Multi-Tracker Partition Fusion

ObaidUllah Khalid, Juan Carlos SanMiguel, and Andrea Cavallaro

***Abstract*— We propose a decision-level approach to fuse the output of multiple trackers based on their estimated individual performance. The proposed approach is composed of three main steps. First, we group trackers into clusters based on the spatiotemporal pair-wise correlation of their short-term trajectories. Then, we evaluate performance based on reverse-time analysis with an adaptive reference frame and define the cluster with trackers that appear to be successfully following the target as the *on-target* cluster. Finally, the state estimations produced by trackers in the *on-target* cluster are fused to obtain the target state. The proposed fusion approach uses standard tracker outputs and can therefore combine various types of trackers. We tested the proposed approach with several combinations of state-of-the-art trackers and also compared it with individual trackers and other fusion approaches. The results show that the proposed approach improves the state estimation accuracy under multiple tracking challenges.**

***Index Terms*— Decision Fusion, online performance evaluation, tracker correlation, visual tracking.**

I. Introduction

VISUAL tracking is widely used in applications such as video surveillance, human–computer interaction, activity recognition, and video indexing. A tracker faces several challenges such as occlusions, clutter, changes in target scale, and appearance and variations in scene illumination. Because no individual tracker can still provide accurate results for all challenges [1], fusing complementary trackers whose expected failures are uncorrelated can increase robustness.

Fusion can be performed at a feature or decision level [2]. Feature-based approaches fuse multiple features in a single tracking framework to adapt to appearance changes [2]–[6]. When the features have variable dimensionality and range, adaptation methods are needed to integrate new features [7]. Decision-level fusion combines the output of multiple trackers [7]–[14]. Fusion can happen sequentially [11], [15], [16] using the outputs from specific trackers [8] or employing likelihood-based fusion [13], in parallel [17] or with a hybrid approach [18].

The online evaluation of the quality of current tracking results uses current and past information only, and may help the fusion process [19]. In order to weight trackers or features prior to fusion, online performance evaluation identifies the trackers that follow the target and estimate the accuracy of their outputs at run time [20]. Fusion approaches estimate performance using target velocity [21], democratic integration [6], filter uncertainty [22], likelihood [13], and tracker correlation [17]. Existing performance evaluators provide different score ranges for each tracker [2], require specific trackers [20], or are computationally expensive [23].

In this paper, we propose a decision-level fusion framework that combines the outputs of selected trackers over time based on the spatiotemporal relationships of their results. The main novelties of the proposed approach are a method to identify the trackers that are expected to be on the correct target and the definition of an adaptive reference frame for online performance analysis. We group trackers hierarchically based on their agreement in estimating the target state in terms of spatial location and direction of movement. Using this spatiotemporal agreement, we determine which groups (clusters) of trackers are in the same region and identify the one that is *on-target*. This identification is achieved using an adaptive time-reversed performance evaluation. This evaluation compares the results of trackers running in the reverse temporal direction with the results of the fused output at a specific frame. This specific frame is adaptively determined via online performance evaluation and motion analysis. The final output is then generated by fusing the outputs of trackers within the *on-target* cluster and the selected *on-target* cluster is propagated over time until a split or merge is detected.

This paper is organized as follows. Section II discusses the related work. The overview of the proposed framework is given in Section III, while the tracker clustering and the reverse-analysis evaluation are described in Section IV and Section V, respectively. The experimental results are presented in Section VI. Finally, Section VII concludes this paper.

II. Related Work

In this section, we discuss methods for online evaluation of trackers and features, and their combination.

Tracker performance can be evaluated using trajectories, observation likelihood, or the spatial uncertainty of the target hypotheses. Comparing target state properties such as target velocity [21] to empirical thresholds limits the approach to specific data. By the reversibility property of Markov chains [23], tracker performance is evaluated using

Manuscript received February 4, 2015; revised June 24, 2015 and November 15, 2015; accepted January 27, 2016. Date of publication March 15, 2016; date of current version June 30, 2017. The work of O. Khalid was supported by the Erasmus Mundus Joint Doctorate in Interactive and Cognitive Environments through the Education, Audiovisual and Culture Executive Agency. Juan C. SanMiguel acknowledges the support of the Spanish Government (HA-Video TEC2014-5317-R). A. Cavallaro acknowledges the support of the Artemis JU and the U.K. Technology Strategy Board through the COPCAMS Project, under Grant 332913. This paper was recommended by Associate Editor W.-C. Siu.

O. Khalid and A. Cavallaro are with the Centre for Intelligent Sensing, Queen Mary University of London, London E1 4NS, U.K. (e-mail: o.khalid@qmul.ac.uk; a.cavallaro@qmul.ac.uk).

J. C. SanMiguel is with University Autonoma of Madrid, Madrid 28049, Spain (e-mail: juancarlos.sanmiguel@uam.es). He performed part of this work when he was Post-Doctoral Research Assistant at Queen Mary University of London.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2016.2542699

another tracker running in the reverse temporal direction until a reference frame in which the tracker under evaluation (forward) is assumed to be correct. Then, reverse–forward results are compared using the Mahalanobis distance at the reference frame [23]. However, this reverse analysis considerably increases the computational cost. The tracker likelihood can also be used as performance indicator [13]. However, distractors (i.e., objects with similar features to those of the target) may produce a high likelihood, thus generating misleading measurements.

Performance measures can take advantage of multihypothesis trackers, such as the spatial uncertainty of the hypotheses (particles) of the particle filter (PF) to weight feature contributions [2], to compute a feature rejection probability [5], or to be employed jointly with reverse analysis [20]. When multiple trackers or features are used, performance weights can be estimated as the distance to the fused output [6], [24], the average change of multiple features [25], or correlation among point trajectories [26]. Such a weighting often requires multiple features whose average result is assumed to be accurate.

Fusion at the feature level considers multiple visual features to be combined in a single-tracker framework. An example is the sum of feature likelihoods (color histograms and intensity gradients) [27]. Results can be improved using a performance-based feature weighting, such as in democratic integration where the weight is computed as the distance between the fused and feature output [6], [24]. Appearance models based on sparse coding [28] have also been used for feature-level fusion, where the weighting is determined by the contribution of the feature template to track the target [29], while other approaches discard features that are far away from the target model [30].

Decision-level fusion combines the output of multiple trackers in cascade or in parallel. A *cascade* for fusion defines an execution order where each tracker output is used by the next tracker. Examples include the combination of the sequential execution of the template-based mean shift (MS) and appearance-based trackers [11], two trackers (region and shape) and two detectors (head and motion for people tracking) [15], and the integration of three PFs and one Kalman filter (KF) [16]. Moreover, trackers can be *integrated within the framework of another tracker* [8], [9]. For instance, a head tracker uses MS to improve the PF tracker predictions [8].

In *parallel* tracker fusion, two trackers may be combined using probability density functions (PDFs) [7] or target motion [21]. Moreover, tracker performance within a parallel framework can be measured as the disagreement with other trackers [17], [31] or can be used to select the best tracker [32]. Other approaches may use tracker correlation to improve the overall tracking performance by correcting PFs [18] and KFs [26]. These approaches determine the accuracy as the spatial uncertainty of hypotheses whose value may vary across trackers, thus making tracker fusion difficult.

Learning-based methods have also been proposed [33], where labels (foreground/background) are assigned to image patches. A Bayesian approach is employed for fusion where tracker accuracy is the distance between the fused output and the output of each tracker. Using likelihood as the performance estimator within *tracker interaction* and sampling-based approaches, the tracker with the highest likelihood is chosen [12]. Similarly, multiple motion and appearance models can be used to form a single compound tracker [13].

Decision-level fusion approaches are summarized in Table I and compared with the proposed approach.

TABLE I

DECISION-LEVEL FUSION APPROACHES. KEY *C*: CASCADE, *P*: PARALLEL, *S*: SPATIAL, *T*: TEMPORAL, GLAD: GENERATIVE MODEL OF LABELS, ABILITIES, AND DIFFICULTIES [34]

Ref.	Type	Reliability evaluator		Tracker correlation		Fused trackers	Fusion method
		S	T	S	T		
[11]	C	-	-	-	-	All	Correction
[8], [9]		-	-	-	-		Kernel-Bayesian
[7]	P	-	-	-	-		Product of PDFs
[14]		✓	-	-	-		Weighted sum
[33]		✓	-	-	-		GLAD
[21]		✓	✓	-	-		Correction
[17]		✓	✓	✓	-		Mixture of Gaussians
[18]		✓	-	-	-		Weighted sum
[26]		✓	-	✓	-		Correction
[31]		✓	✓	✓	-	Selected	Correction + Average
[35]		✓	✓	✓	-		Average
[12], [13]		✓	-	-	-		Interaction
Proposed		✓	✓	✓	✓		Average

III. OVERVIEW

We propose a framework to cluster trackers over time and to select the best performing ones for fusion to improve the overall accuracy of target state estimation. The proposed approach is inspired by the *test and select framework* [36] for ensemble combination where accurate classifiers are fused assuming that their errors are diverse. Considering trackers as classifiers, we extend this framework to video tracking by introducing spatiotemporal correlation and adaptive online performance evaluation (Fig. 1).

Let $\mathbf{I} = \{I_t\}_{t=1}^{T}$ be a video sequence of T frames and $\mathbf{F} = \{F^k\}_{k=1}^{K}$ be a set of K trackers. Let the target state x_t^k be a bounding box, defined by a 4D vector $(u_t^k, v_t^k, w_t^k, h_t^k)$, where u_t^k and v_t^k are the target positions and w_t^k and h_t^k are its width and height, respectively. Each tracker F^k uses the observation z_t^k and the target model at frame I_{t-1}, ϕ_{t-1}^k, to estimate the target state at time t

$$x_t^k = F^k\left(x_{t-1}^k, z_t^k, \phi_{t-1}^k\right) \tag{1}$$

where x_{t-1}^k is the state estimate (i.e., the tracker output) at the previous time step.

Let *on-target* and *off-target* be the labels that indicate whether a tracker successfully follows the target. The goal is to identify the successful trackers for the given outputs x_t^k by labeling them as

$$x_t^k \rightarrow l_t^k \in \{\textit{on-target}, \textit{off-target}\}. \tag{2}$$

We determine l_t^k by recognizing groups of trackers (clusters) following the same region in the frame and identifying the cluster with the *on-target* trackers $\mathbb{C}_t^* = \{F^n\}_{n=1}^{N} \subseteq \mathbf{F}$ $(N \leq K)$.

