ROBUST ONLINE FACE TRACKING-BY-DETECTION

Francesco Comaschi[1]*, Sander Stuijk*[1]*, Twan Basten*[1,2]*, Henk Corporaal*[1]

[1]Eindhoven University of Technology, The Netherlands
[2]TNO Embedded Systems Innovation, Eindhoven, The Netherlands
{f.comaschi, s.stuijk, a.a.basten, h.corporaal}@tue.nl

ABSTRACT

The problem of online face tracking from unconstrained videos is still unresolved. Challenges range from coping with severe online appearance variations to coping with occlusion. We propose RFTD (Robust Face Tracking-by-Detection), a system which combines tracking and detection into a single framework to robustly track a face from unconstrained videos. RFTD is based on the idea that adaptive and stable algorithmic components can complement each other in the task of online tracking. An online Structured Output SVM (SO-SVM) is combined with an offline trained face detector to break the self-learning loop typical in tracking. In turn, the face detector is supervised by a Deformable Part Model (DPM) landmark detector to asses the reliability of the face detection output. Extensive evaluation shows that RFTD delivers consistently good tracking performances across different scenarios, i.e., high mean success rate and lowest standard deviation across benchmark videos.

Index Terms— Face tracking, tracking-by-detection, structured output SVM, deformable models

1. INTRODUCTION

Online tracking of human faces in a video is a fundamental task in several computer vision applications ranging from video-surveillance to online gaming. In particular, the detection and tracking of human faces is an enabling step required for higher-level tasks such as face recognition and behavior understanding.

Despite the increasing interest in this field [1, 3, 12], robust online face tracking under real-world conditions still represents an unresolved problem because of a variety of challenges involved such as appearance variations, illumination changes and occlusions. The robustness of a tracker expresses its capability of not losing track of the target regardless of the complexity of the tracking scenario [11, 14]. It can be assessed by evaluating the mean and standard deviation of the success rate across videos characterized by different challenges. A robust online face tracker should be capable of: (i) dealing with arbitrary complex video streams; (ii) adapt to the target appearance variations; (iii) run continuously for long sequences without drifting or accumulating errors. The last two requirements can be interpreted in terms of the stability-plasticity dilemma [13]: from one side a robust tracker should be adaptive enough to follow a target undergoing appearance and viewpoint changes, on the the other hand it needs to be stable in order to avoid drifting.

A possible approach to face tracking lying on the stability end is to simply apply one or multiple rigid face detectors at each frame, and to associate the detection responses into tracks at a later stage [12]. However, most face detectors are trained offline and cannot adapt to online appearance variations of the target. A mitigation to the lack of plasticity is provided by Deformable Part Model (DPM) detectors [23], which allow for some flexibility by associating a deformation cost to the relative position of facial landmarks. However, also in this case it is very difficult to provide the detector with enough training samples to cover the full spectrum of appearance variations that will occur at run-time [10]. Towards the plasticity end, an approach to online tracking which has proven to be very effective in capturing the target appearance changes is tracking-by-detection [7], where a target-specific classifier is trained online. However, these trackers suffer from the opposite problem: the tracker provides its own samples for online training (self-learning), thus incurring into the risk of wrong updates and consequent drifting [13].

These considerations are at the basis of several works [5, 10, 13, 15] which provide online tracking solutions either by combining multiple trackers or by integrating a detection and a tracking component together. However, these solutions are not robust across a wide range of videos.

In this paper we propose RFTD (Robust Face Tracking-by-Detection), a robust online face tracker which is capable of effectively tracking a face in unconstrained videos by combining complementary components operating at different levels of the stability-plasticity spectrum. On the adaptive end, we use a Structured Output Support Vector Machine (SO-SVM) tracker because of its effectiveness in finding the target location frame-by-frame. In order to mitigate the drifting associated with the self-learning process, we supervise the tracker's updating procedure with a rigid offline trained face detector. Finally, to prevent the risk of wrong updates caused by the face detector's false positives, we supervise the detector candidate samples with a DPM-based facial landmark detector, whose confidence is used to assess the samples reliability before updating.

We prove the effectiveness of RFTD by extensive evaluation on 15 different benchmark videos in unconstrained scenarios, and we provide a quantitative comparison with several tracking methods recently proposed in literature. RFTD consistently performs well across different scenarios and we prove its robustness against perturbations in the tracker initialization.

