visual tracking via adaptive tracker selection with multiple features, in: Proceedings of the 12th European Conference on Computer Vision - Volume Part IV, ECCV'12, 2012, pp. 28–41. 3

[16] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. P. Pflugfelder, G. Fernández, G. Nebehay, F. Porikli, L. Cehovin, A novel performance evaluation methodology for single-target trackers, arXiv 2015 abs/1503.01313.
URL http://arxiv.org/abs/1503.01313 4

[17] A. Gupta, S. Nadarajah, Handbook of Beta Distribution and Its Applications, Statistics: A Series of Textbooks and Monographs, 2004. 5, 9

[18] L. Rabiner, A tutorial on hidden markov models and selected applications in speech recognition, Proceedings of the IEEE 77 (2) (1989) 257–286. 6, 9, 10

[19] L. E. Baum, T. Petrie, G. Soules, N. Weiss, A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains, The Annals of Mathematical Statistics 41 (1) (1970) 164–171. 6, 7, 9

[20] A. P. Dempster, N. M. Laird, D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistical Society, series B 39 (1) (1977) 1–38. 7, 10

[21] D. G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision 60 (2) (2004) 91–110. 11

[22] E. Rublee, V. Rabaud, K. Konolige, G. Bradski, Orb: An efficient alternative to sift or surf, in: International Conference on Computer Vision, ICCV '11, Washington, DC, USA, 2011, pp. 2564–2571. 11

[23] R. Ortiz, Freak: Fast retina keypoint, in: Conference on Computer Vision and Pattern Recognition, Washington, DC, USA, 2012, pp. 510–517. 11

[24] M. Kristan, J. Perš, V. Sulic, S. Kovacic, A graphical model for rapid obstacle image-map estimation from unmanned surface vehicles, in: Asian Conference on Computer Vision. Accepted, to be published., 2014. 11

[25] T. Vojir, J. Matas, The enhanced flock of trackers, in: Registration and Recognition in Images and Videos, Vol. 532 of Studies in Computational Intelligence, 2014, pp. 113–136. 13, 14

[26] T. Vojir, J. Noskova, J. Matas, Robust scale-adaptive mean-shift for tracking, in: Image Analysis, Vol. 7944 of Lecture Notes in Computer Science, 2013, pp. 652–663. 13

[27] J. F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, IEEE Transactions on Pattern Analysis and Machine Intelligence 37 (3) (2015) 583–596. 13

[28] B. D. Lucas, T. Kanade, An iterative image registration technique with an application to stereo vision, in: International Joint Conference on Artificial Intelligence, 1981. 13

[29] Y. Wu, J. Lim, M.-H. Yang, Online object tracking: A benchmark, in: Computer Vision and Pattern Recognition, 2013, pp. 2411–2418. 15, 17

[30] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, et al., The visual object tracking vot2013 challenge results, in: The IEEE International Conference on Computer Vision (ICCV) Workshops, 2013. 17, 18

[31] J. Xiao, R. Stolkin, A. Leonardis, An enhanced adaptive coupled-layer lgtracker++, in: Computer Vision Workshops (ICCVW), 2013 IEEE International Conference on, 2013, pp. 137–144. 18

[32] S. Hare, A. Saffari, P. H. S. Torr, Struck: Structured output tracking with kernels, in: International Conference on Computer Vision, 2011, pp. 263–270. 18

[33] P. Hall, C. Heyde, Martingale limit theory and its application, Probability and mathematical statistics, Academic Press, 1980. 22