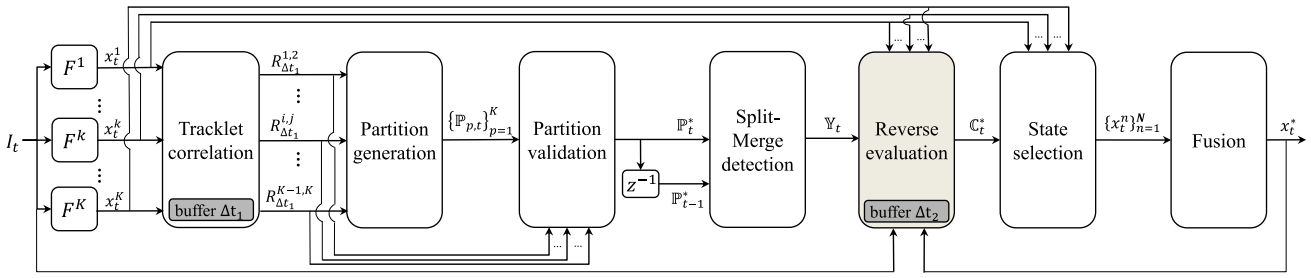


Fig. 1. Block diagram of the proposed approach to fuse the output of K trackers.

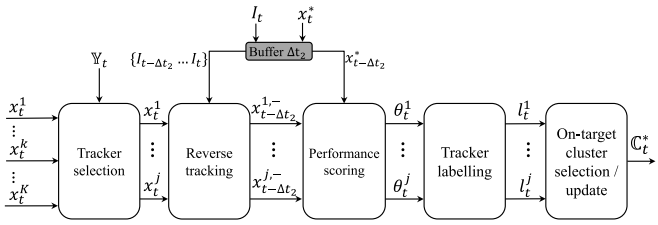


Fig. 2. Block diagram of the reverse evaluation that identifies the *on-target* cluster $\mathbb{C}_t^*$.

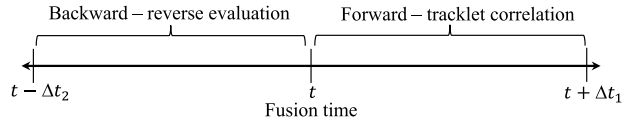


Fig. 3. Temporal windows Δt_1 and Δt_2 employed by the proposed approach to account for forward and backward data, respectively. Forward data are used to determine the relationships among trackers via their trajectories (Section IV). Backward data are used to check tracker performance via a time-reversed evaluator (Section V).

Assuming that all the trackers are initialized with a ground truth, the approach starts with a single *on-target* cluster. When trackers fail, they split into different clusters, of which only one (i.e., $\mathbb{C}_t^*$) or none successfully tracks the target. For each frame I_t, we compute $\binom{K}{2}$ scores $R_{\Delta t_1}^{i,j}$ to determine the spatiotemporal relationships between the pairs of trackers, measured as similarity of spatial location and direction of movement of short-term trajectories (tracklet correlation) over a temporal window Δt_1. These spatiotemporal scores are then employed to generate partition hypotheses $\{\mathbb{P}_{p,t}\}_{p=1}^{K}$ to divide the K trackers into clusters. After validating the best partition $\mathbb{P}_t^*$ by exploring the correlations among tracklet data, the *on-target* cluster $\mathbb{C}_t^*$ is determined by online performance evaluation of the trackers that are expected to be following the target. Such an evaluation uses reverse tracking [23] over a sliding temporal window Δt_2 (Fig. 2), which requires standard tracker outputs (e.g., bounding boxes), thus providing a generic evaluator across trackers.

The proposed approach employs two temporal windows Δt_1 and Δt_2 (Fig. 3), to buffer data from future and past time instants, respectively. The temporal window used for tracklet correlation makes the proposed approach suitable for applications that can tolerate a short latency Δt_1.

$\mathbb{C}_t^*$ is propagated until the detection of a split or a merge, which happens when trackers leave or join the cluster $\mathbb{C}_t^*$, respectively. A split or merge indicates that some or all of the *on-target* trackers may have failed. When such changes occur, all trackers are reevaluated to determine the new *on-target* cluster $\mathbb{C}_t^*$ in the partition $\mathbb{P}_t^*$.

Only the trackers belonging to the *on-target* cluster $\mathbb{C}_t^*$ are used to compute the final target state x_t^*

$$x_t^* = \frac{1}{N}\sum_{n=1}^{N} x_t^n. \tag{3}$$

The proposed clustering (PC) helps to reduce the computational load by avoiding the application of reverse evaluation over all trackers when they maintain their spatiotemporal relationships over time.

IV. Tracker Clustering

A. Tracklet Correlation

We combine the spatial and temporal features of the short-term trackers' trajectories (tracklets) to obtain a set of pairwise correlation scores $R_{\Delta t_1}^{i,j}$, for $1 \le i$ and $j \le K$ with $i \ne j$, for pairs of trackers F^i and F^j over a temporal window Δt_1. The spatial agreement for F^i and F^j is based on their outputs x_t^i and x_t^j at frame I_t

$$O_t^{i,j} = \frac{2\left|A_t^i \bigcap A_t^j\right|}{\left|A_t^i\right| + \left|A_t^j\right|} \tag{4}$$

where A_t^i and A_t^j are the sets containing the pixels of the bounding boxes generated by trackers F^i and F^j, respectively, and $|.|$ is the cardinality of a set. $O_t^{i,j} \in [0,1]$ and a value of 1 (0) represents a full agreement (disagreement). The spatial agreement over time is computed by averaging $O_t^{i,j}$ over Δt_1

$$O_{\Delta t_1}^{i,j} = \frac{1}{\Delta t_1}\sum_{t}^{t+\Delta t_1} O_t^{i,j}. \tag{5}$$

In order to estimate the agreement for motion direction, we compute a score $r_{\Delta t_1}^{i,j}$ using the directional feature $\vec{d}^k$ of each F^k [37] over Δt_1 that encodes the trajectory direction

$$\vec{d}_t^k = \left(u_{t+\Delta t_1}^k - u_t^k, v_{t+\Delta t_1}^k - v_t^k\right). \tag{6}$$

The directional similarity score $r_{\Delta t_1}^{i,j}$ is computed between F^i and F^j using the cosine similarity

$$r_{\Delta t_1}^{i,j} = \cos\left(\left(\vec{d}_t^i \cdot \vec{d}_t^j\right) \Big/ \left(\left|\vec{d}_t^i\right| \cdot \left|\vec{d}_t^j\right|\right)\right) \tag{7}$$

where $r_{\Delta t_1}^{i,j} \in [-1,1]$ and the negative values represent (estimated) targets moving in opposite directions.

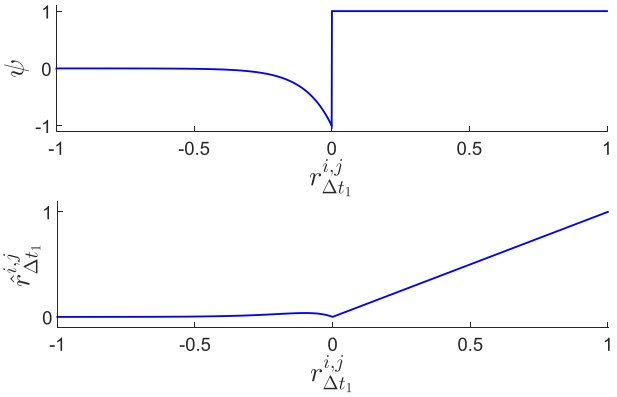


Fig. 4. Weighting function ψ (top) and weighted directional score $\hat{r}^{i,j}_{\Delta t_1}$ (bottom) for directional feature normalization using $\lambda = 10$.

The desired correlation score $R^{i,j}_{\Delta t_1}$ is obtained by combining $O^{i,j}_{\Delta t_1}$ and $r^{i,j}_{\Delta t_1}$ after normalization of $r^{i,j}_{\Delta t_1}$ to [0, 1]. Since we are interested in agreement on the direction of motion, we seek values $r^{i,j}_{\Delta t_1} \in [0, 1]$ and define a weighted directional similarity score $\hat{r}^{i,j}_{\Delta t_1}$ as

$$\hat{r}^{i,j}_{\Delta t_1} = \psi\left(\lambda, r^{i,j}_{\Delta t_1}\right) \cdot r^{i,j}_{\Delta t_1} \tag{8}$$

where $\hat{r}^{i,j}_{\Delta t_1} \in [0, 1]$ and $\psi \in [-1, 1]$ is a weighting function that assigns a constant weight to $r^{i,j}_{\Delta t_1} \in [0, 1]$, while $r^{i,j}_{\Delta t_1} \in [-1, 0]$ are given low weights. Such a function is defined as

$$\psi\left(\lambda, r^{i,j}_{\Delta t_1}\right) = \begin{cases} 1, & \text{if } 0 \le r^{i,j}_{\Delta t_1} \le 1 \\ -e^{\lambda \cdot r^{i,j}_{\Delta t_1}}, & \text{if } -1 \le r^{i,j}_{\Delta t_1} < 0 \end{cases} \tag{9}$$

where $\lambda \in (0, \infty)$ is the decay rate of ψ. Values of λ close to zero give smooth transitions for $\psi \in [-1, 0]$ turning into high $\hat{r}^{i,j}_{\Delta t_1}$ values when $r^{i,j}_{\Delta t_1} \in [-1, 0]$. High values of λ give abrupt transitions for ψ turning into $\hat{r}^{i,j}_{\Delta t_1}$ values close to zero. Fig. 4 shows the relations between $r^{i,j}_{\Delta t_1}$ and $\hat{r}^{i,j}_{\Delta t_1}$ for $\lambda = 10$ [Fig. 4 (bottom)] and between $r^{i,j}_{\Delta t_1}$ and ψ [Fig. 4 (top)].

$R^{i,j}_{\Delta t_1}$ is finally computed as

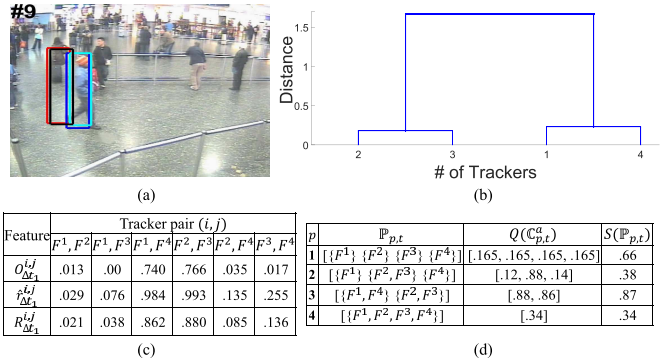
$$R^{i,j}_{\Delta t_1} = \omega \cdot O^{i,j}_{\Delta t_1} + (1 - \omega) \cdot \hat{r}^{i,j}_{\Delta t_1} \tag{10}$$

where $\omega \in [0, 1]$. High (low) values of ω prioritize the spatial overlap (trajectory direction), which can be useful for short (long) Δt_1.

B. Partition Generation

A single partition $\mathbb{P}_{p,t}$ of $\mathbf{F}$ is a collection of nonempty clusters $\mathbb{C}^a_{p,t}$ ($a = 1, \ldots, |\mathbb{P}_{p,t}|$) such that each tracker in $\mathbf{F}$ is in exactly one $\mathbb{C}^a_{p,t}$, i.e., all $\mathbb{C}^a_{p,t}$ are mutually disjoint.

