Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 2246 – 2259

doi:10.3906/elk-1709-245

Long-term multiobject tracking using alternative correlation filters

Kemal Batuhan BAŞKURT*, **Refik SAMET**
Department of Computer Engineering, Faculty of Engineering, Ankara University, Ankara, Turkey

Received: 27.09.2017 • **Accepted/Published Online:** 26.05.2018 • **Final Version:** 28.09.2018

Abstract: We propose a real-time multiobject-tracking approach that is minimally affected by environmental conditions and target appearance change. The aim of the proposed approach is to track any object in a scene, regardless of object type, since tracking all of the objects in a scene is critical and widely used in surveillance applications. Thus, motion detection results are used to initialize the trackers. The proposed object-tracking approach is realized with two types of independent correlation filters estimating location and scale. Alternative correlation filters representing different appearances of the target are also proposed in order to increase the robustness of the approach to scene and target changes. Tracking sustainability is provided by putting alternative correlation filters into use when the quality of the tracking output decreases to a critical level. Motion blobs are also used to minimize object boundary drift, which is a challenging problem, especially for long-term tracking. The proposed approach was tested on an object-tracking benchmark dataset and outperformed most state-of-the-art methods.

Key words: Correlation filter, visual object tracking, real-time video processing

1. Introduction

Rapid incrementing of the use of surveillance cameras in daily life has resulted in the need to find effective video processing methods. Visual object tracking is an important part of video processing for surveillance cameras. Although visual object tracking has been studied for decades, it is still a challenging problem because of difficulties posed by environmental conditions. Factors such as illumination variation, camera motion, and appearance variation of a target according to angle of view are problematic for existing methods, and so the development of a robust method that is able to track any type of object in an uncontrolled environment has gained importance in this scope.

Visual object-tracking studies can be divided into two main groups, namely off-line and on-line learning trackers. Off-line methods need to be trained properly regarding the target object, and the target object is then tracked automatically once it has been detected using the trained model. The main problem with these methods is that the trained model might be insufficient after a while because of changes in target appearance. Retraining for handling changes is time-consuming and makes the trained model too complicated for real-time processing. Furthermore, all target object types need to be pretrained, as only trained objects are detected. On-line methods are aimed at tracking objects whose initial positions are provided to the tracker because there is no preliminary information about the target object. They have more general usage as they can track any type of target.

*Correspondence: batuhanbaskurt@gmail.com

An on-line and real-time correlation filter-based object-tracker approach is proposed in this article. The main motivation of the proposed approach is to achieve robust long-term tracking against challenging environmental conditions. In the proposed approach, multiple objects can be tracked at the same time by an object tracker pool. Motion detection output is used to trigger the tracker and provide the ability to track any type of moving object, and the motion information is also used to increase the robustness of correlation filter-based visual object tracking when object drifting occurs. The proposed approach adapts the appearance change of the target using on-line updating of the target object model while tracking. Moreover, alternative correlation filters representing different appearances of each target are proposed in order to handle appearance variation and are activated when the reliability of the main correlation filter decreases in situations where the target's appearance might have changed. Continuity of long-term tracking is achieved in this way.

The paper is organized as follows. Section 2 contains a description of related visual object-tracking methods, and details of the proposed approach are presented in Section 3. Section 4 presents the implementation details of the proposed approach, while experimentation and a discussion are provided in Section 5. Section 6 contains conclusions on the study.

2. Related work

Object tracking has been researched and in demand for many years. There are many studies in the literature aimed at finding solutions to the various problems of object tracking. There are extensive surveys on this subject. McCall et al. [1] compared object-tracking methods for vehicle tracking when changing lanes in driver assistance systems, while Santiago et al. [2] analyzed object-tracking approaches in sporting events for the statistical analysis of athletes. Conversely, Smuelders et al. [3] compared object-tracking methods from a more general perspective, regardless of target type, on different datasets.

A common approach to object-tracking methods is searching for the target object in a window formed around the current position in the next video frame. The region with the highest similarity between reference object and target object, regarding the different metrics used in the search window, is determined as the new location of the object [4–6]. Some studies update the target model based on each new video frame by taking positive and negative samples from around the object instead of just considering the target object boundary, and the tracking model is supported by redetection after certain periods [7,8]. The difference between these methods is based on object detection, as Hare et al. [7] trained a structured-SVM using the samples of the target while Kalal et al. [8] built a discriminative classifier for on-line learning. The robustness of the methods to appearance change is increased by updating the model with positive and negative samples.

However, an increase in the number of samples learned on-line also decreases the run-time performance of the methods. To address the run-time performance problem, Bolme et al. [9] proposed transferring the correlation process of a window search to the frequency domain, which provides fast position localization. In [10], a separate correlation filter was added to estimate the scale by emphasizing that only the location estimate is inadequate. Li and Zhu [11] also addressed the scale change problem by sampling the target with different scales. Their multiple-feature integration, which combines raw pixel, histogram of oriented gradient (HOG) [12], and color information, increased the robustness against the challenging scenarios. Vojir et al. [13] adopted a mean-shift algorithm, which is commonly used in object tracking to address the scale adaptation problem. Distance between RGB histograms of the reference and the candidate region was used to estimate object scale. Their simple approach provided a fast and robust solution for the scale adaptation problem, even though it still had problems in challenging conditions such as illumination variation. Hong et al. [14] inferred that human

memory and object tracking are realized by two different models: long-term and short-term. The long-term model is updated by the short-term model only if predefined update conditions are satisfied, and attempts to deal with the appearance change of the target in long-term tracking are handled in this way.