This work was supported in part by the COMMIT program under the SenSafety project.

2. RELATED WORK

Recent progress in face detection has led to association-based face tracking approaches, where one or many face detectors are applied at each frame and detection responses are associated into tracks based on some affinity measures [12]. These approaches rely entirely on the offline training process and cannot adapt to a target undergoing appearance variations. Compared to rigid face detectors, DPM-based approaches have advantages in handling certain degrees of appearance variations and have been successfully applied to face detection and facial landmarks localization [16, 23]. In RFTD we use the effectiveness of DPM in capturing facial landmarks to improve the robustness of our tracking framework.

Facial landmarks have been widely employed in non-rigid face tracking scenarios [1, 22], where the aim is to continuously monitor the location of the facial landmarks. These methods are mostly intended for face-analysis applications where a single target is clearly visible in front of the camera most of the time, but they have been rarely applied in unconstrained videos characterized by extreme appearance changes, abrupt camera movements, occlusions and scale variations. When aiming at tracking visual objects in challenging real-world scenarios, an approach which has recently proved to be very effective is tracking-by-detection [5, 6, 7], which attempts to learn a classifier to distinguish a target object from its local background. One of the greatest challenges related to tracking-by-detection is the sampling process, where the tracker has to select its own training samples. Slight inaccuracies during sampling can lead to wrong updates of the tracker thus leading to further tracking inaccuracies.

An attempt to mitigate this problem is represented by the SO-SVM framework [7], where the acquisition of positively and negatively labeled data is circumvented by integrating the labeling process into the learner itself. However, the framework proposed in [7] still suffers from the drifting associated to the self-learning loop. In order to break this loop, RFTD combines an offline trained face detector with a SO-SVM tracker, thus alleviating the drifting problem without losing the capability to adapt to appearance variations of the target.

In literature, a combination of adaptive and stable components for tracking has already been used. In [3], a detection and a recognition module are integrated with an online face tracker in order to provide the capability to reacquire the target after full occlusion or leaving the field of view. The system described in [3] relies on information provided by other targets in the scene, and results are reported only on videos containing multiple faces. RFTD does not rely on the presence of similar targets in the scene and it is specifically targeted at improving the robustness of online tracking, without a module for re-acquisition or recognition, which could be integrated as a future extension of the current system.

A very popular approach integrating detection and tracking is the TLD framework [10], where an optical flow tracker is combined with an online trained Random Ferns detector. The TLD framework offers re-acquisition capabilities and is particularly suited for long video sequences. An extension to the TLD framework is represented by the Context Tracker [5], which exploits the idea of distracters and supporters to improve the tracking performance. RFTD does not include a module for re-acquisition and does not make explicit use of auxiliary information to improve the tracking robustness. It nevertheless compares favorably against these methods, as shown in Sec. 4.

Two other related approaches are described in [13, 15]. In [13], the authors combine a moderately adaptive Random Forest tracker with a more stable template-matching tracker and an optical flow component. RFTD specifically targets the problem of online face tracking and the stable component of our framework is represented by an offline trained face detector. Moreover, in [13] the update of the adaptive tracker is based on bounding box overlap, while in RFTD the update process is supervised by the DPM-detector confidence.

In [15], a recognizer component is introduced whose training samples are provided by an online-boosted tracker and validated by an offline trained detector. Also in our method we include an offline trained classifier to guarantee a stable component in the framework, but we use a SO-SVM tracker as the main tracking component and a DPM landmark detector to supervise the online learning process, thus achieving better tracking results as shown in Sec. 4.

3. TRACKING FRAMEWORK

Our goal is the robust online tracking of a single face from unconstrained videos. A robust face tracker needs to adapt to fast appearance changes while being able to recover from drift in arbitrary complex video streams. To reach this goal, we combine complementary components with different adaptivity capabilities into a single framework.