At each time step, K trackers can be grouped into clusters $\mathbb{C}^a_{p,t}$, forming a single partition $\mathbb{P}_{p,t}$, where $|\mathbb{P}_{p,t}| \in [1, K]$. Let [.] represent a partition. For example, $[\{F^1, \ldots\ldots, F^k\}]$ means that all trackers are clustered together ($|\mathbb{P}_{p,t}| = 1$, initial condition) and $[\{F^1\}, \ldots\ldots, \{F^k\}]$ means that each tracker is a single cluster ($|\mathbb{P}_{p,t}| = K$).



(a) (b)

Feature	Tracker pair (i, j)						
	F^1, F^2	F^1, F^3	F^1, F^4	F^2, F^3	F^2, F^4	F^3, F^4
$O^{i,j}_{\Delta t_1}$	.013	.00	.740	.766	.035	.017
$r^{i,j}_{\Delta t_1}$	.029	.076	.984	.993	.135	.255
$R^{i,j}_{\Delta t_1}$	.021	.038	.862	.880	.085	.136

(c)

p	$\mathbb{P}_{p,t}$	$Q(\mathbb{C}^a_{p,t})$	$S(\mathbb{P}_{p,t})$
1	$[\{F^1\}\ \{F^2\}\ \{F^3\}\ \{F^4\}]$	[.165, .165, .165, .165]	.66
2	$[\{F^1\}\ \{F^2, F^3\}\ \{F^4\}]$	[.12, .88, .14]	.38
3	$[\{F^1, F^4\}\ \{F^2, F^3\}]$	[.88, .86]	.87
4	$[\{F^1, F^2, F^3, F^4\}]$	[.34]	.34

(d)

Fig. 5. (a) Tracking results for frame 9 of the *MCTTR0205a* sequence (TRECVID)— F^1 (red rectangle), F^2 (blue rectangle), F^3 (turquoise rectangle), and F^4 (black rectangle). (b) Dendrogram obtained by HC. (c) Pair-wise tracker correlations scores. (d) Hypothesized partitions and cluster scores, where $\mathbb{P}_{3,t}$ has the highest score.

Our aim is to hypothesize a set of partitions $\{\mathbb{P}_{p,t}\}^B_{p=1}$ to cluster the trackers. All possible partitions $\mathbb{P}_{p,t}$ can be systematically enumerated with an exhaustive search [38]. The set size is given by the Bell number $\mathscr{B}$ [39], which exponentially increases with K. For example, with $K = 8$ trackers $\mathscr{B} = 4140$ partitions are generated.

To reduce the computational complexity, we use a greedy search that determines the most plausible partitions for a given number of clusters. Since the optimum number and composition of clusters is unknown, we take advantage of the hierarchical structure of the tracker relationships to generate a set of partitions whose cardinality ranges from 1 (i.e., a single cluster) to K (i.e., each tracker is a cluster), with $K \ll \mathscr{B}$.

We use hierarchical clustering (HC) [40] to determine the relationships between trackers based on the pair-wise tracker correlation scores $R^{i,j}_{\Delta t_1}$. Based on the distance between the trackers' outputs, we obtain a dendrogram, which is inspected by a divisive (top-down) approach to determine each partition $\mathbb{P}_{p,t}$. The search starts with the partition that groups all trackers into one cluster $\mathbb{P}_{1,t}$. Recursively moving down the tree, a different $\mathbb{P}_{p,t}$ is generated at each level, with the final partition having each tracker in a separated cluster $\mathbb{P}_{K,t}$. A partition $\mathbb{P}_{p,t}$ is obtained as

$$\mathbb{P}_{p,t} = f(\beta^*(p)) \tag{11}$$

where $p = 1, \ldots, K$ and $f(\beta^*)$ is an HC-based function that provides a cluster partition given an optimal distance threshold β^*, which is computed as

$$\beta^*(p) = \arg\min_{\beta} \{|f(\beta)| - p\} \tag{12}$$

where $\beta = 0, \ldots, \max\{R^{i,j}_{\Delta t_1}\}$.

The proposed greedy search has a linear relationship between the size of $\{\mathbb{P}_{p,t}\}^K_{p=1}$ and K, which significantly speeds up the search. Fig. 5(a) and (c) shows an example for four trackers and the scores for their spatiotemporal relations, which are used to compute the dendrogram illustrated in Fig. 5(b).

C. Partition Validation

After generating the set of partitions $\{\mathbb{P}_{p,t}\}_{p=1}^{K}$, the objective is to select the partition $\mathbb{P}_t^*$ that best represents the spatiotemporal relations among trackers. We therefore define the score $S(\mathbb{P}_{p,t})$ as

$$S(\mathbb{P}_{p,t}) = \frac{1}{|\mathbb{P}_{p,t}|} \sum_{a=1}^{|\mathbb{P}_{p,t}|} Q\big(\mathbb{C}_{p,t}^a\big) \tag{13}$$

where $Q(\mathbb{C}_{p,t}^a)$ is the score for a single cluster $\mathbb{C}_{p,t}^a \in \mathbb{P}_{p,t}$. $S(\mathbb{P}_{p,t})$ determines the partition $\mathbb{P}_t^*$ as

$$\mathbb{P}_t^* = \underset{p}{\operatorname{argmax}}\{S(\mathbb{P}_{p,t})\} \tag{14}$$

where $p = 1, \ldots, K$ and $Q(\mathbb{C}_{p,t}^a)$ is dependent on the pair-wise relationship score between trackers F^i and F^j in $\mathbb{C}_{p,t}^a$, which is obtained in the *tracklet correlation* block (Section IV-A).

The score $Q(\mathbb{C}_{p,t}^a)$ is computed as

$$Q\big(\mathbb{C}_{p,t}^a\big) = \begin{cases} \dfrac{1}{\nu} \displaystyle\sum_{i=1}^{\left|\mathbb{C}_{p,t}^a\right|} \sum_{j=1}^{\left|\mathbb{C}_{p,t}^a\right|} R_{\Delta t_1}^{i,j}, & \text{if } \left|\mathbb{C}_{p,t}^a\right| > 1 \\ 1 - \displaystyle\max_{b \in 1,\ldots,|\mathbb{P}_{p,t}|} \big(Q\big(\mathbb{C}_{p,t}^a \bigcup \mathbb{C}_{p,t}^b\big)\big), & \text{if } \left|\mathbb{C}_{p,t}^a\right| = 1 \end{cases} \tag{15}$$

where $\nu = \binom{|\mathbb{C}_{p,t}^a|}{2}$ is the total number of tracker-pair combinations within the cluster. Since a pair-wise score for a single tracker in a cluster, $|\mathbb{C}_{p,t}^a| = 1$, cannot be obtained, we compute its pair-wise scores with trackers in other clusters. Therefore, $\mathbb{C}_{p,t}^a \bigcup \mathbb{C}_{p,t}^b$ indicates the hypothetical case where the tracker in $\mathbb{C}_{p,t}^a$ becomes part of the cluster $\mathbb{C}_{p,t}^b$ and b is any of the remaining clusters within $\mathbb{P}_{p,t}$ ($b \neq a$).

When each tracker is a single cluster, i.e., $|\mathbb{P}_{p,t}| = K$, each cluster score $Q(\mathbb{C}_{p,t}^a)$ is computed as

$$Q\big(\mathbb{C}_{p,t}^a\big) = 1 - Q\big(\mathbb{C}_{p,t}^b\big) \tag{16}$$

where $\mathbb{C}_{p,t}^b$ is the cluster containing all the trackers. Fig. 5(d) shows the computed cluster and partition scores, where $\mathbb{P}_{3,t}$ achieves the highest score.

D. Split–Merge Detection

After determining $\mathbb{P}_t^*$, the *split–merge detection* step identifies changes between $\mathbb{P}_t^*$ and the previous partition $\mathbb{P}_{t-1}^*$. Such changes may occur due to trackers leaving or joining the *on-target* cluster in the previous time step $\mathbb{C}_{t-1}^* \in \mathbb{P}_{t-1}^*$, hence modifying the structure of $\mathbb{P}_{t-1}^*$. Thus, $\mathbb{C}_{t-1}^*$ cannot be propagated to the current time and the reverse evaluation is required to identify the current $\mathbb{C}_t^*$. We apply the reverse evaluation over a set of trackers $\mathbb{Y}_t$, selected as

$$\mathbb{Y}_t = \begin{cases} \mathbb{C}_{t-1}^* & \text{if } \mathbb{P}_{t-1}^* \equiv \mathbb{P}_t^* \\ \mathbb{P}_t^* & \text{otherwise} \end{cases} \tag{17}$$

where the condition $\mathbb{P}_{t-1}^* \equiv \mathbb{P}_t^*$ checks the similarity between the number of clusters and their members (i.e., trackers). When this condition is satisfied, an existing cluster $\mathbb{C}_t^a \in \mathbb{P}_t^*$ equivalent to $\mathbb{C}_{t-1}^*$ is used at the current time. However, when a split or merge occurs, all the trackers in $\mathbb{P}_t^*$ are evaluated.

V. On-Target Cluster Identification

We evaluate the performance of each tracker in the set $\mathbb{Y}_t$ (*tracker selection* block in Fig. 2). This performance evaluation either determines the *on-target* cluster $\mathbb{C}_t^*$ of the valid partition $\mathbb{P}_t^*$ or validates the *on-target* cluster from the previous time step $\mathbb{C}_{t-1}^*$. We cast this problem as an online tracker evaluation and use the time reversibility of target motion to assess the performance of the trackers. We first review reverse-based evaluation methods and then we present our proposed improvements.

A. Reverse-Based Online Evaluation

Reverse-based evaluation [23] measures the performance of a tracker during runtime using the generated results. For each frame where we evaluate the tracker, a reversed tracker (i.e., the same tracker operating in reverse time) is applied. Using the tracker output x_t^k as the reverse-tracker initialization $x_t^{k,-}$, the reverse tracker obtains its output as

$$x_{t-1}^{k,-} = F^k\big(x_t^{k,-}, z_t^k, \phi_t^k\big) \tag{18}$$

where $x_{t-1}^{k,-}$ is the reverse-tracker output at time $t-1$. Then the result of the reverse tracker and that of the tracker are compared to obtain a similarity score θ_t^k by means of the Mahalanobis distance between the likelihood distributions of the forward and reverse target estimations. This comparison is performed at a certain time instant t_{ref}, that is associated to the reference frame I_{ref}. I_{ref} is a frame where the tracker is known to be *on-target* and it is usually $I_{\text{ref}} = I_1$ [23], i.e., the frame where the target is initialized.

This approach has two major limitations. First, the forward–reverse similarity uses the Mahalanobis distance that returns unbounded scores $\theta_t^k \in [0, +\infty)$, which can have a different range of values depending on the trackers employed in the fusion framework. Hence, θ_t^k may be inappropriate to compare the trackers to be combined. Second, running the reverse tracker until the first frame implies a considerable growth in computational time as the sequence progresses. A faster approximation is proposed where I_{ref} is moved ahead in time. However, tracker errors may accumulate over time by the reverse tracker, thus leading to drift [41]. For example, if the tracker loses the target and gets locked on the background, the forward–reverse similarity may give high scores θ_t^k, since the reverse tracker is incorrectly initialized by the wrong tracker estimations.