Object-tracking methods via off-line learning usually use deep learning methods, which have been intensively studied in recent years. Danelljan et al. proposed a novel approach to use deep convolutional features instead of commonly used hand-crafted features [15]. Another pioneering study in this area is MDNet, which pretrains a convolutional neural network to represent and track the target object [16]. The tracking accuracy of these methods is high, but their computational complexity is also high and thus the run-time performance of these methods can be low. High computational capacity graphics cards using GPU programming techniques are required to perform these approaches.

As a consequence, off-line tracking's pretraining requirement makes the approach hard to adapt to environmental and target appearance changes. Although approaches based on convolutional neural networks build more generic tracking models, their computational complexity is still an important problem to apply to video surveillance cameras in real time. On the other hand, on-line tracking approaches provide a light-weight solution that can be performed in real time, but their tracking accuracy still needs to be improved, especially in challenging uncontrolled environments.

3. The proposed approach

The aim of this study is to track any moving object in a scene. A real-time and on-line learning-based tracker is proposed for this purpose, since off-line learning-based trackers require pretrained data for the target object type. Motion detection results are used to trigger object tracking, after which two different correlation filters, respectively responsible for location and scale estimation, are used [10,17]. According to previous research, feeding the tracker with both HOG features [12] and pixel values increases performance [10]. Thus, HOG features using a cell size of 4×4 pixels are used as input and alternative correlation filters are used for handling appearance variation in the proposed approach. The proposed correlation filter-based approach is able to track multiple objects at the same time. Details of the proposed approach are explained in three categories: motion detection, the baseline, and the proposed approach.

3.1. Motion detection

A pixel-based adaptive segmenter (PBAS) [18] is used for motion detection in the proposed approach. The PBAS performs pixel-based analysis of a scene in the time domain in order to adaptively learn the dynamic environment. A complex background with both crowded and calm regions at the same time is handled using pixel-based analysis. The PBAS was chosen because of its robustness to scene changes, high adaptability, and real-time running performance. The PBAS labels motion pixels in a scene (as shown in Figure 1), after which connected component analysis is applied to obtain motion blobs from the motion pixels. Finally, the motion blobs are used to feed the object tracker.

3.2. Baseline

Our approach is based on discriminative correlation filter-based tracking as proposed by Danelljan et al. [10]. First, two correlation filters, respectively responsible for location and scale estimation, are created using the given initial target position. The best correlation between the filter and candidate regions in the following video

Figure 1. Motion detection result.

frame is solved as an error minimization problem. Finally, the correlation filters are updated with the new position of the target. Details of the baseline are explained in the following subsections.

3.2.1. Correlation filter initialization

Correlation filter-based trackers need an initial boundary f of the target to start tracking, provided that the target appearance has been modeled using a correlation filter, which is applied around the current location of the target in the next video frame. Eight random affine transformations f_{i} of initial boundary f are created for use as an input series. Next, desired correlation output g_{i} is constructed using a Gaussian function $(\sigma = 2.0)$, with its peak located at the target center for each f_{i}. Correlation on the spatial domain corresponds to element-wise multiplication on the frequency domain according to the convolution theorem as follows:

$$G = F\,H^{*}, \tag{1}$$

$$H_{\mathrm{i}}^{*} = \frac{G_{\mathrm{i}}}{F_{\mathrm{i}}}, \tag{2}$$

where $F = F(f)$, $H = F(h)$, h is the correlation filter, and F represents fast Fourier transformation. Thus, the correlation operation is performed in the frequency domain, then converted back to the spatial domain in order to gain run-time performance and achieve real-time processing. The filter minimizing error between current output $F_{\mathrm{i}}H^{*}$ and desired output G_{i} is calculated in order to find the correlation between training inputs F_{i} and desired outputs as follows:

$$H = \sum_{i} |F_i H^{*} - G_i|^2. \tag{3}$$

3.2.2. On-line update

The target object's appearance regularly changes because of rotation, scale, illumination variation, angle of view, etc., and the correlation filter created using the initial target boundary is updated during tracking to handle target appearance change. While the correlation filter is adapted for appearance changes in the ratio of update coefficient η, it also stays consistent with the target history as follows:

$$H_{\mathrm{t}}^{*} = \frac{A_{\mathrm{t}}}{B_{\mathrm{t}}}, \tag{4}$$

$$A_{\mathrm{t}} = \eta G_{\mathrm{t}} F_t^* + (1-\eta) A_{\mathrm{t}-1}, \tag{5}$$