3.1. Overview of RFTD

The block diagram of RFTD is depicted in Fig. 1. The components of the framework are characterized as follows: *tracker* predicts the face location frame-by-frame by evaluating the confidence of an online-learned SO-SVM. The tracker is effective in capturing the appearance variations of the target, but the self-learning loop of the SO-SVM can easily drift in case of wrong updates. Therefore, the tracker provides the Region Of Interest (ROI) to an offline-learned *face detector* which supervises the learning process by searching for possible face candidates around the current target location. The aim of this supervision flow is to check if the tracker is drifting from the real target. In order to avoid false positives caused by the face detector, i.e., location around the current target that the detector erroneously classifies as faces, the face candidates are provided to a DPM *landmark detector*, which assess the reliability of the provided face candidates and estimates the best candidate to be employed for the tracker update.

Fig. 2 depicts a sequence of three frames and the corresponding support vectors learned by the online tracker. In frame 459, the tracker (green rectangle) starts drifting because the target undergoes severe appearance and illumination variation. From the bottom row, we can see how the drifting affects the online learning process by providing wrong samples to the online SO-SVM. In frame 475, the face detector erroneously classifies a region within the ROI (red rectangle) as the current location of the target (blue rectangle). However, this region is discarded by the landmark detector as false positive. In frame 486, the face detector finds two possible candidates for the re-initialization of the tracker. One of them

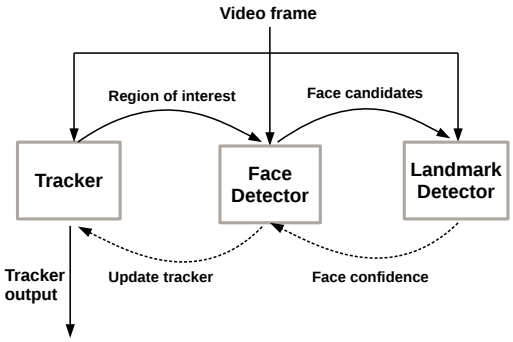


Fig. 1: Block diagram of RFTD. The dashed lines represent the supervision chain.

is correctly validated by the landmark detector (red points) and the candidate is provided to the tracker to re-initialize the online SVM (bottom row). The components of RFTD are described below in more detail.

Fig. 2: Example frames of RFTD (top) and corresponding support vectors of the online SO-SVM tracker (bottom). The tracker output is green, the ROI is red while the candidates from the face detector are blue. The support vectors are either positive (green) or negative (red).

3.2. Tracker

The tracking component of RFTD is based on SO-SVM tracking, which treats the tracking problem as one of structured output prediction [7]. In SO-SVM tracking, a kernelized structured output SVM is learned online to provide adaptive tracking. The tracker maintains an estimate of the position $\mathbf{p}$ of a 2D bounding box within frame $\mathbf{f}_t$, where $t = 1, \ldots, T$ is time. Given a bounding box position $\mathbf{p}$, Haar features are extracted from an image patch within the bounding box: $\mathbf{x}_t^{\mathbf{p}} \in \mathcal{X}$. The objective of the tracker at time t is to estimate a 2D translation of the target $\mathbf{y}_t \in \mathcal{Y}$, where $\mathcal{Y} = \{(\Delta u, \Delta v) |\ \Delta u^2 + \Delta v^2 < r^2\}$, r being a search radius. If $\mathbf{p}_{t-1}$ is the target position at time $t-1$, the 2D position of the target at time t can be found as $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \mathbf{y}_t$ which is given by $(u_t, v_t) = (u_{t-1}, v_{t-1}) + (\Delta u, \Delta v)$. In structured SVM a discriminant function $F : \mathcal{X} \times \mathcal{Y}$ is introduced to find the translation function according to:

$$\mathbf{y}_t = f(\mathbf{x}_t^{\mathbf{p}_{t-1}}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}_t^{\mathbf{p}_{t-1}}, \mathbf{y}) \qquad (1)$$

where F is restricted to be in the form $F(\mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \mathbf{\Phi}(\mathbf{x}, \mathbf{y}) \rangle$, $\mathbf{\Phi}(\mathbf{x}, \mathbf{y})$ being a joint kernel map. F measures the compatibility between $(\mathbf{x}, \mathbf{y})$ pairs and gives a high score to those which are well matched. In this way, F can be learned from a set of example pairs $\{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$ by minimizing a convex objective function. The discriminant function F is defined by a set of support vectors $(\mathbf{x}_i, \mathbf{y}) \in \mathcal{S}$, $\mathbf{x}_i$ being a support pattern. In Fig. 2 we show examples of the support vector set $\mathcal{S}$ during tracking.