We address these shortcomings for reverse-tracking evaluation as described next.

B. Performance Score and Reference Frame Update

To address the limitation associated to the unbounded θ_t^k scores, we compare the reverse-tracker and fused outputs to obtain a $\theta_t^k \in [0, 1]$

$$\theta_t^k = G\big(x_{\text{ref}}^{k,-}, x_{\text{ref}}^*\big) \tag{19}$$

where $x_{\text{ref}}^{k,-}$ and x_{ref}^* are the reverse-tracker and fused outputs at I_{ref}, respectively, and G defines the output similarity that is computed using (4), where A_t^i and A_t^j are replaced

by $A_{\text{ref}}^{x^*}$ and $A_{\text{ref}}^{k,-}$, respectively. $A_{\text{ref}}^{x^*}$ and $A_{\text{ref}}^{k,-}$ are the sets containing the pixels of the bounding boxes of x_{ref}^* and $x_{\text{ref}}^{k,-}$, respectively.

To limit the growth of computational time when $I_{\text{ref}} = 1$, we update I_{ref} over time so that the computational cost is bounded and the reverse evaluation can be applied to long sequences. We implement such an update assuming that the fused output is *on-target* and that the target has changed position from I_{ref} to the current frame, thus making the motion information useful for reverse analysis.

The motion of bounding boxes is minimal when the tracker is *on-target* and the target is static or when the tracker drifts from the target and gets locked onto a background region. Because it is difficult to differentiate between these two situations, we analyze significant motion changes of the trackers compared with their average motion. The maximum motion M^k is computed over temporal window Δt_2 using F^k trajectory (Fig. 3), and the top-left $(u_{1,t}^k, v_{1,t}^k)$ and bottom-right $(u_{2,t}^k, v_{2,t}^k)$ coordinates of the bounding box. The motion for u_1^k over Δt_2 is computed as

$$M_{u_1^k} = \frac{1}{\Delta t_2} \sum_{t'=t-\Delta t_2}^{t} \left(M_{u_{1,t'}^k} - M_{u_{1,t'-1}^k} \right). \tag{20}$$

The motion for $v_{1,t}^k, u_{2,t}^k$, and $v_{2,t}^k$ is computed using (20), where $M_{u_1^k}$ is replaced by $M_{v_1^k}$, $M_{u_2^k}$, and $M_{v_2^k}$, respectively. $M^k = \max(M_{u_1^k}, M_{v_1^k}, M_{u_2^k}, M_{v_2^k})$ returns the maximum motion for F^k. $M^{x^*} = (1/\Delta t_2) \sum_{t'=1}^{t} (M_{x_{t'}^*} - M_{x_{t'-1}^*})$ determines the motion of the fused output, which is used as a common threshold to compare the motion of all the trackers in the framework.

The performance of each tracker is computed using (19). To determine a single I_{ref} for all the trackers, we use $\max(M^k)$ and $\max(\theta_t^k)$ to select the best performing tracker for that temporal window.

We adaptively estimate and update I_{ref} by combining the motion analysis and performance of the tracker as

$$I_{\text{ref}} = \begin{cases} I_{t-\Delta t_2}, & \text{if } \max(M^k) \geq M^{x^*} \text{ and } \max\left(\theta_t^k\right) \geq \tau_1 \\ I_{\text{ref}}, & \text{otherwise} \end{cases} \tag{21}$$

where $\tau_1 = 0.5$ is the minimum tracker accuracy [1].

C. On-Target Cluster Selection/Update

Reverse evaluation identifies the *on-target* trackers using the individual performance scores θ_t^k of trackers in $\mathbb{Y}_t$. Trackers with $\theta_t^k \geq \tau_1$ are labeled *on-target*, enabling the method to select $\mathbb{C}_t^*$ as the cluster $\mathbb{C}_t^a$ with all *on-target* trackers

$$\mathbb{C}_t^* = \left\{ \mathbb{C}_t^a \in \mathbb{P}_t^* : l_t^k = \textit{on-target} \quad \forall F_t^k \in \mathbb{C}_t^a \right\}. \tag{22}$$

VI. Experimental Results

A. Setup

1) Data Set: For evaluating the proposed approach, multi-tracker partition fusion (TPF), we consider the following data sets: Students,[1] CAVIAR,[2] PETS (2009[3] and 2001),[4] LTDT,[5] TRECVID2009,[6] MIT Traffic,[7] David,[8] and AVSS2007.[9] We have selected 22 sequences (3580 frames) to cover indoor and outdoor scenarios containing tracking challenges such as occlusions, background clutter, pose, motion, and illumination changes. Table II describes the selected sequences and Fig. 6 shows the target initializations.

TABLE II

Sequences Used in the Experiments. KEY BC: Background Clutter, *P*: Pose Changes, *O*: Occlusions, *I*: Illumination Changes, *S*: Scale Changes, *M*: Motion Changes, BS: Background Similarity

Dataset		Sequence name	Target Class	Size		Total Frames	Challenges
				Target	Frame		
Students	P1	University Students	Person	21 x 75	720 x 576	250	BC, P, O, I
	P2		Person	22 x 69		250	BC, P, I
	P3		Person	25 x 61		165	P, I
CAVIAR	P4	Browse_WhileWaiting1	Person	50 x 24	384 x 288	200	P, I
	P5	OneLeaveShopReenter1Front	Person	16 x 56		195	P, I, S, BS
	P6	OneLeaveShopReenter2front	Person	14 x 50		300	P, I, S, BS
	P7	ThreePastShop2cor	Person	56 x 142		170	P, I, S
PETS	P8	S2.L2 walking	Person	14 x 50	768 x 576	140	P, I, S, BS
	P9	PETS2001 Dataset 1	Vehicle	56 x 142		150	P, I, S
	P10	S2.L1 walking	Person	22 x 68		150	BC, I,S
	P11	PETS2001 Dataset 1	Vehicle	72 x 56		180	S, M
	P12	S2.L1 walking	Person	72 x 56		90	O, M
	P13		Person	72 x 56		150	M, O, P
	P14		Person	72 x 56		110	S, M
LTDT	P15	NissanSkylineChase	Person	37 x 21	640 x 275	300	I, M, S
David	P16	David Indoor	Head	91 x 116	320 x 240	130	I, S, M, BS
AVSS2007	P17	Abandoned baggage	Person	60 x 240	720 x 576	200	P, I
TRECVID	P18	MCTTR0205a	Person	72 x 226	720 x 576	50	P, I, BS
	P19		Person	64 x 204		40	P, I, BS
	P20		Person	64 x 204		40	I, M, BS
MIT Traffic	P21	MV2_001	Vehicle	34 x 26	720 x 480	160	I, S, M
	P22	MV2_006	Vehicle	70 x 36	720 x 576	160	O, I, M

2) Trackers: We apply the proposed TPF to combinations of up to eight trackers using publicly available authors' implementations.

The first tracker is the sparse-features-based tracker (ST) [42], which is PF-based and uses sparse (intensity) features to generate the target appearance model. The *maximum a posteriori* criterion is employed to estimate the target state. The second tracker is the adaptive fragments-based tracker (AFT) [43] that models the target appearance with various fragments. Fragment reliability is based on color similarity between the current and previous fragments, to integrate highly reliable fragments within a PF framework. The third tracker is the locally orderless tracker (LOT) [44] that divides the

[1] http://graphics.cs.ucy.ac.cy/research/downloads/crowd-data
[2] http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/
[3] http://www.cvg.reading.ac.uk/PETS2009
[4] http://www.cvg.reading.ac.uk/slides/pets.html
[5] http://www.micc.unifi.it/LTDT2014
[6] http://trecvid.nist.gov/trecvid.data.html#tv09
[7] http://www.ee.cuhk.edu.hk/~xgwang/HBM.html
[8] http://www.cs.toronto.edu/~dross/ivt/
[9] http://www.avss2007.org/

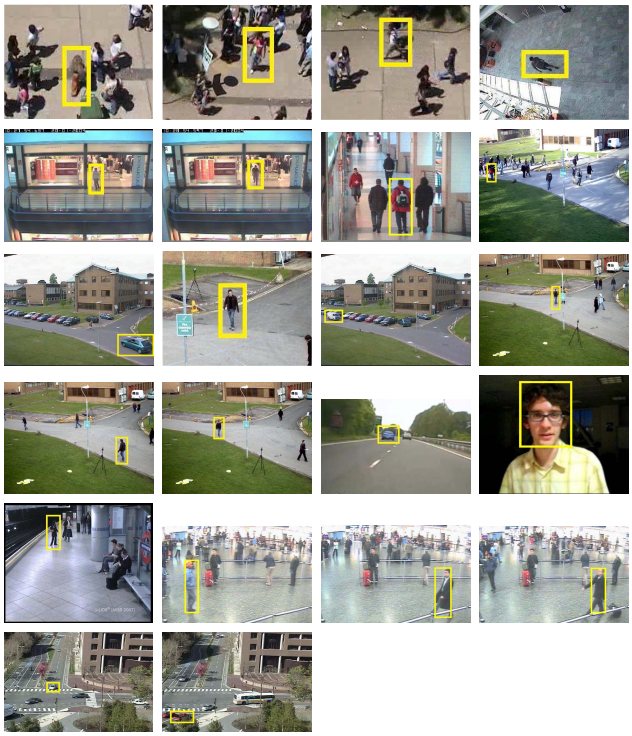


Fig. 6. Target initializations. Top-left to bottom-right: the order along the rows corresponds to the row order in Table II.

target into *superpixels* using the HSV color space and employs a PF to track the target. The fourth tracker is the incremental visual tracker (IVT) [45] that performs online updating to account for appearance changes and uses a PF to track the target over time. The fifth tracker is the scale and orientation adaptive MS tracker (AMS) [46] that estimates the changes in scale and orientation of the target using the MS framework by employing Gaussian kernels and image moments. The sixth tracker is the fast compressive tracker (FCT) [47] that projects the original image to a low-dimensional space. The projected features are then used to formulate tracking as a binary classification task via a naive Bayesian classifier. The seventh tracker is the L1 Tracker (L1T) [48], which is based on PF and models the target by sparse linear combinations of target and trivial templates (set of unit vectors). Assuming an affine motion model, tracking is performed by solving the L_1 minimization problem. The eighth tracker is the least soft-threshold squares tracker (LSST) [49], which is based on PF and performs linear regression via least soft-threshold squares distance between the observation and the target model.

We have implemented six TPF configurations: TPF_3 (ST, AFT, and LOT), TPF_4 (ST, AFT, LOT, and IVT), TPF_5 (ST, AFT, LOT, IVT, and FCT), TPF_6 (ST, AFT, LOT, IVT, FCT, and AMS), TPF_7 (ST, AFT, LOT, IVT, FCT, AMS, and L1T), and TPF_8 (ST, AFT, LOT, IVT, FCT, AMS, L1T, and LSST). We use TPF_3 for Sections VI-C and VI-D, while Sections VI-E–VI-G use all six configurations.