$$B_{\mathrm{t}} = \eta F_{\mathrm{t}} F_{\mathrm{t}}^* + (1-\eta) B_{\mathrm{t}-1}, \tag{6}$$

where t indicates current filter and $t-1$ points to the previous value of the filter. Peak to side-lobe ratio (PSR) is used to measure the robustness of the correlation filter. Correlation output g is divided into two regions: a peak with the highest value and a side-lobe containing the rest. A peak region is assumed to be an 11 $\times$ 11 pixel area around the peak. Subsequently, the PSR is calculated as:

$$PSR = \frac{g_{\max} - \mu_{\mathrm{sl}}}{\sigma_{\mathrm{sl}}}, \tag{7}$$

where $g_{\max}$ represents the value in the peak region and μ_{sl} and σ_{sl} represent the average and standard deviation of the side-lobe region, respectively. The correlation filter is only updated with the current target when the PSR value is within the confidence level range. Furthermore, $PSR_{\min}$ is used as the minimum reliable threshold.

3.2.3. Scale estimation

The scale estimation of the target is performed after the location estimation is realized in the new video frame. It is carried out by finding the best correlation between the current target and samples of the scale pyramid initialized at the beginning. The same series of equations, Eqs. (1)–(7), used for location estimation are applied, the unique difference being that the best correlation is searched for in the scale pyramid instead of the target neighborhood.

3.3. The proposed multiobject tracker

Sections 3.1 and 3.2 explained the existing methods used in the proposed approach. This section explains the novelty and the contribution of the proposed approach. A motion-triggered multiobject tracker pool, the contribution of the alternative correlation filters against variation of environmental conditions, and the proposed solution for boundary drifting problem are explained in this section. The pseudocode of the proposed approach is provided in Algorithms 1–3.

3.3.1. The motion-triggered multiobject tracker

On-line learning-based object trackers need the target's initial position to start. Our approach uses motion blobs as input, where each one is associated with an object tracker and is operated by the tracker while it is visible in the scene. Each tracker T contains main correlation filter H, alternative correlation filters for different appearances $HA_{\mathrm{a}}\,(a \in 1, 2, \ldots, A)$, target trajectory $Tr_{\mathrm{l}}\,(l \in 1, 2, \ldots, L)$, and status (S) for the active/inactive state of the tracker; thus:

$$T = \{HHA_{1,2,\ldots,\mathrm{A}}, Tr_{1,2,\ldots,\mathrm{L}} S\}. \tag{8}$$

Tracker pool $P = \{T_1 T_2, \ldots, T_{\mathrm{N}}\}$ is created, which includes each tracker associated with an object in the scene. A tracker whose target has disappeared or has become occluded becomes inactive instead of being deleted directly. Inactive trackers are kept in the pool for a period of time t_{passive}. Next, an attempt is made to associate new motion blobs in each frame with both active and inactive trackers. An occluded or turned-back

Algorithm 1 Multiobject tracker.

Initialize active $P \leftarrow \emptyset$ and inactive $P_{\text{inactive}} \leftarrow \emptyset$ tracker pools

for each video frame F **do**
  Initialize updated tracker list $T_{\text{updated}} \leftarrow \emptyset$
  Get motion blob list B by applying PBAS to F
  for each motion blob b in B **do**
    Initialize matched tracker list $T_{\text{matched}} \leftarrow \emptyset$
    $T_{\text{matched}} \leftarrow$ **call** Find-Match with b, P
    if T_{matched} **equal** $\emptyset$ **then**
      $T_{\text{matched}} \leftarrow$ **call** Find-Match with b, P_{inactive}
    end if
    if T_{matched} **not equal** $\emptyset$ **then**
      Find best matched candidate tracker T_{c} using Eq. (9)
      call Update-Tracker for T_{c} with b using Eqs. (4)–(6)
      Push T_{c} to T_{updated}
    else
      Create new tracker with b using Eq. (1)
    end if
  end loop
  call Update-Tracker for each T **in** P and T not in T_{updated}
  Update P_{inactive}
end loop

Algorithm 2 Find-Match.

Input: motion blob b, tracker pool P
Output: matched tracker list T_{matched}

for each tracker T **in** P **do**
  if rectangle(b) **intersect** rectangle(T) **then**
    Push T to T_{matched}
  end if
end loop

Algorithm 3 Update-Tracker.

Input: tracker T, motion blob b
Output: tracker T

Update correlation filter H using Eqs. (4)–(6)
Update alternative correlation filters HA using Eq. (11)
Check boundary drifting using Eq. (12)
if boundary drifting **exists then**
  Apply boundary correction using Eqs. (13)–(15)
end if

target is reassociated with the same tracker if it becomes visible again during t_{passive}. Hence, reactivating an old tracker instead of creating a new one for a recently disappeared object is important for long-term tracking, since

losing and regaining a target and creating more than one tracker for the same object decreases the robustness of the method.