Since evaluating $F(\mathbf{x}, \mathbf{y})$ requires the computation of inner products between $(\mathbf{x}, \mathbf{y})$ and each support vector, both the computational and storage costs grow linearly with the number of support vectors. In order to limit these costs, we restrict the number of support vectors to $B = 50$, employing the method proposed in [7]. In practice, in RFTD the use of a face detector to re-initialize the tracker prevents an excessive increase of the number of support vectors.

The online learning component of the tracker relies on the output provided by the tracker itself, since the current tracker location $(\mathbf{x}_t^{\mathbf{P}^t}, \mathbf{y})$ is supplied to the prediction function as a positive labeled sample, while other samples are weighted according to a loss function based on bounding box overlap [7]. In this way, a small error in the tracker output can lead to the wrong labeling of the samples and consequently cause the tracker to drift (see Fig. 2). To avoid the self-learning loop, we introduce an offline trained face detector which supervises the update of the online tracker.

3.3. Face detector

In RFTD we use the AdaBoost classifier developed by Viola and Jones [17] since it provides a good trade-off between robustness and computational efficiency. In particular, at each frame of the video we take a padded area around the current target bounding box (75% of current size) and we scan three face detectors in parallel (one for frontal and nearly frontal and two for profile and half-profile views). The sliding-window search mechanism and the cascade structure of the face detector [17] allow to rapidly scan for faces at multiple scales, thus helping the SO-SVM tracker in tracking across rapid scale variations.

As any other offline trained detector, the AdaBoost detector is subject to two different kinds of errors, i.e., false negatives (miss the target) and false positives (background regions classified as faces) [10]. In RFTD, the frame-by-frame tracking of the SO-SVM compensates for the frame in which the face detector is not capable of finding the face location (false negatives). However, we still need a mechanism to prevent the detector from updating the tracker with false positives. Examples of such false positives which would lead to wrong updates are shown in Fig. 2. To solve this problem, we use the confidence provided by the DPM landmark detector in order to asses the reliability of the face candidates generated by the face detector.

3.4. Landmark detector

Instead of coupling face detection and landmark localization into a single module like in [23], we separate the two tasks in order to have independent modules that can collaborate in a supervision chain (see Fig. 1). The parameters of the DPM model are learned offline with a SO-SVM framework [16]. Similarly as in the tracking module, the SO-SVM framework offers the advantage over conventional SVM to be

able to directly optimize a complex output function, in this case the average localization error of the landmarks. More details about the learning procedure can be found in [16]. In order to be able to cope with multiple views, the viewing angle (yaw angle) is discretized in five different views: $\Phi = \{-profile, -half\text{-}profile, frontal, half\text{-}profile, profile\}$. For each view a tree-structured undirected graph $G = (V, E)$ is considered where V is a finite set of vertices representing the landmarks and $E \subset \binom{V}{2}$ is a set of edges between pairs of landmarks whose positions are related. Example graphs can be found in [16]. The chosen tree-structure, and consequently the organization of the edges, is motivated by efficiency reasons concerning the off-line learning of the landmark detector's parameters. Let I be the detector output and $\mathbf{l} = (\mathbf{l}_1, \ldots, \mathbf{l}_{|V|-1})$ be a configuration of landmark locations (i.e., the x, y coordinates of landmarks, $\mathbf{l}_i = (x_i, y_i)$) and finally let $\mathbf{u}$ be the vector of parameters composed of parameters $\mathbf{u}_i^{\phi q}$ and $\mathbf{u}_{ij}^{\phi g}$ associated with the unary and pairwise potentials, respectively. Then, the scoring function of the landmark detector on the given detector output I can be defined as follows:

$$f(I, \phi, \mathbf{l}; \mathbf{u}) = \sum_{i \in V} q_i^{\phi}(\mathbf{l}_i, I; \mathbf{u}_i^{\phi q}) + \sum_{(i,j) \in E} g_{ij}^{\phi}(\mathbf{l}_i, \mathbf{l}_j; \mathbf{u}_{ij}^{\phi g}) \tag{2}$$

The first part of the scoring function, denoted as the *appearance model*, is composed by unary potentials q_i^{ϕ} which measure the quality of the fit of individual landmarks to the image. It is a linearly parametrized function $q_i^{\phi}(\mathbf{l}_i, I; \mathbf{u}_i^{\phi q}) = \langle \mathbf{u}_i^{\phi q}, \mathbf{\Psi}_i^{\phi q}(I, \mathbf{l}_i) \rangle$, where $\mathbf{\Psi}_i^{\phi q}$ is a Sparse Local Binary Pattern (S-LBP) features descriptor of a patch from the image I around the position $\mathbf{l}_i$, while $\mathbf{u}_i^{\phi q}$ is the weight vector learned with the SO-SVM framework [16].