We compare TPF with the eight selected trackers, two recent trackers (the kernelized correlation filter (KCF) [3] and STRUCK (STR) [50]) and three state-of-the-art (SOA) decision-level fusion approaches: average fusion (AvgF), visual tracker sampler (VTS) [13], and symbiotic tracker (SymT) [17]. STRUCK [50] is a tracking-by-detection approach using SVMs with Gaussian kernels. Three features have been tested (Haar, raw pixels, and intensity histograms) and we report the results for histograms as they outperformed Haar features and raw pixels. KCF [3] employs correlation among filters based on histograms of oriented gradients features. AvgF combines the eight trackers by assigning equal weights to each tracker. SymT estimates the trackers' relationships based on their spatial agreement only, and the tracker performance is based on the displacements between consecutive frames. SymT has been reimplemented as described in [17]. VTS combines two motion and four appearance models to get eight trackers, using a likelihood-based tracker performance measure. For STRUCK, KCF and VTS we use the authors' implementation.

3) Parameters of the Proposed Approach: For TPF, the temporal window for the reverse analysis is initially set to $\Delta t_2 = 10$ since it provides a good speed–accuracy trade-off as shown in [23]. This value is updated if the motion or the performance of the trackers is below the thresholds (see Section V-B). The temporal window for tracklet correlation is set to $\Delta t_1 = 10$ to keep an initial forward–backward symmetry for analysis, since no prior information is available to define the importance of one analysis over the other. $\omega = 0.5$ ensures equal weighting for the features. For (8), we heuristically found that $\lambda \in [5, 15]$ gives the desired ψ behavior, so we used $\lambda = 10$. Finally, $\tau_1 = \tau_2 = 0.5$ as previously set [1].

B. Evaluation Measures

We measure the deviation from ground-truth (GT) data as the overlap score $O_t^{k,\text{GT}}$ between the output of tracker F^k and GT annotation using (4). A_t^i and A_t^j are replaced by A_t^{GT} and A_t^k, the sets of pixels contained in the GT and F^k target estimations, respectively. Values close to 1 (0) indicate high (low) tracking performance. The mean of $O_t^{k,\text{GT}}$ is computed for each sequence.

We also measure the performance of TPF when assigning the *on-target* and *off-target* labels to trackers and clusters from the valid partition $\mathbb{P}_t^*$. GT information is used to compute the overlap score for each cluster $O_t^{\mathbb{C}^a,\text{GT}}$ by taking the average of $O_t^{k,\text{GT}}$ for the trackers within the cluster. The *on-target* trackers are defined for $O_t^{k,\text{GT}} \geq \tau_2$ and $O_t^{\mathbb{C}^a,\text{GT}} \geq \tau_2$ corresponds to the *on-target* cluster $\mathbb{C}_t^*$. To simplify the notation in the remaining sections, we denote the mean GT overlap $O_t^{k,\text{GT}}$ for each sequence as O_G.

Using n_{TP}, n_{FP}, n_{TN}, and n_{FN} as number of true positives (TP), false positives (FP), true negatives (TN) and false negatives (FN), we compute the precision, $\text{Pr} = n_{\text{TP}}/(n_{\text{TP}} + n_{\text{FP}})$, the recall, $\text{Re} = n_{\text{TP}}/(n_{\text{TP}} + n_{\text{FN}})$, and F-score $= 2 \cdot (\text{Pr} \cdot \text{Re}/\text{Pr} + \text{Re})$ [1]. Values for the F-score close to 1 (0) indicate high (low) accuracy. n_{TP} (n_{FP}) and n_{TN} (n_{FN}) are the number of clusters or trackers correctly (incorrectly) labeled *on-target* and *off-target*, respectively.

C. Tracker Clustering

We evaluate the performance of the partition generation approach and the features employed to cluster the trackers.

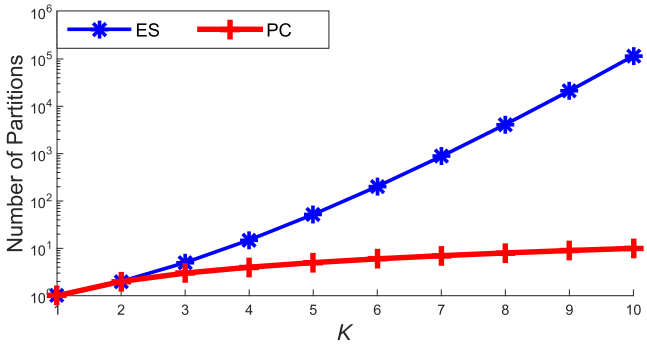


Fig. 7. Comparison of the proposed clustering (PC) with exhaustive search (ES) to generate partitions $\{\mathbb{P}_{p,t}\}_{p=1}^{B}$ of K trackers. With ES, B grows exponentially, while the PC is bounded by $B = K$.

TABLE III

COMPARISON OF FEATURE COMBINATIONS FOR THE PROPOSED APPROACH. THE RESULTS SHOW THE F-SCORE AT THE TRACKER LEVEL AND CLUSTER LEVEL, WITH DIFFERENT FEATURE WEIGHTS ω IN (10)

	Tracker-level			Cluster-level		
	ω=0	ω=0.5	ω=1	ω=0	ω=0.5	ω=1
P1	.92	.90	.87	.94	.92	.89
P2	.98	.98	.97	.99	.99	.98
P3	.91	.92	.97	.87	.90	.98
P4	.87	.90	.94	.80	.87	.92
P5	.87	.90	.94	.93	.93	.99
P6	.44	.48	.45	.38	.42	.40
P7	.86	.78	.84	.99	.87	.87
P8	.88	.94	.93	.90	.97	.96
P9	.87	.81	.82	.99	.95	.84
P10	.46	.82	.37	.57	.90	.34
P11	.92	1	.94	1	1	.98
P12	.63	.77	.83	.60	.75	.83
P13	.88	.95	.87	.87	.98	.90
P14	.95	.96	.98	1	1	1
P15	.96	.97	.98	1	1	.99
P16	.81	.81	.93	.81	.88	1
P17	.52	.55	.56	.53	.67	.47
P18	.51	.93	.88	.60	.99	.96
P19	.98	.98	.98	1	1	1
P20	.95	1	.98	.97	1	1
P21	.95	.96	.96	.95	.98	.99
P22	.91	.95	.52	.99	1	.60
Mean	.82	**.88**	.84	.85	**.91**	.86

1) Comparison of the Proposed Clustering (PC) With Exhaustive Search (ES): Fig. 7 compares the generated set of partitions $\{\mathbb{P}_{p,t}\}_{p=1}^{B}$ with an increasing number of trackers K for both approaches. The accuracy of their results is equal as the PC and ES select the same valid partition. However, the size grows exponentially for ES with an increasing K, whereas PC keeps the size of $\{\mathbb{P}_{p,t}\}_{p=1}^{B}$ bounded with respect to $B = K$.

2) Performance Analysis of Features: TPF combines the features $O_{\Delta t_1}^{i,j}$ and $\hat{r}_{\Delta t_1}^{i,j}$ to get $R_{\Delta t_1}^{i,j}$. An accuracy comparison at the cluster level and tracker level is presented in Table III. The results indicate that combining both features outperforms using single features. At the tracker level, using both features improves the F-score by 5% (7%) compared with using only the overlap (direction) feature. Similarly at the cluster level, an improvement of 6% (7%) is observed in comparison with the overlap (direction) feature.

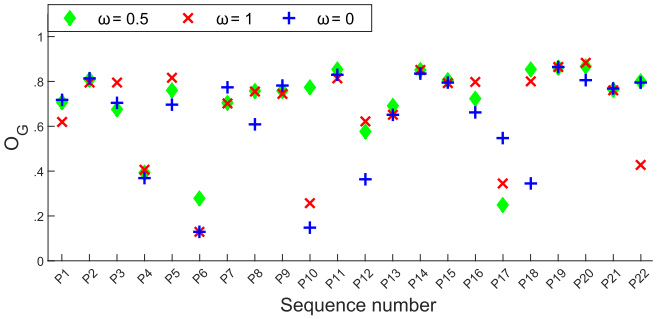


Fig. 8. Tracker accuracy O_G using individual features ($\omega = 1$ for overlap and $\omega = 0$ for direction) and their equal combination ($\omega = 0.5$).

TABLE IV

F-SCORE WITH (TPF) AND WITHOUT (TPF') MOTION ANALYSIS FOR THE THREE FUSED TRACKERS WITH THE REFERENCE FRAME I_{ref} UPDATED USING MOTION ANALYSIS (SECTION V)

	ST [42]		AFT [43]		LOT [44]		Sequence mean for all trackers	
	TPF'	TPF	TPF'	TPF	TPF'	TPF	TPF'	TPF
P1	.92	.88	.96	.92	.94	.90	.94	.90
P2	1	1	1	.97	1	.97	1	.98
P3	.95	.99	.95	.91	1	.87	.97	.92
P4	.85	.94	.78	.95	.59	.74	.74	.88
P5	.81	.87	.94	.92	.61	.76	.79	.85
P6	.26	.41	.41	.60	.52	.43	.40	.48
P7	.53	.67	.74	.82	.63	.78	.63	.76
P8	.88	.87	.99	.94	1	.98	.95	.93
P9	.90	.72	.90	.78	.89	.91	.89	.80
P10	.88	.75	.91	.83	.88	.82	.89	.80
P11	1	1	1	1	1	1	1	1
P12	.57	.82	.61	.66	.46	.85	.55	.77
P13	.79	.98	.83	.96	.94	.92	.86	.95
P14	1	.98	.96	.95	.91	.94	.96	.96
P15	.99	.98	.96	.96	.95	.99	.97	.98
P16	.91	.85	.99	.86	.88	.73	.93	.81
P17	.51	.54	.80	.40	.88	.48	.73	.47
P18	.29	.33	.05	.87	.08	1	.14	.73
P19	1	1	1	1	1	.95	1	.98
P20	1	1	.97	.99	1	1	.99	1
P21	.94	.88	1	1	1	.99	.98	.96
P22	.92	.96	.81	.97	.92	.86	.88	.93
Tracker mean	.81	**.83**	.84	**.87**	.82	**.86**	.83	**.85**

Fig. 8 shows the tracking accuracy O_G for the three features, where $\omega = 0.5$ improves the results globally in 60% of the sequences. Individual features do not always increase the performance since no feature is optimal for all situations.