In the proposed approach, we take motion blobs as input objects, and the association between them and the trackers is managed as follows. If there are motion blobs but no trackers in the pool, these motion blobs are considered as new targets, after which a new tracker for each target is created. On the other hand, if both new motion blobs and trackers exist, the tracker's object positions are compared with the motion blobs. When a tracker and blob are matched, the tracker is updated by the motion blob rectangle using Eqs. (4)–(6). If a motion blob is intersected by more than one tracker, the most highly correlated one (with minimum correlation error) is selected as a match with regard to Eq. (3). Motion blobs without any tracker intersections are first searched for in the inactive trackers of recently disappeared objects. If the correlation filter output of any inactive tracker is closer than error threshold E_{conf}, the current blob is assumed to be a recently disappeared object and is associated with the inactive tracker. In this case, the tracker is updated and its status becomes active. A new tracker is created for each motion blob that still cannot be associated with any of the existing trackers.

The correlation filters of existing trackers in the pool are updated, as performed in the baseline model using Eqs. (4)–(6), when there are no detected motion blobs in the scene. In the case of intersections with more than one object in the scene, the tracker pool provides a tracker comparison to make the best decision. Since false matching of intersected objects decreases the robustness of a method, communication between trackers makes matching more robust as more information is available compared to existing single-object tracker methods. The tracker with the minimum correlation filter error is associated with the blob at the intersection region, defined as

$$H = \min_{H} \{H_1, H_2, \ldots, H_{\text{K}}\} \tag{9}$$

where K is the number of intersected trackers. Thus, updating a tracker with data of other objects is minimized in our approach.

3.3.2. The alternative correlation filters

Existing correlation filter-based object-tracking methods generally estimate the target position using a single-object model, whereas the proposed approach creates alternative correlation filters for different appearances of the target. In order to handle the significant appearance variation, alternative filters are put into use when the target is lost or the PSR of the correlation filter is lower than PSR_{min}. Subsequently, Eq. (3) is applied to all alternative correlation filters. If one of the alternative filters HA_{a} attains a lower correlation error than the main filter, it replaces it and is updated as follows:

$$H = \min_{H} \{H, HA_{\text{a}}\}. \tag{10}$$

A candidate filter Hc is only added to the alternative filters when it is reliable and represents a different view than the existing ones. The PSR value of the candidate filter must be higher than the alternative filter with less PSR in the list. Filters with a PSR less than PSR_{min} are not added to the alternative filters, as explained in Section 3.2.2. The correlation filter's output error is used to determine the representation of different appearances of the target while confidential error threshold E_{conf} prevents object mismatch. Moreover, the error difference between the candidate filter and the main filter must be less than E_{conf}. Finally, the candidate's output error with regard to the main filter must be higher than the corresponding alternative filter in order to

replace it. Candidate filter H_C must fulfill all three conditions before being added to the alternative filters. Hence,

$$PSR_{\mathrm{H}_C} > PSR_{\mathrm{HA}_a}, \tag{11}$$

$$|H - H_{\mathrm{C}}| < E_{\mathrm{conf}},$$

$$H_{\mathrm{C}} > HA_{\mathrm{a}}.$$

The alternative filter with the lowest PSR is replaced with the new filter in the case of more than one alternative filter fulfilling the replacement conditions.

3.3.3. The boundary drift problem

A target's boundary drift from the correct position is one of the biggest problems reported in the literature. Drifting causes the inclusion of the background region into the target and decreases tracking robustness. Background features become dominant in the target area after a while, which makes the tracker fail by staying in the background.

In the proposed approach, we use motion detection results to minimize the boundary-drifting problem. The correlation filter is forced to move closer to the motion blob b when the distance between the center of target rectangle g and the filter is higher than the threshold defined using deviation factor m_{dev} through the target's width, as shown in Figure 2.

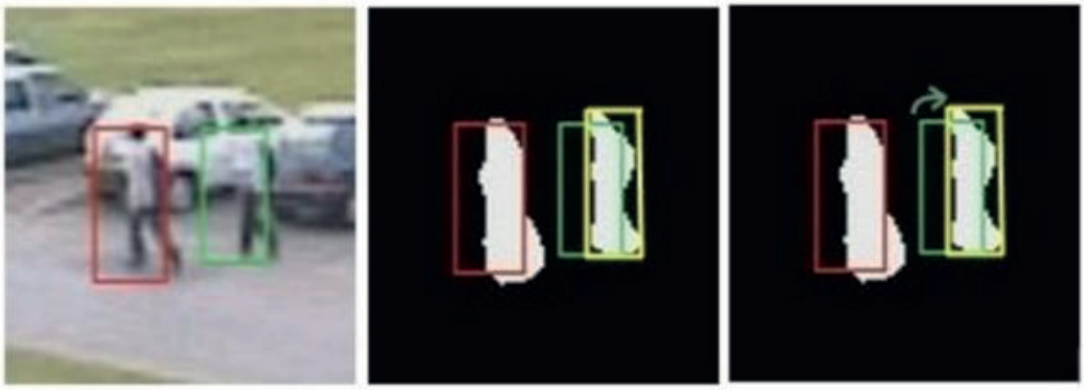

Figure 2. Difference between correlation filter output and a motion blob. The green rectangle represents the tracker estimation, while the yellow one is the motion blob of the same tracker.