The second part, denoted as *deformation cost*, is composed of pair-wise potentials $g_{ij}^{\phi}(\mathbf{l}_i, \mathbf{l}_j; \mathbf{u}_{ij}^{\phi g})$ which correspond to the likeliness of the mutual position of the connected pair of landmarks. It is also a linearly parametrized function $g_{ij}^{\phi}(\mathbf{l}_i, \mathbf{l}_j; \mathbf{u}_{ij}^{\phi g}) = \langle \mathbf{u}_{ij}^{\phi g}, \mathbf{\Psi}_{ij}^{\phi g}(\mathbf{l}_i, \mathbf{l}_j) \rangle$, where $\mathbf{\Psi}_{ij}^{\phi g}$ is defined as a quadratic function of the displacement vector, i.e.:

$$\begin{array}{rcl} \mathbf{\Psi}_{ij}^{\phi g}(\mathbf{l}_i, \mathbf{l}_j) & = & (dx, dy, dx^2, dy^2), \quad \text{where} \\ (dx, dy) & = & (x_j, y_j) - (x_i, y_i) \end{array} \tag{3}$$

Eq. (2) can be used to assess the confidence of the face candidate regions provided by the face detector, before providing them to the tracker for the online SO-SVM update.

4. EXPERIMENTAL RESULTS

We show the robustness of RFTD through an extensive evaluation on 15 challenging video sequences from the recent visual tracking benchmark described in [19, 20]. We provide a quantitative comparison with several visual trackers and we report different performance metrics. The experimental results show the robustness of the proposed method across different tracking scenarios as well as against perturbations in the initial conditions.

4.1. Benchmark and selected trackers

For our experiments we use the Visual Tracking Benchmark described in [19], since for this dataset several results from other algorithms are available for comparison. The benchmark contains 50 fully annotated sequences and includes the results from the evaluation of 29 different trackers. Since our framework specifically targets the problem of online face tracking, we evaluate our online face tracker on 15 video sequences[1] from the benchmark containing a face and we compare our results to the 6 trackers reported to be the best performing ones [18, 20], these being Struck [7], SCM [21], ASLA [9], CSK [8], L1APG [2] and OAB [6]. Moreover, we compare to three other trackers which present similarities with our framework, these being TLD [10], CXT [5] and BSBT [15]. This gives a total of 10 algorithms, including ours. The output provided by the chosen trackers on the selected video sequences can be downloaded from the benchmark website [18], together with the ground-truth.

In [19], the authors rank the video sequences from the most difficult to the easiest based on the overall performances of the evaluated trackers. In comparison to the existing trackers, it is interesting to note that our tracker performs particularly well on the 10 most challenging video sequences, where the majority of the other trackers report lower success rates. These videos are characterized by several challenges such as illumination variation, scale variation, occlusion, extreme appearance variation and fast motion.

4.2. Evaluation protocol

For quantitative analysis we measure the success rate based on the commonly used bounding box overlap criterion [14, 19]. Given the tracked bounding box $\mathbf{b}_{tr}$ and the ground-truth bounding box $\mathbf{b}_{gt}$, the overlap score is defined as $S(\mathbf{b}_{tr}, \mathbf{b}_{gt}) = \frac{|\mathbf{b}_{tr} \cap \mathbf{b}_{gt}|}{|\mathbf{b}_{tr} \cup \mathbf{b}_{gt}|}$, where $\cap$ and $\cup$ represent the intersection and the union of two regions, respectively, and $|\cdot|$ defines the number of pixels in a region [19]. To measure the performance on a video sequence, we count the number of successful frames whose overlap S is larger than a given threshold t_o. The success plot shows the ratios of successful frames at the thresholds varied from 0 to 1.