D. On-Target Cluster Identification

1) Performance Analysis for Motion: Table IV compares the proposed approach with and without motion analysis (TPF and TPF', respectively) to update I_{ref}, in terms of the F-score for selecting the *on-target* trackers. The results for P4–P7 indicate the case when trackers might lose the target due to background clutter and get locked on the background. Since TPF' is unable to detect this situation in P4, it determines the trackers to be always *on-target*. TPF improves TPF' by 10%, 22%, and 25% for ST, AFT, and LOT, respectively (19% mean improvement). For P5, P6, and P7, TPF improves the performance by 8%, 20%, and 20%, respectively. P12 remains occluded between frames 29–39 where ST loses the target and becomes locked on to foreground objects being labeled *on-target* by TPF', whereas AFT and LOT are labeled

TABLE V

MEAN OVERLAP SCORE COMPARISON IN TERMS OF O_G (TEN INDEPENDENT RUNS). KEY: ST [42], AFT [43], LOT [44], IVT [45], FCT [47], AMS: MS TRACKER [46], L1T [48], LSST [49], AvgF, SymT [17], VTS [13], STR: STRUCK [50], AND KCF [3]

	Fused trackers								Proposed approaches						Selected state-of-the-art				
	ST	AFT	LOT	IVT	FCT	AMS	L1T	LSST	TPF_3	TPF_4	TPF_5	TPF_6	TPF_7	TPF_8	AvgF	SymT	VTS	STR	KCF
P1	.65	.70	.64	.64	.14	.72	.20	.22	.72	.70	.53	.67	.67	.50	.25	.35	.24	.75	.21
P2	.80	.76	.79	.82	.58	.23	.82	.50	.81	.80	.78	.74	.74	.73	.40	.65	.77	.68	.87
P3	.40	.70	.73	.57	.40	.47	.79	.75	.70	.68	.74	.70	.82	.82	.78	.81	.77	.78	.85
P4	.35	.43	.25	.26	.41	.30	.23	.31	.41	.41	.41	.42	.42	.42	.30	.33	.37	.39	.64
P5	.81	.82	.31	.64	.39	.39	.38	.75	.76	.71	.51	.70	.47	.57	.64	.72	.59	.68	.85
P6	.12	.14	.10	.13	.10	.29	.31	.10	.13	.12	.12	.12	.13	.12	.12	.12	.14	.13	.10
P7	.91	.79	.78	.91	.81	.81	.92	.93	.85	.86	.81	.85	.85	.89	.89	.89	.85	.80	.83
P8	.63	.72	.78	.43	.04	.60	.55	.41	.77	.77	.71	.65	.63	.58	.47	.62	.78	.74	.67
P9	.75	.74	.89	.81	.74	.75	.89	.83	.77	.83	.81	.80	.81	.82	.84	.84	.92	.82	.74
P10	.11	.15	.74	.11	.44	.70	.12	.12	.77	.75	.55	.48	.37	.76	.12	.12	.14	.91	.11
P11	.75	.74	.91	.81	.75	.12	.90	.92	.86	.85	.83	.77	.80	.82	.69	.90	.84	.54	.76
P12	.36	.52	.25	.35	.63	.33	.25	.35	.55	.40	.39	.40	.37	.35	.30	.33	.36	.28	.27
P13	.60	.63	.42	.55	.67	.48	.51	.52	.71	.63	.70	.68	.61	.67	.53	.55	.56	.69	.44
P14	.87	.68	.79	.85	.78	.73	.78	.87	.86	.83	.82	.76	.82	.76	.82	.83	.85	.82	.82
P15	.74	.80	.78	.74	.77	.61	.90	.89	.81	.76	.79	.78	.81	.82	.82	.83	.78	.69	.79
P16	.78	.79	.74	.56	.78	.80	.71	.92	.74	.85	.83	.82	.82	.89	.70	.77	.86	.78	.79
P17	.26	.22	.65	.37	.20	.31	.53	.28	.26	.37	.31	.30	.40	.36	.43	.41	.24	.16	.84
P18	.42	.33	.87	.10	.85	.79	.69	.86	.86	.89	.87	.81	.80	.85	.28	.38	.87	.84	.12
P19	.86	.81	.77	.87	.87	.56	.78	.86	.89	.87	.87	.90	.89	.89	.87	.88	.80	.76	.89
P20	.90	.79	.85	.89	.89	.58	.84	.79	.88	.85	.86	.85	.86	.85	.83	.85	.87	.78	.90
P21	.75	.73	.84	.67	.73	.78	.83	.83	.77	.79	.77	.73	.84	.74	.78	.79	.85	.75	.74
P22	.52	.42	.36	.52	.68	.80	.81	.13	.43	.54	.50	.51	.52	.50	.43	.54	.37	.13	.13
Mean	.57	.61	.65	.57	.58	.55	.62	.60	.70	.69	.66	.66	.66	.67	.56	.61	.62	.63	.61

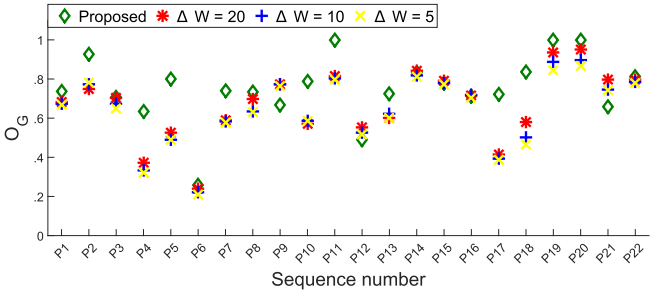


Fig. 9. Comparison of I_{ref} selected by the proposed approach and the original approach based on fixed temporal windows $\Delta W = 5, 10, 20$.

off-target. Using motion, TPF improves by 44%, 8%, and 85% (40% mean improvement). For P18, ST loses the target at frame 7 due to a similar background. TPF' assumes ST to be *on-target*, while AFT and LOT are labeled *off-target*. TPF uses motion to correctly label AFT and LOT *on-target* achieving an overall improvement of 420% in comparison with TPF.' ST remains *on-target* for the first six frames of the sequence, where TPF incorrectly labels it *off-target* in five out of the six frames, hence resulting in lower values of TPF. For P17, the target does not move for most of the sequences. ST and AFT lose the target at frame 45 due to a similar background and form a cluster. Due to the stationary target, TPF assumes the ST-AFT cluster to be *on-target* resulting in incorrect labels for all trackers, hence decreasing the performance by 35%. Globally, TPF improves TPF' by 2%, 4%, and 5% for ST, FT, and LOT, respectively.

2) Evaluation of Fast Approximation: Fig. 9 compares the proposed update for the reference frame I_{ref} with the original approach [23]. The average result for three trackers (ST, AFT, and LOT) is presented in terms of the overlap score O_G between the GT and the existing forward estimation in I_{ref} obtained by the tracker in [23] and TPF. TPF improves [23] in 16 out of 22 sequences. I_{ref} is updated only when tracker(s) are found to be *on-target*. For instance, all three trackers fail between frames 95–110 for P4. TPF detects and does not update I_{ref} after frame 110, whereas the tracker in [23] keeps moving I_{ref} forward, thus accumulating tracker errors. For P14–P16, TPF achieves tracking accuracy similar to the tracker in [23], while for P9 and P21, the three trackers are able to track the target throughout the sequences.

E. Combining Trackers

Performance comparisons are based on the overlap score O_G to measure the area overlap between the final target estimate and the GT data.

Table V compares the six TPF configurations ($\text{TPF}_3, \ldots, \text{TPF}_8$), showing that TPF_3 is the best and the average tracking accuracy decreases with the increasing number of trackers. There are two main reasons for this accuracy drop. First, low-performing trackers in the *on-target* cluster decrease the overall tracking accuracy when fused using the average. This is indicated by the results for P7, P9, and P15, where all trackers are *on-target* for most of the sequences. This can be further validated by a comparison of results with AvgF (Table V) for these sequences. Second, the number of splitting and merging of clusters naturally increases as we include more trackers, thus increasing the chances of wrong reverse-analysis evaluations. For P1, the target undergoes occlusions between frames 75–95.

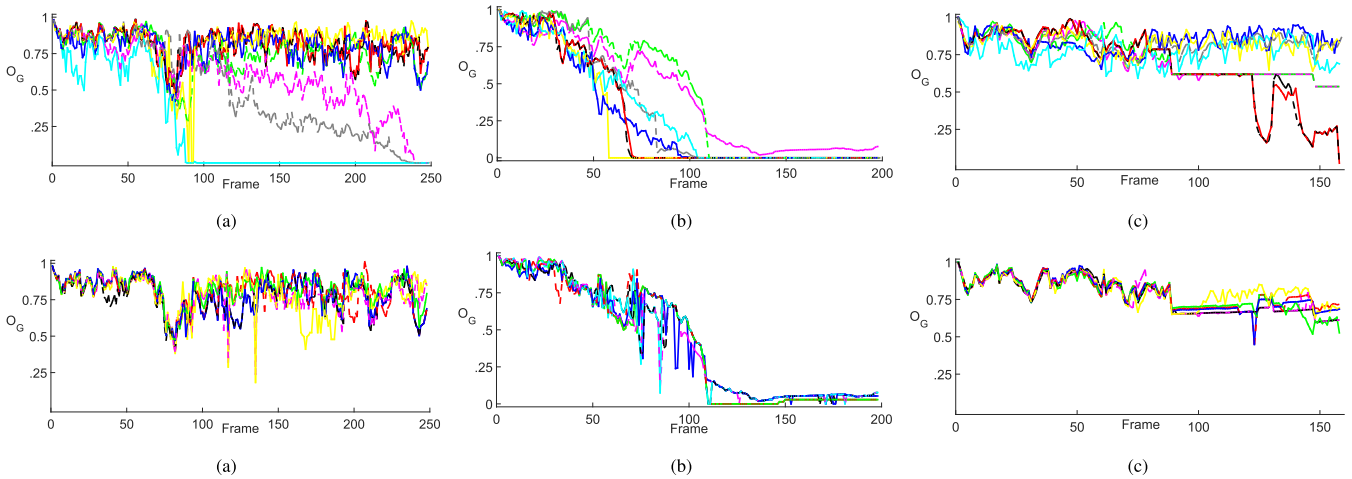


Fig. 10. O_G scores for trackers and TPF configurations under analysis for selected sequences. (a) Students-P2, (b) CAVIAR-P4, (c) MITTraffic-P21. Top row: Trackers; —: ST; - - -: AFT; —: LOT; - - -: IVT; —: AMS; - - -: FCT; —: L1T; —: LSST. Bottom row: TPF configurations; —: TPF_3; - - -: TPF_4; —: TPF_5; - - -: TPF_6; —: TPF_7; - - -: TPF_8.

FCT, L1T, and LSST lose the target due to occlusion; however, FCT (a deterministic tracker) achieves the best performance score during this interval, reducing the overall accuracy. The results for TPF_5 and TPF_8 indicate this scenario. Similarly for P10, a drop in accuracy of TPF_5, TPF_6, and TPF_7 occurs when the target undergoes occlusions between frames 20–35. All trackers lose the target at frame 20. However, LOT and AMS regain the target. The target remains stationary from frame 45 till the end of the sequence. This scenario allows failed trackers to achieve a higher performance score during reverse analysis, hence reducing the tracking accuracy. TPF removes low-performing trackers to improve the overall tracking accuracy. The results for P3, P10, P13, P18, and P19 indicate that TPF outperforms all the trackers. Furthermore, TPF has a performance similar to the best performing tracker(s) for the other sequences except for P6, P12, P17, and P22. TPF_3 achieves an overall improvement of 23%, 15%, 8%, 23%, 21%, 27%, 13%, and 17% in O_G in comparison with the individual trackers ST, AFT, LOT, IVT, FCT, AMS, L1T, and LSST, respectively. Moreover, all other TPF configurations ($TPF_4, \ldots, TPF_8$) also achieve better results compared with the eight trackers.