$$width\,(g) > |center\,(b) - center\,(g)| > m_{\mathrm{dev}} * width(g), \tag{12}$$

where *width* represents the motion blob's width in pixels and *center* is the pixel coordinates of the center point of the motion blob. In the drifting case, a temporal correlation filter for update H_{update} is created and initialized using Eqs. (1) and (2) by a motion blob rectangle. Hence, Eqs. (4)–(6) are respectively redefined as:

$$H_{\mathrm{t}}^{*} = \frac{A_{\mathrm{u}}}{B_{\mathrm{u}}}, \tag{13}$$

$$A_{\mathrm{t}} = \eta G_{\mathrm{u}} F_{\mathrm{u}}^{*} + (1-\eta) A_{\mathrm{t}-1}, \tag{14}$$

$$B_{\mathrm{t}} = \eta F_{\mathrm{u}} F_{\mathrm{u}}^{*} + (1-\eta) B_{\mathrm{t}-1}. \tag{15}$$

The current correlation filter is updated using H_{update} instead of its current value in the video frame of index t and thus is forced to move close to the motion blob by H_{update} according to update coefficient η while its history is preserved. This operation is repeated for the next video frame if the deviation between the motion blob and the target boundary still remains. The boundary drifting problem is handled iteratively in this way until the difference between them becomes acceptable.

The proposed drift-handling method is a supplementary component of our object-tracking approach. Nevertheless, the performance decrement on object tracking caused by this method is not desired, so drift handling is excluded when the difference between the widths of the motion blob and the filter boundary is higher than the filter boundary width, as showed in Eq. (12). In this case, it is assumed that a particular motion blob represents more than one object, which may mislead the object tracking.

4. Implementation

One of the most important points of the proposed approach is its ability of real-time multiobject tracking, since robustness and real-time performance in multiobject tracking are challenging features to keep together. The proposed approach works in real time and is minimally affected by the number of objects in a scene at the same time; thus, a multithreaded programming approach was applied for this purpose. The tracker pool described in Section 3.3.1 was managed on the main thread and the correlation and update operations of each tracker were performed on separate threads created for each tracker, as shown in Figure 3. Each tracker thread notified the main thread of its results after finishing its operation.

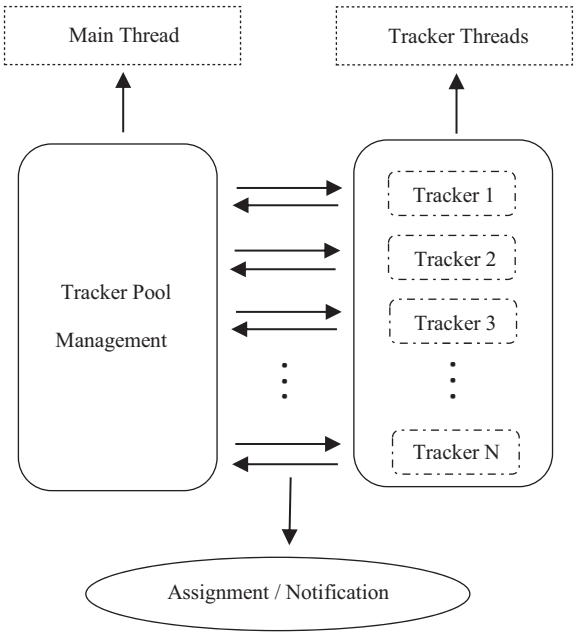

Figure 3. Multithreaded implementation.

The Table shows the effect of number of objects being tracked at the same time with an Intel Xeon Quad Core 3.40 GHz processor with 760 × 570 resolution images. The interval of the number of objects is determined by folding the CPU cores. The number of objects does not affect the run-time performance while the total number of threads is less than the number of cores, but performance decreases when a number of objects as resources must be shared between threads and more time is needed for process switching.

Table. Average frame processing time for a given number of objects.

Number of objects	Average frame processing time (ms)
1–8	17.2
8–16	23.8
> 16	35.4

The algorithm parameters used in the implementation are explained below. Correlation filter size $M \times N$ was initialized as double the motion blob size. Update coefficient η in Eqs. (5), (6), (14), and (15) was set at 0.025, as has been commonly used in the literature [10]. $PSR_{\min}$ was set at 5 for a reliable PSR threshold. The maximum number of alternative correlation filters for each tracker was limited to 4. This value has a direct effect on run-time performance, as a higher value increases computational complexity. Even though using a higher value might increase the robustness, it also should be limited for run-time performance. Inactive time t_{passive} is an application-level parameter for an object disappearing from a scene that controls the invisible time of any disappeared object to prevent the creation of redundant trackers for the same object, and it was set at 10 s. It has a minor effect on the robustness of the proposed approach. A target object that exited from a scene could reappear at a different point, as it might have changed position while out of the scene. Thus, the position intersection condition was not required for matching with inactive trackers (see Section 3.3.1); only the correlation filter output was used in this matching operation. Maximum correlation error threshold E_{conf} could be adjusted dynamically during run-time from its initial value (300) determined by experiments performed on the dataset. A list of correlation errors (E_{list}) was stored to dynamically determine the E_{conf} parameter. The correlation error calculated on each tracker was inserted into the error list. E_{conf} was updated as shown in Eq. (16) upon every 5th insertion:

$$E_{\text{conf}} = median\,(E_{\text{list}}) \times 2. \tag{16}$$

The size of the correlation error list was limited to 500 so as to not adversely affect run-time performance, and the oldest value in the list was deleted in the case of excess items.