In [4, 11], robustness is evaluated by using the lowest theoretical threshold ($t_o = 0$) and by measuring only complete failures where the tracker has no overlap with the ground-truth. Such evaluation protocol requires manual intervention to reinitialize the trackers in case of failure, thus penalizing many of the trackers selected in this paper, which have the capability to automatically recover from tracking failure. Moreover, the choice of the threshold $t_o = 0$ favors the trackers which report large regions of the image as the target location.

In this paper we first rank the algorithms according to the commonly used Area Under Curve (AUC) which is the average of all success rates at different thresholds when the thresholds are evenly distributed. It can be proven that AUC score in this case actually matches the average overlap over the entire sequence [4]. However, for high values of the threshold t_o, the percentage of successful frames highly depends on the ex-

[1] The 15 video sequences ordered from the most difficult according to the tracking results reported in [19] are: *soccer, freeman4, freeman1, fleetface, freeman3, girl, jumping, trellis, david, boy, faceocc2, dudek, david2, mhyang, faceocc1*.

act definition of the ground-truth. Since the used benchmark is not specifically designed for face tracking, no clear guidelines have been provided in order to define the ground-truth around faces. The lack of clear guidelines in the definition of the ground-truth can be seen in Fig. 3, where in the top-row the ground-truth (yellow rectangles) includes the head of the target, while in the bottom-row it is more tightly drawn around the face. For this reason, we rank the algorithm also according to the commonly used fixed threshold of $t_o = 0.5$ [10, 13, 14], thus relying less on the exact definition of the ground-truth.

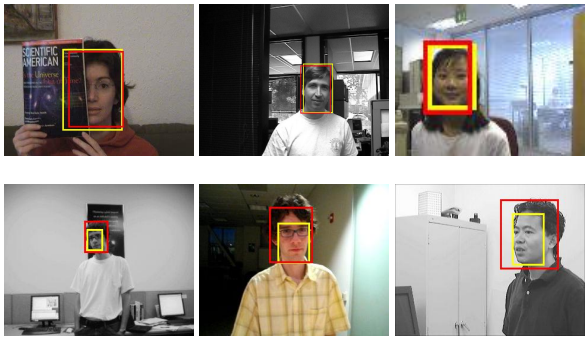


Fig. 3: Example video frames from the benchmark with ground-truth (yellow rectangles) and output from our method (red rectangles). Even though our tracker is on the target in all cases, a large value of t_o in the performance evaluation (right side of the success plots in Fig. 5) classifies the bottom row as wrong outputs.

In order to prove the robustness of the proposed method not only across different scenarios, but also against perturbations in the initialization of the tracker, we report the results from two different tests. The first test, referred to as One-Pass Evaluation (OPE) [19], is the conventional way of evaluating trackers, with initialization from the ground-truth in the first frame. The second test, referred to as Spatial Robustness Evaluation (SRE) [19], measures the robustness of a tracker against spatial perturbations of the initial bounding box, which is likely to happen in real-world scenarios where the tracker is usually initialized by a detector rather than by a human annotator. The SRE is measured by introducing 4 center shifts, 4 corner shifts and 4 scale variations of the initial bounding box as described in [19]. Thus we evaluated our tracker 12 times on each video for SRE. The reported result for both OPE and SRE is the average of 5 evaluations, because of the randomness involved in the tracking component of RFTD [7].

4.3. Quantitative results

In Fig. 4 we report the OPE obtained with our method as well as with the 9 other trackers over the 15 videos when considering the success rate for a fixed threshold $t_o = 0.5$. This figure shows the robustness of the considered trackers against different tracking scenarios.

In Fig. 4, each colored dot represents a different video from the benchmark, the red line is the mean success rate across all videos, while the blue and the red patches represent the Standard Deviation (SD) and the Standard Error of the Mean (SEM) respectively. From Fig. 4 we can see that the proposed

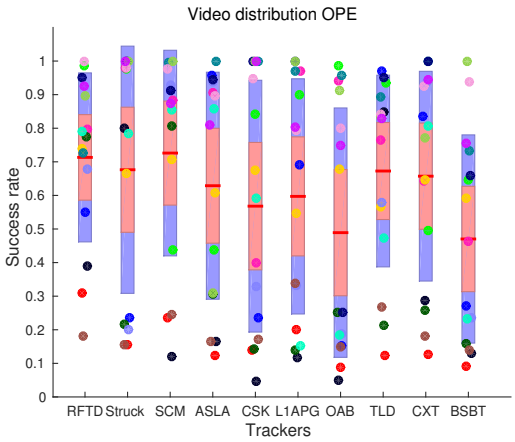


Fig. 4: Comparison of success rate across the benchmark videos for a fixed threshold $t_o = 0.5$.