Fig. 10 compares the tracker accuracy using O_G values for selected sequences. The target in P2 changes its pose, causing AMS and L1T to lose the target between frames 80–90. Both failing trackers at this point are discarded by TPF. The performance of FCT and LSST drops gradually after frame 130 due to background clutter. The performance of TPF_7 drops at frame 140, where the output is corrupted by low-performing trackers (FCT and LSST), which are incorrectly labelled as *on-target*, while other TPF configurations make use of the best performing trackers. For P4, all trackers lose the target between frames 60–110. All TPF configurations identify and achieve accuracy close to the best performing tracker (AFT). However, TPF fails when all trackers are off-target. P21 undergoes scale changes as it moves away from the camera. O_G for all TPF configurations drops after frame 80, since all trackers remain *on-target* and form a single cluster. After frame 130, O_G for ST and IVT drops significantly since the trackers cannot handle scale changes. However, these trackers are discarded by TPF, while the performance for TPF_7 further improves as a better performing tracker (L1T) is added to the framework.

F. Comparison With the State-of-the-Art Approaches

Table V compares the TPF configurations and the related SOA. AvgF and SymT have been tested using the eight trackers. STRUCK is the best for P1 and P10, achieving the best average results among the selected SOA approaches. KCF achieves the best results for P2, P3, P4, P5, P17, and P20. However, it is unable to handle occlusions as shown for P10, P12, and P22. SymT fails to determine a low-performing tracker, hence reducing the overall tracking accuracy. SymT achieves good performance when most of the trackers are accurate as indicated by the results for P7 and P15. TPF, on the other hand, is able to use the best performing trackers and the overall accuracy is not dependent on the percentage of successful trackers. VTS performs relatively well and shows the best results for P8, P9, and P21. However, it fails for P1 and P10 due to occlusions and for P17 and P22 due to similarly colored background. Although the SOA approaches outperform some employed trackers (ST, IVT, FCT, and AMS; see Table V), TPF_3 shows an overall improvement of 23%, 15%, 13%, 11%, and 15% in O_G in comparison with AvgF, SymT, VTS, STRUCK and KCF, respectively.

Sample tracking results are shown in Figs. 11 and 12, where it can be seen that TPF correctly discards wrong trackers as they start failing. For clarity, we present only comparisons between TPF_3 and the SOA. Examples in Fig. 11 show that all the trackers correctly follow the target at the beginning of the sequence. As target occlusions are more frequent, only STRUCK is able to perform similarly to TPF_3, as seen in

TABLE VI
COMPUTATIONAL COST OF THE TRACKERS AND THE PROPOSED APPROACH (TPF) MEASURED IN FRAMES PER SECOND

	Fused Trackers								Proposed approaches					
	ST [42]	AFT [43]	LOT [44]	IVT [45]	FCT [47]	AMS [46]	L1T [48]	LSST [49]	TPF_3	TPF_4	TPF_5	TPF_6	TPF_7	TPF_8
FPS	50.1±0.1	5.9±4.0	0.3±0.1	48.1±1.4	8.0±6.2	108.0±31.2	7.7±1.7	3.4±0.6	2.5±4.8	0.3±0.4	0.2±0.3	0.2±0.1	0.1±0.1	0.1±0.1

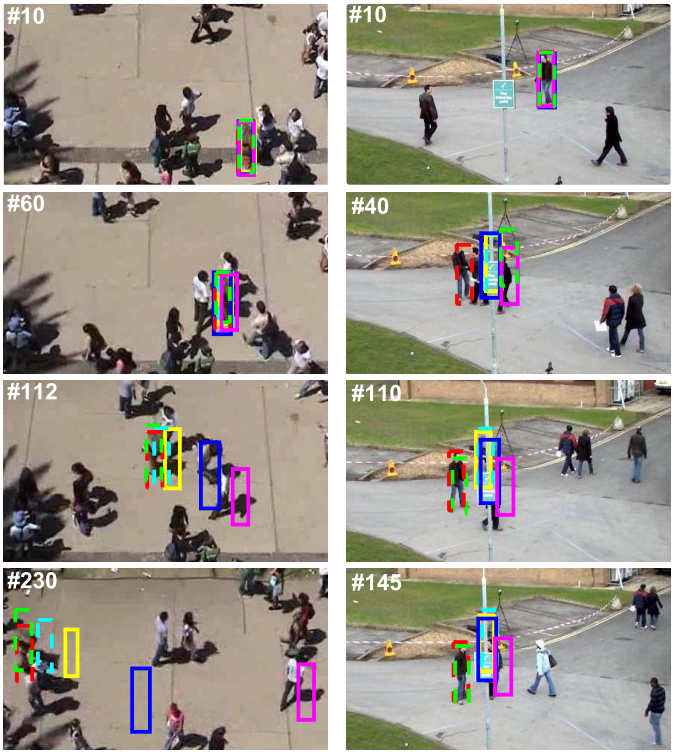


Fig. 11. Sample tracking results for Students-P1 (left column) and PETS-P10 (right column). - - -: TPF; - - -: STRUCK; —: VTS; —: SymT; —: AvgF; —: KCF.

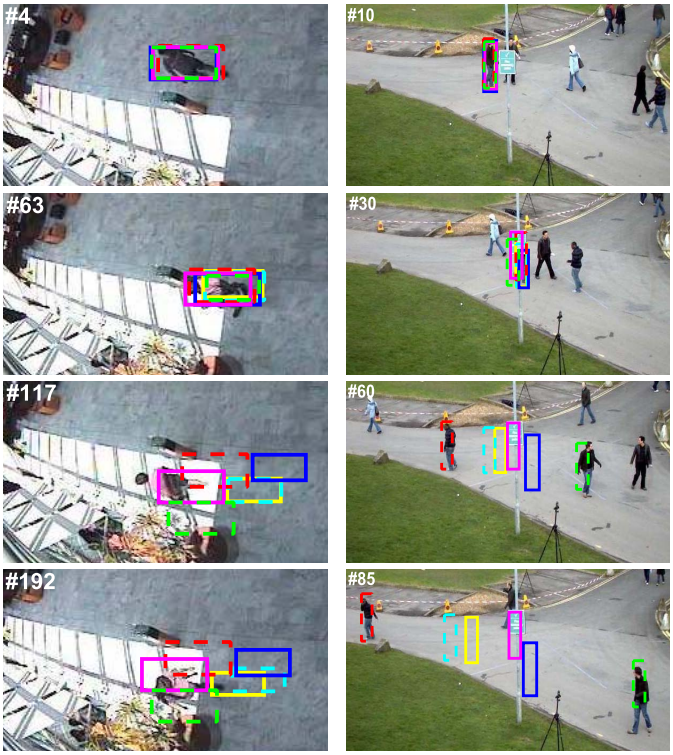


Fig. 12. Sample tracking results for CAVIAR-P4 (left column) and PETS-P12 (right column). - - -: TPF; - - -: STRUCK; —: VTS; —: SymT; —: AvgF; —: KCF.

frame 230 for P1 and frame 145 for P10. The right column of Fig. 12 (right) depicts the situation where only TPF_3 is able to adapt to changes in target scale and occlusions, whereas all the compared trackers fail, as seen in frame 85. Fig. 12(left) shows an example where none of the trackers obtain accurate position estimations after an illumination change (frames 117 and 192) and the best trackers (KCF, STRUCK, and TPF_3) achieve low accuracy.

G. Computational Cost

Fig. 13 presents the cost for the *trackers*, *tracker clustering* (Section IV), and *on-target cluster identification* (Section V) in terms of average computational time. The cost of the fusion stage is negligible and therefore ignored. The cost of the *trackers* considers running in parallel the trackers to fuse and depends on the employed approaches, being heavily influenced by the slowest tracker (LOT). The computational time for *tracker clustering* slightly increases with the number of trackers. Since *on-target cluster identification* uses reverse analysis, the computational time becomes dependent on the trackers in the *on-target* cluster $\mathbb{C}_t^*$ and the tracking challenges present in the sequence. This trend

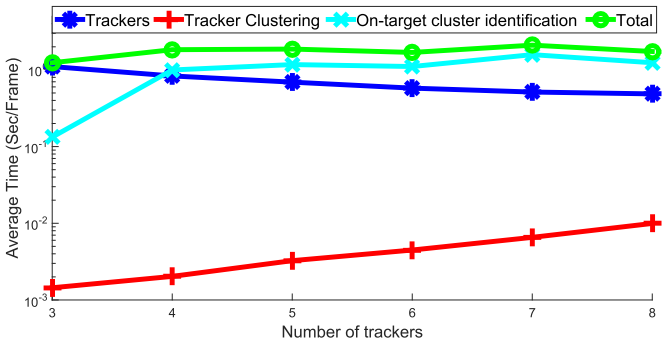


Fig. 13. Average computational time for the stages of the proposed approach. For each configuration, the average is computed over the complete data set and the total number of trackers.

is also highlighted by the overall cost for the TPF configurations presented in Table VI, where TPF_3 achieves the best computational cost. Fig. 14 shows the average number of trackers used by TPF, highlighting its advantage to cluster trackers, and using only the ones *on-target* for the various TPF combinations.

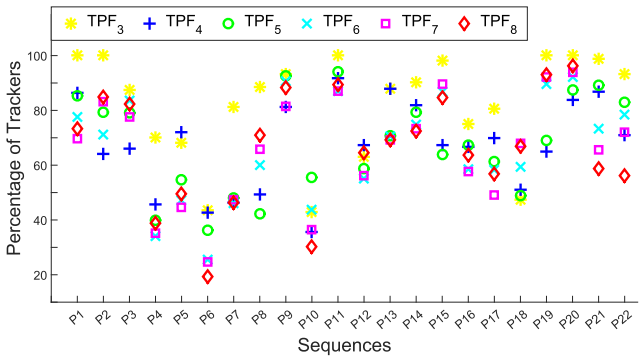


Fig. 14. Percentage of trackers used by the proposed approach for different tracker combinations.

VII. Conclusion

We presented an approach to dynamically select and combine the results of successful (i.e., *on-target*) trackers in a decision-level fusion framework. The proposed approach determines the relationships between trackers by analyzing the position and direction of movement of their estimated states. These spatiotemporal features are combined to estimate pair-wise tracker correlation scores that determine clusters of similarly performing trackers over time. An adaptive online evaluator identifies the trackers that are *on-target* and propagates them over time until a split or merge of this group (cluster) of trackers is detected. The final target state is estimated by fusing the outputs from the trackers that are in the *on-target* cluster. The experimental results show that the proposed approach outperforms state-of-the-art methods and the combined trackers. Moreover, the proposed time-reversed evaluation improves the original approach using motion analysis and tracker performance to temporally update the reference frame.

As future work, we will include the performance weight of each tracker in the fusion stage.

References

[1] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, Jul. 2014.