Finally, a scale pyramid with a scale factor of 1.02 and 33 levels was created for scale estimation, which was the same as the baseline.

Furthermore, quality of the motion blobs has a direct effect on the accuracy of the proposed approach. False detections caused by the motion detection increases the complexity of the proposed approach as redundant trackers are created. Broken or partially missed detection of the object also decreases the tracking accuracy, especially for the first detection, as the correlation filter is created by the motion blob. In this sense, the PBAS minimizes the aforementioned defects with its pixel-based adaptation to the scene.

5. Experimental results

The proposed approach was compared with baseline [10], Muster [14], Struck [7], TLD [8], ASMS [13], and SAMF [11]. TLD and Struck are robust approaches that not only consider positive samples of the target object but also negative samples around the target to increase the robustness against appearance changes. Muster uses two different models instead of a single model to represent the target. Therefore, it is a good candidate to compare the effect of the multiple models on handling the target's appearance change problem. ASMS is the top-performing real-time tracker in terms of accuracy according to the visual object tracking challenge (VOT) [19]. SAMF is one of the top-performing trackers on pooled AR-rank of the same challenge. Experiments were

performed using surveillance sequences (captured by fixed camera) from the Object-Tracking Benchmark (OTB) dataset [20], which is one of the most popular datasets used in VOT. OTB contains 100 different recent image sequences from the literature for the evaluation of object tracking. The sequences are labeled in 11 categories according to attributes such as illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutter, and low resolution. VOT criteria (accuracy and robustness) were used in the evaluation. Both OTB and VOT have been commonly used in the evaluation of object-tracking methods in the literature; thus, testing our approach using them provided a global and up-to-date evaluation.

Accuracy was represented by the average overlap of an estimated target boundary with ground truth data during successful tracking and is scaled between 0 and 1 [19]. Robustness was measured by the average number of failures in 1000 frames during tracking. Failure was determined by the overlap threshold between the estimated target boundary and the ground truth. In the VOT evaluation system, trackers were reinitialized 5 frames after failure, and the first 10 frames were not included in order to prevent artifacts in the evaluation. In addition to accuracy and robustness, speed, represented by the average processing time of a frame in milliseconds, was used to measure run-time performance. Figure 4 shows the accuracy (for different overlap threshold values) of the compared methods for all sequences and specifically illumination variation, scale variation, and occlusion categories, which are challenging scenarios for object tracking. The robustness results of the methods for all sequences are shown in Figure 5, and finally the average run-time performances are shown in Figure 6. Run-time performance of SAMF was not presented because its authors only provided the MATLAB code, while our comparison was performed in C++.

The proposed approach attained better results for all three criteria in the comparison of the methods. In particular, its robustness performance showed the benefit of using alternative filters and the motion-supported drift solution, which were important contributions to our approach. The presented results show that the proposed approach outperformed the other methods, especially for long-term tracking, which is important for surveillance applications. It can be deduced from these results that multimodel representation of the target object provides a better solution than a single model. SAMF, which integrates multiple features, showed the second-best accuracy performance according to Figure 4. This result supported our multimodel representation in a different domain, resulting in multiple feature representation also increasing the success of the method by providing more generic solutions across challenging scenarios.

Run-time performance of the Muster method was significantly lower than the others, as illustrated in Figure 6. Loss of run-time performance is expected because of the multimodel architecture of Muster. Although multimodel representation was applied in our approach, run-time performance was second in the comparison. ASMS is the fastest method through its simple mean-shift approach. These results demonstrated the effectiveness of our multithreaded tracker pool. Another observation obtained from Figure 4 was that use of a smaller overlap threshold increases the accuracy of all the methods, since less tracker failure happens with smaller overlap thresholds.

6. Conclusions

A novel multiobject-tracking method triggered by motion detection is proposed in this paper. Most existing correlation filter-based tracking methods work in real-time for just a single object initialized by the user. The proposed approach is triggered by motion detection without requiring user interaction and is able to track any type of object. It also works in real-time for multiple targets through multithreaded implementation.