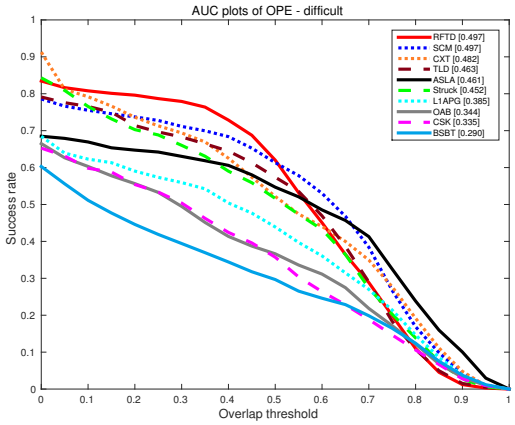
RFTD method performs well on the majority of the videos, achieving a success rate above 70% on 10 videos, while in only three cases it is below 50%. Similar performances are reached by the SCM tracker, which proves to be one of the best contenders when considering OPE. RFTD performs particularly well on the 10 most difficult video sequences, where the majority of the other trackers report lower success rates. This can be seen from Fig. 5a where we report the success plot for OPE of the 10 trackers evaluated over the 10 most difficult videos. From Fig. 5a we can see that our method is very robust in keeping track of the target (left-side of the plot) while it is less accurate in the exact definition of the bounding box (right-side of the plot). However, the success rate in the right side of the plot is highly influenced by the exact definition of the ground-truth, as shown in Fig. 3.

Table 1: Success rate (%) of tracking methods over *All* (15) and *Difficult* (10) videos, when considering the *AUC* or a fixed *Threshold* $t_o = 0.5$. Results are reported for *OPE* and *SRE*. Bold and underlined numbers indicate the column best and second best scores respectively.

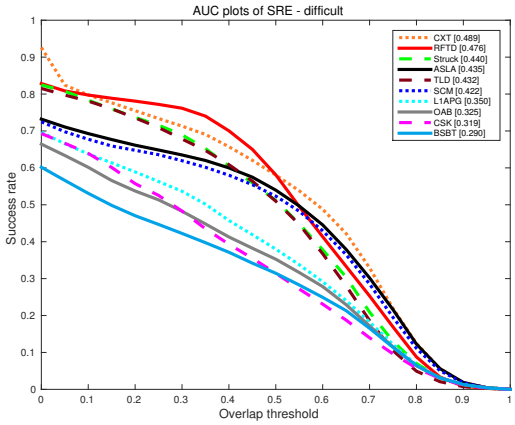
Tracker	OPE AUC All / Difficult	OPE Threshold All / Difficult	SRE AUC All / Difficult	SRE Threshold All / Difficult
RFTD	55.2 / **49.7**	71.3 / **62.0**	53.5 / 47.6	68.0 / **58.1**
Struck	55.9 / 45.2	67.6 / 51.7	52.9 / 44.0	65.8 / 51.2
SCM	**58.3** / **49.7**	**72.6** / 61.3	51.3 / 42.2	64.6 / 52.4
ASLA	53.8 / 46.1	62.9 / 54.7	51.2 / 43.5	63.5 / 54.0
CSK	48.0 / 33.5	56.8 / 52.2	44.6 / 31.9	51.6 / 31.5
L1APG	50.7 / 38.5	59.7 / 43.9	45.5 / 35.0	54.5 / 38.0
OAB	42.6 / 34.4	48.9 / 36.6	41.7 / 32.5	49.0 / 35.3
TLD	51.8 / 46.3	67.3 / 57.4	48.2 / 43.2	58.2 / 51.0
CXT	57.3 / 48.2	65.7 / 52.2	**55.7** / **48.9**	**69.1** / **58.1**
BSBT	40.6 / 29.0	47.0 / 29.7	38.3 / 29.0	45.1 / 31.5

In order to show the robustness of the proposed tracker against perturbation in the initial conditions, in Fig. 5b we report the SRE success plot of our method in comparison with the other 9 trackers over the 10 most challenging videos.