[2] J. Wu, S. Hu, and Y. Wang, "Adaptive multifeature visual tracking in a probability-hypothesis-density filtering framework," *Signal Process.*, vol. 93, no. 11, pp. 2915–2926, Nov. 2013.

[3] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.

[4] E. Erdem, S. Dubuisson, and I. Bloch, "Visual tracking by fusing multiple cues with context-sensitive reliabilities," *Pattern Recognit.*, vol. 45, no. 5, pp. 1948–1959, May 2012.

[5] V. Badrinarayanan, P. Perez, F. Le Clerc, and L. Oisel, "Probabilistic color and adaptive multi-feature tracking with dynamically switched priority between cues," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.

[6] M. Spengler and B. Schiele, "Towards robust multi-cue integration for visual tracking," *Mach. Vis. Appl.*, vol. 14, no. 1, pp. 50–58, Apr. 2003.

[7] I. Leichter, M. Lindenbaum, and E. Rivlin, "A general framework for combining visual trackers—The 'black boxes' approach," *Int. J. Comput. Vis.*, vol. 67, no. 3, pp. 343–363, May 2006.

[8] X. Zhang, W. Hu, H. Bao, and S. Maybank, "Robust head tracking based on multiple cues fusion in the kernel-Bayesian framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 7, pp. 1197–1208, Jul. 2013.

[9] E. Maggio and A. Cavallaro, "Hybrid particle filter and mean shift tracker with adaptive transition model," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 2. Mar. 2005, pp. 221–224.

[10] C. Chang, R. Ansari, and A. Khokhar, "Multiple object tracking with kernel particle filter," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 1. Jun. 2005, pp. 566–573.

[11] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "PROST: Parallel robust online simple tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 723–730.

[12] J. H. Yoon, D. Y. Kim, and K.-J. Yoon, "Visual tracking via adaptive tracker selection with multiple features," in *Proc. 12th Eur. Conf. Comput. Vis.*, Oct. 2012, pp. 28–41.

[13] J. Kwon and K. M. Lee, "Tracking by sampling and integrating multiple trackers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1428–1441, Jul. 2014.

[14] M. Heber, M. Godec, M. Rüther, P. M. Roth, and H. Bischof, "Segmentation-based tracking by support fusion," *Comput. Vis. Image Understand.*, vol. 117, no. 6, pp. 573–586, Jun. 2013.

[15] N. T. Siebel and S. J. Maybank, "Fusion of multiple tracking algorithms for robust people tracking," in *Proc. 7th Eur. Conf. Comput. Vis.*, May 2002, pp. 373–387.

[16] F. Moreno-Noguer, A. Sanfeliu, and D. Samaras, "Dependent multiple cue integration for robust tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 4, pp. 670–685, Apr. 2008.

[17] Y. Gao, R. Ji, L. Zhang, and A. Hauptmann, "Symbiotic tracker ensemble toward a unified tracking framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 7, pp. 1122–1131, Jul. 2014.

[18] T. A. Biresaw, A. Cavallaro, and C. S. Regazzoni, "Tracker-level fusion for robust Bayesian visual tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 25, no. 5, pp. 776–789, May 2015.

[19] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.

[20] J. C. SanMiguel, A. Cavallaro, and J. M. Martínez, "Adaptive online performance evaluation of video trackers," *IEEE Trans. Image Process.*, vol. 21, no. 5, pp. 2812–2823, May 2012.

[21] K. Shearer, K. D. Wong, and S. Venkatesh, "Combining multiple tracking algorithms for improved general performance," *Pattern Recognit.*, vol. 34, no. 6, pp. 1257–1269, Jun. 2001.

[22] E. Maggio, F. Smerladi, and A. Cavallaro, "Adaptive multifeature tracking in a particle filtering framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 10, pp. 1348–1359, Oct. 2007.

[23] H. Wu, A. C. Sankaranarayanan, and R. Chellappa, "Online empirical evaluation of tracking algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1443–1458, Aug. 2010.

[24] J. Triesch and C. Malsburg, "Democratic integration: Self-organized integration of adaptive cues," *Neural Comput.*, vol. 13, no. 9, pp. 2049–2074, Sep. 2001.

[25] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *Proc. 20th Int. Conf. Pattern Recognit.*, Aug. 2010, pp. 2756–2759.

[26] T. A. Biresaw, A. Cavallaro, and C. S. Regazzoni, "Correlation-based self-correcting tracking," *Neurocomputing*, vol. 152, no. 1, pp. 345–358, Mar. 2015.

[27] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 1998, pp. 232–237.

[28] S. Zhang, H. Yao, X. Sun, and X. Lu, "Sparse coding based visual tracking: Review and experimental comparison," *Pattern Recognit.*, vol. 46, no. 7, pp. 1772–1788, Jul. 2013.

[29] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2042–2049.

[30] Z. Hong, X. Mei, D. Prokhorov, and D. Tao, "Tracking via robust multi-task multi-view joint sparse representation," in *Proc. Int. Conf. Comput. Vis.*, Dec. 2013, pp. 649–656.

[31] Q. Li, X. Wang, W. Wang, Y. Jiang, Z.-H. Zhou, and Z. Tu, "Disagreement-based multi-system tracking," in *Proc. Asian Conf. Comput. Vis.*, vol. 7729. Nov. 2012, pp. 320–334.

[32] B. Stenger, T. Woodley, and R. Cipolla, "Learning to track with multiple observers," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 2647–2654.

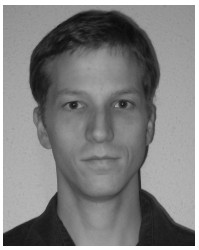
[33] B. Zhong, H. Yao, S. Chen, R. Ji, T.-J. Chin, and H. Wang, "Visual tracking via weakly supervised learning from multiple imperfect oracles," *Pattern Recognit.*, vol. 47, no. 3, pp. 1395–1410, Mar. 2014.

[34] J. Whitehill, T.-F. Wu, J. Bergsma, J. R. Movellan, and P. Ruvolo, "Whose vote should count more: Optimal integration of labels from labelers of unknown expertise," in *Proc. Adv. Neural Inf. Process. Syst.*, Dec. 2009, pp. 2035–2043.
[35] C. Bailer, A. Pagani, and D. Stricker, "A superior tracking approach: Building a strong tracker through fusion," in *Proc. 13th Eur. Conf. Comput. Vis.*, vol. 8695. Sep. 2014, pp. 170–185.
[36] A. J. C. Sharkey, N. E. Sharkey, U. Gerecke, and G. O. Chandroth, "The 'test and select' approach to ensemble combination," in *Multiple Classifier Systems*, vol. 1857. Heidelberg, Germany: Springer Berlin Heidelberg, Jun. 2000, pp. 30–44.
[37] N. Anjum and A. Cavallaro, "Multifeature object trajectory clustering for video analysis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1555–1564, Nov. 2008.
[38] J. F. Lucas, *Introduction to Abstract Mathematics*. Lanham, MD, USA: Rowman and Littlefield, 1990.
[39] J. H. Conway and R. K. Guy, *The Book of Numbers*. New York, NY, USA: Springer, 1996.
[40] F. Jiang, Y. Wu, and A. K. Katsaggelos, "A dynamic hierarchical clustering method for trajectory-based unusual video event detection," *IEEE Trans. Image Process.*, vol. 18, no. 4, pp. 907–913, Apr. 2009.
[41] M. Yang, Y. Wu, and G. Hua, "Context-aware visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1195–1209, Jul. 2009.
[42] D. Wang, H. Lu, and M.-H. Yang, "Online object tracking with sparse prototypes," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 314–325, Jan. 2013.
[43] E. Erdem, S. Dubuisson, and I. Bloch, "Fragments based tracking with adaptive cue integration," *Comput. Vis. Image Understand.*, vol. 116, no. 7, pp. 827–841, Jul. 2012.
[44] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," *Int. J. Comput. Vis.*, vol. 111, no. 2, pp. 213–228, Jan. 2014.
[45] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, May 2008.
[46] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Scale and orientation adaptive mean shift tracking," *IET Comput. Vis.*, vol. 6, no. 1, pp. 52–61, Jan. 2012.
[47] K. Zhang, L. Zhang, and M. Yang, "Fast compressive tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 2002–2015, Oct. 2014.
[48] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2259–2272, Nov. 2011.
[49] D. Wang, H. Lu, and M.-H. Yang, "Least soft-threshold squares tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2013, pp. 2371–2378.
[50] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.

ObaidUllah Khalid received the B.S. degree in computer engineering from Sir Syed University of Engineering and Technology, Karachi, Pakistan, in 2004, and the M.S. degree in network engineering from Illinois Institute of Technology, Chicago, IL, USA, in 2008.

He served as a Lecturer with National University of Sciences and Technology (NUST), Karachi. Since 2012, he has been with Queen Mary University of London, London, U.K., and Alpen-Adria Universität Klagenfurt, Klagenfurt, Austria, as a Ph.D. Researcher, under the supervision of Prof. A. Cavallaro and Prof. B. Rinner. His research interests include tracker-level fusion, online performance evaluation, and video tracking.

Mr. Khalid received the NUST Scholarship for his M.S. studies and the Erasmus Mundus Fellowship for his double doctorate in interactive and cognitive environments.



Juan Carlos SanMiguel received the Ph.D. degree in computer science and telecommunication from University Autonoma of Madrid, Madrid, Spain, in 2011.

He was a Post-Doctoral Researcher with Queen Mary University of London, London, U.K., from 2013 to 2014, under a Marie Curie IAPP Fellowship. He is currently an Assistant Professor with University Autonoma of Madrid and a Researcher with the Video Processing and Understanding Laboratory. His research interests include computer vision with a focus on online performance evaluation and multicamera activity understanding for video segmentation and tracking.

Andrea Cavallaro received the Ph.D. degree in electrical engineering from Swiss Federal Institute of Technology, Lausanne, Switzerland, in 2002.

He was a Research Fellow with BT Group PLC, London, U.K., in 2004. He is currently a Professor of Multimedia Signal Processing and the Director of the Centre for Intelligent Sensing at Queen Mary University of London, London. He has authored over 160 journal and conference papers, one monograph on video tracking (Wiley, 2011), and three edited books entitled *Multi-Camera Networks* (Elsevier, 2009), *Analysis, Retrieval and Delivery of Multimedia Content* (Springer, 2012), and *Intelligent Multimedia Surveillance* (Springer, 2013).

Dr. Cavallaro received the Royal Academy of Engineering Teaching Prize in 2007; three student paper awards on target tracking and perceptually sensitive coding at the IEEE International Conference on Acoustics, Speech, and Signal Processing in 2005, 2007, and 2009; and the best paper award at the IEEE Advanced Video and Signal Based Surveillance Conference in 2009. He is an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and a member of the Editorial Board of *IEEE Multimedia*. He was an Area Editor of *IEEE Signal Processing Magazine* and an Associate Editor of IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE TRANSACTIONS ON SIGNAL PROCESSING, and *IEEE Signal Processing Magazine*.