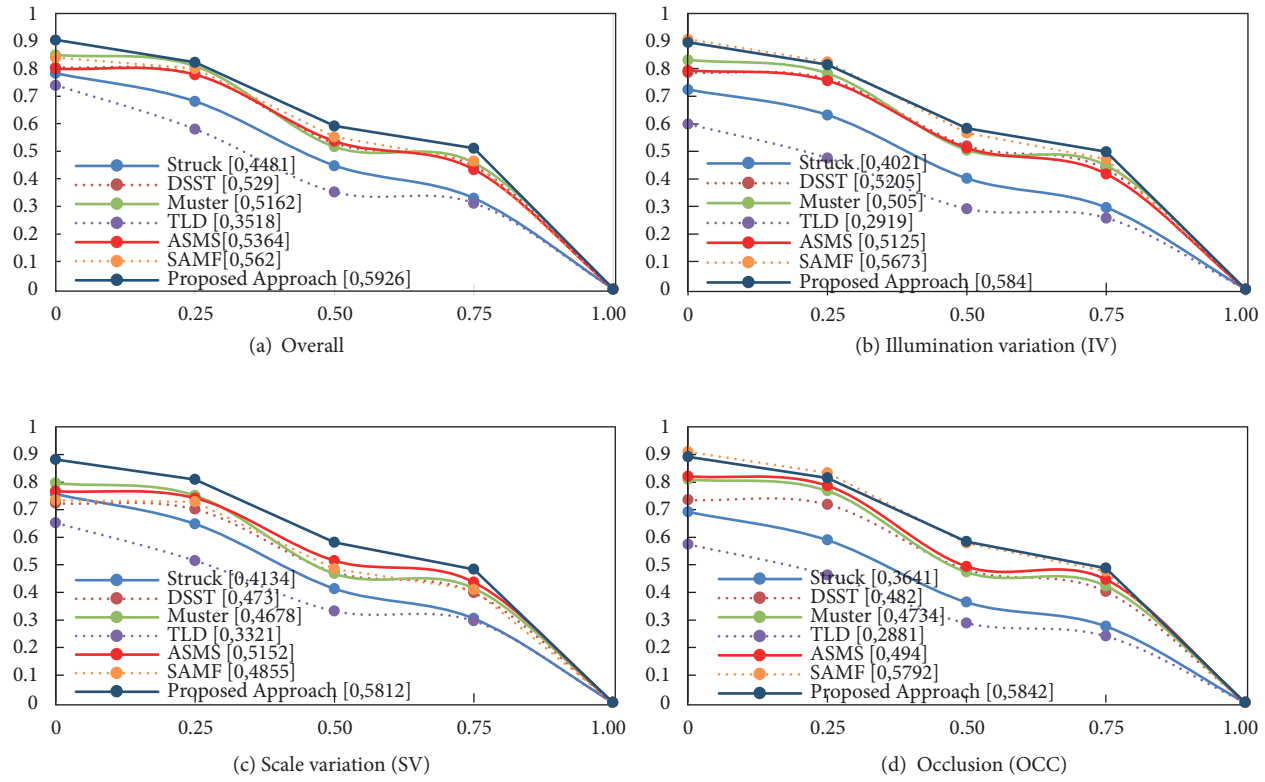

Figure 4. Comparison of the proposed method and classical approaches: a) accuracy comparison for overall sequences, b) illumination variation, c) scale variation, and d) occlusion categories. Vertical axis represents accuracy scaled from 0 to 1. Horizontal axis represents overlap threshold used in experiments. The average accuracies over all sequences for an overlap threshold of 50% are specified in the legend.

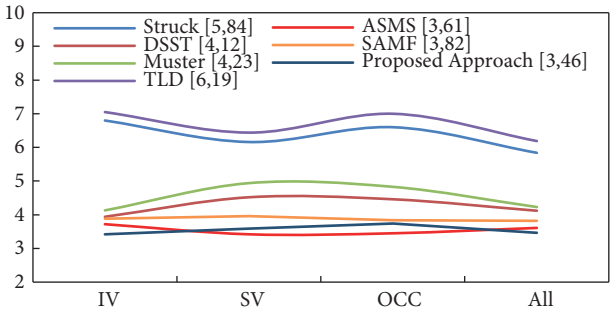

Figure 5. Robustness comparison of all sequences for the illumination variation (IV), scale variation (SV), and occlusion (OCC) categories and finally the overall average. The average robustness over all sequences is specified in the caption.

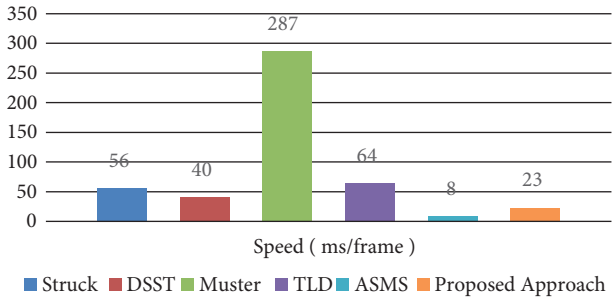

Figure 6. Average run-time performance comparison of all sequences.

Experimental results showed that our approach was able to track multiple objects in real-time from the OTB dataset and outperformed most current state-of-the-art methods.

Object boundary drifting, a common problem in existing tracking methods, was minimized using support for motion blobs in the proposed approach. Furthermore, the proposed method was minimally affected by scale and appearance variation (other important problems in visual object tracking) by applying alternative

correlation filters. Robustness in long-term tracking of the proposed approach provides the opportunity to track the same target when viewed by different cameras. Thus, we plan to use the proposed approach in multicamera object tracking in future work.