Fig. 5b shows that also in terms of SRE our method is one of the top contenders, being second only to CXT, which uses auxiliary context information from objects surrounding the target. The introduction of a module exploiting context information would be an interesting extension of our framework to



(a)



(b)

Fig. 5: Success plot of OPE (top) and SRE (bottom) for the 10 most challenging videos of the benchmark. The number in squared brackets is the AUC.

increase its robustness even further.

In Tab. 1 we report several performance metrics of the proposed method in comparison with the 9 selected trackers. From Tab. 1 we can see that RFTD is the best (bold number) or the second best (underlined number) in most cases. In comparison to the other trackers, RFTD performs well on the 10 most challenging sequences (*Difficult* videos), and is relatively more robust (*threshold*) than accurate (*AUC*). The best contenders are SCM, which combines a sparsity-based discriminative classifier and a sparsity-based generative model, and CXT, which includes a detection module and exploits context information to assist the tracking. These results suggest that the combination of independent modules assisting each other in a tracking framework is a successful idea.

The C++ implementation of our method is freely available on the website http://www.es.ele.tue.nl/video/. RFTD has an average throughput of 5 *fps* on an Intel Core i7 with a 3.07 GHz clock (single thread). The reported average tracking speed of CXT and SCM on a platform with similar computational power are 15.3 and 0.51 *fps* respectively [20].

5. CONCLUSIONS

We proposed RFTD, a framework for the robust online tracking of a face in unconstrained videos. Robust tracking is achieved by combining three modules with different adaptivity capabilities, i.e., an online SO-SVM tracker, an offline trained cascade face detector and a DPM-based landmark detector. We evaluated the robustness of the proposed framework across different tracking scenarios involving challenges such as fast motion, appearance variations, occlusion and light changes. Robustness against perturbation in the initial conditions is also evaluated. We show that RFTD performs consistently well across different scenarios.

6. REFERENCES

[1] T. Baltrusaitis et al. 3d constrained local model for rigid and non-rigid facial tracking. In *CVPR*, 2012.

[2] C. Bao et al. Real time robust L1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012.

[3] Z. Cai et al. Person-specific face tracking with online recognition. In *FG*, 2013.

[4] L. Cehovin et al. Is my new tracker really better than yours? In *WACV*, 2014.

[5] T. B. Dinh et al. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR*, 2011.

[6] H. Grabner et al. Real-time tracking via on-line boosting. In *BMVC*, 2006.

[7] S. Hare et al. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[8] J. F. Henriques et al. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012.

[9] X. Jia et al. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012.

[10] Z. Kalal et al. Tracking-learning-detection. *IEEE PAMI*, 34(7):1409–1422, 2012.

[11] M. Kristan et al. The visual object tracking VOT2014 challenge results. In *ECCV Workshops*, 2014.

[12] M. Roth et al. Robust multi-pose face tracking by multi-stage tracklet association. In *ICPR*, 2012.

[13] J. Santner et al. PROST: parallel robust online simple tracking. In *CVPR*, 2010.

[14] A. W. M. Smeulders et al. Visual tracking: An experimental survey. *IEEE PAMI*, 36(7):1442–1468, 2014.

[15] S. Stalder et al. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *ICCV Workshops*, 2009.

[16] M. Uricar et al. Real-time multi-view facial landmark detector learned by the structured output SVM. In *FG*, 2015.

[17] P. Viola and M. Jones. Robust real-time face detection. *IJCV*, 57(2):137–154, 2004.

[18] The visual tracker benchmark. `http://visual-tracking.net`. Accessed: 2015-11-10.

[19] Y. Wu et al. Online object tracking: A benchmark. In *CVPR*, 2013.

[20] Y. Wu et al. Object tracking benchmark. *IEEE PAMI*, 37(9):1834–1848, 2015.

[21] W. Zhong et al. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012.

[22] M. Zhou et al. AAM based face tracking with temporal matching and face segmentation. In *CVPR*, 2010.

[23] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, 2012.