References

[1] McCall JC, Trivedi MM. Video-based lane estimation and tracking for driver assistance: survey, system and evaluation. IEEE Trans Intell Transp Syst 2006; 7: 20-37.

[2] Santiago CB, Sousa A, Estriga ML, Reis LP, Lames M. Survey on team tracking techniques applied to sports. In: AIS 2010 International Conference on Autonomous and Intelligent Systems; 21–23 June 2010; Povoa de Varzim, Portugal. pp. 1-6.

[3] Smeulders AW, Chu DM, Cucchiara R, Calderara S, Dehghan A, Shah M. Visual tracking: an experimental survey. IEEE T Pattern Anal 2014; 36: 1442-1468.

[4] Baker S, Matthews I. Lucas-kanade 20 years on: a unifying framework. Int J Comput Vis 2004; 56: 221-255.

[5] Comaniciu D, Ramesh V, Meer P. Real-time tracking of non-rigid objects using mean-shift. In: IEEE 2000 Conference on Computer Vision and Pattern Recognition; 13–15 June 2000; Hilton Head Island, SC, USA. New York, NY, USA: IEEE. pp. 142-149.

[6] Godec M, Roth PM, Bischof H. Hough-based tracking of non-rigid objects. Comput Vis Image Underst 2013; 117: 1245-1256.

[7] Hare S, Golodetz S, Saffari A, Vineet V, Cheng M, Hicks SL, Torr PH. Struck: Structured output tracking with kernels. IEEE T Pattern Anal 2016; 38: 2096-2109.

[8] Kalal Z, Matas J, Mikolajczyk K. Pn learning: Bootstrapping binary classifiers by structural constraints. In: IEEE 2010 Conference on Computer Vision and Pattern Recognition; 13–18 June 2010; San Francisco, CA, USA. New York, NY, USA: IEEE. pp. 49-56.

[9] Bolme DS, Beveridge JR, Draper BA, Lui YM. Visual object tracking using adaptive correlation filters. In: IEEE 2010 Conference on Computer Vision and Pattern Recognition; 13–18 June 2010; San Francisco, CA, USA. New York, NY, USA: IEEE. pp. 2544-2550.

[10] Danelljan M, Hager G, Khan F, Felsberg M. Accurate scale estimation for robust visual tracking. In: BMVA 2014 British Machine Vision Conference; 1–5 September 2014; Nottingham, UK. Durham, UK: BMVA Press.

[11] Li Y, Zhu J. Background segmentation with feedback: A scale adaptive kernel correlation filter tracker with feature integration. In: Springer 2014 European Conference on Computer Vision; 6–12 September 2014; Zurich, Switzerland. Berlin, Germany: Springer. pp. 254-265.

[12] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: IEEE 2005 Conference on Computer Vision and Pattern Recognition; 20–25 June 2005; San Diego, CA, USA. New York, NY, USA: IEEE. pp. 886-893.

[13] Vojir T, Noskova J, Matas J. Robust scale-adaptive mean-shift for tracking. Pattern Recognit Lett 2014; 49: 250-258.

[14] Hong Z, Chen Z, Wang C, Mei X, Prokhorov D, Tao D. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In: IEEE 2015 Conference on Computer Vision and Pattern Recognition; 7–12 June 2015; Boston, MA, USA. New York, NY, USA: IEEE. pp. 749-758.

[15] Danelljan M, Robinson A, Khan F, Felsberg M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In: IEEE 2016 European Conference on Computer Vision; 8–16 October 2016; Amsterdam, the Netherlands. New York, NY, USA: IEEE. pp. 472-488.

[16] Nam H, Han B. Learning multi-domain convolutional neural networks for visual tracking. In: IEEE 2016 Conference on Computer Vision and Pattern Recognition Workshops; 26 June–1 July 2016; Las Vegas, NV, USA. New York, NY, USA: IEEE. pp. 4293-4302.

[17] Danelljan M, Hager G, Khan F, Felsberg M. Learning spatially regularized correlation filters for visual tracking. In: IEEE 2015 International Conference on Computer Vision; 11–18 December 2015; Santiago, Chile. New York, NY, USA: IEEE. pp. 4310-4318.

[18] Hoffman M, Tiefenbacher P, Rigoll G. Background segmentation with feedback: The pixel-based adaptive segmenter. In: IEEE 2012 Computer Society Conference on Computer Vision and Pattern Recognition Workshops; 16–21 June 2012; Rhode Island, USA. New York, NY, USA: IEEE. pp. 38-43.

[19] Kristan M, Matas J, Leonardis A, Felsberg M, Cehovin L, Fernandez G, Vojir T, Hager G, Nebehay G, Pflugfelder R. The visual object tracking vot2015 challenge results. In: IEEE 2015 International Conference on Computer Vision Workshops; 11–18 December 2015; Santiago, Chile. New York, NY, USA: IEEE. pp. 564-586.

[20] Wu Y, Lim J, Yang MH. Object tracking benchmark. IEEE T Pattern Anal 2015; 37: 1834-1848.