A Super-Fast Online Face Tracking System for Video Surveillance

Xiaosong Lan∗, Zhiwei Xiong†, Wei Zhang‡, Shuxiao Li∗, Hongxing Chang∗ and Wenjun Zeng†
∗Institute of Automation, Chinese Academy of Sciences, Beijing, China
†Microsoft Research, Beijing, China
‡University of Waterloo, Waterloo, Canada

Abstract—In this paper, we propose a novel and practical system for robust online face tracking in surveillance videos. The proposed system has two contributions: 1) sustained high performance for long-term tracking even faces come in and out frequently, and 2) extremely low complexity which allows for real-time deployment on various platforms. These advantages are achieved by designing a regular update framework based on a state-of-the-art face detector and a new histogram-assisted KLT (HAKLT) tracker. Experimental results demonstrate a superior and super-fast (>100fps) face tracking system in practice.

I. INTRODUCTION

Security has become one of the top priorities for the world today. With the popularity of surveillance cameras in both public and private spaces, intelligent video analysis is in urgent need for multimedia services on various platforms. Among different functionalities that could be useful, online face tracking is especially attracting since the real-time face trajectory facilitates a wide range of applications.

Face tracking has been extensively studied in the computer vision field. The early algorithms are mainly based on skin color [1], [2], which are sensitive to the environment especially the skin-color alike background. As a classic feature tracker, KLT [3], [4] is widely used for face tracking due to its efficiency and robustness to scale change. However, since KLT only uses the local information, it may cause drifting sometimes. Nowadays, learning-based tracking methods are popular, e.g., MIL [5], TLD [6], SFCT [7], and Struck [8], which can be adapted to face as well. These tracking methods generally require manual inputs for initialization.

The challenge of face tracking in surveillance videos lies in that, in an unconstrained environment, faces may come in and out frequently and it will be impractical to manually initialize the tracker. Therefore, using face detection for initialization is a natural choice. The problem is, one-time initialization may not work for long-term tracking, while frame-by-frame simultaneous face detection and tracking [9], [10] is not computationally efficient for online processing, especially when deployed on devices with limited computing power.

In this paper, we propose a novel and practical system for robust online face tracking in surveillance videos. The proposed system is based on an elaborate framework integrating a state-of-the-art face detector [11] and a new histogram-assisted KLT (HAKLT) tracker. Specifically, we impose a regular update mechanism on the tracker that is supported by the detector. The detector will be triggered once a predefined frame number is reached or any tracklet is determined failed. (Here each tracklet denotes a face trajectory and the tracker may contain several tracklets simultaneously in case of multiple faces.) Meanwhile, the temporarily failed tracklets are buffered to address the reappearing faces. In this way, sustained high performance can be guaranteed for long-term tracking while the overall complexity remains low.

To further reducing the computationally complexity for online processing, we design a simple yet effective HAKLT tracker by combining KLT with the color histogram. The color histogram helps get rid of the drifting problem of KLT while retaining its robustness to scale change. Experimental results demonstrate that, under the regular update framework, HAKLT enables a superior and super-fast (>100fps) face tracking system in practice. It is worth mentioning that, the proposed system can be readily extended to other tracking tasks, such as human body and license plate.

II. SYSTEM FRAMEWORK

Fig. 1 shows the framework of our proposed online face tracking system. There are four main modules in this framework: detection, tracking, buffering, and updating. At the very beginning, an initial face detection triggers the tracking module. The tracking module then runs on each successive frame, and it will be suspended once a predefined frame number $N_1$ is reached or any tracklet is determined failed. In the latter case, the buffering module is triggered to record the information of the temporarily failed tracklets. Then, the detection module is triggered again to provide anchor results. If there are faces detected, the updating module is triggered and the active tracklets are updated based on the detection results as well as the buffered tracklets. Otherwise, we check whether there are still active trackles. A positive response (‘Y’) will return to the tracking module while a negative response (‘N’) will triggers another detection module. The details of detection, buffering, and updating modules are given below, and the tracking module will be elaborated in the next section.

Detection. In this work, we use a state-of-the-art face detector [11], which advances the classic Viola-Jones detector while retaining its high efficiency [12]. Still, the detection module is supposed to be sparsely triggered to minimize the overall complexity. Besides initialization, the detector also provides anchor results for the updating module, which helps activate the buffered tracklets and rectify the drifted tracklets.

Buffering. To address the reappearing faces, we introduce a buffering module in the proposed framework. If any tracklet is determined failed, we move it to a buffer instead of deleting...
it directly. If a later detected face can be associated with a buffered tracklet in $N_2$ frames, the tracklet is activated and updated by the detection result. Otherwise, the buffered tracklet is deleted. In this way, longer face trajectories can be preserved, which is desired in many practical applications.

**Updating.** The regular update mechanism makes the proposed framework distinct from the existing solutions. On the one hand, it relieves the computational burden of frame-by-frame detection. On the other hand, it guarantees sustained high performance for long-term tracking. The updating module works as follows. Given the detection results in the current frame, we first find their counterparts from the active tracklets. Specifically, we use the overlap ratio between a detected face and a tracked face as the indicator of correspondence. The overlap ratio is calculated as

$$R(p_i, q_j) = \frac{|p_i \cap q_j|}{|p_i \cup q_j|}$$

where $p_i$ denotes the region of the $i$-th detected face and $q_j$ denotes the region of the $j$-th tracked face in the current frame. $\cap$ and $\cup$ represent the intersection and union of two regions, respectively. $|\cdot|$ denotes the number of pixels in the region. For each $p_i$, if its largest overlap ratio among the active tracklets, say $R(p_i, q_k)$, is larger than a given threshold $T_1$, $p_i$ is associated with the $k$-th tracklet and $q_k$ is updated by $p_i$. An example is shown in Fig. 2(a), where two active tracklets are both updated by the latest detected faces. If $p_i$ cannot be associated with any active tracklets, we then find its counterpart from the buffered tracklets following the same principle. An example is shown in Fig. 2(b), where a buffered tracklet is activated by the latest detected face. A detected face that cannot be associated with either the active or buffered tracklets is assigned a new tracklet. An example is shown in Fig. 2(c), where the latest detected face (of the girl) is regarded as a newcomer. Note if an active tracklet has not been updated in $N_3$ frames, it is regarded as drifted to the background and will be deleted.

### III. Histogram-assisted KLT

Different from classic trackers such as skin-color based and KLT, recent tracking methods generally learn the appearance of the object on-the-fly and are more robust to the appearance changes caused by illumination and pose variation. However, these methods cannot avoid the degradation due to learning the background or occlusion, so the drifting problem may still be introduced by self-learning. In practice, there exists no long-term tracker that can successfully handle occlusion, background clutter, illumination and pose variation at the same time. Fortunately, in our proposed face tracking system, such a super tracker is not necessary thanks to the regular update framework as elaborated above. Therefore, we design a simple yet effective tracker by combining KLT and the color histogram, which is robust to both drifting and scale change.

The tracking module with HAKLT is shown in Fig. 3. A number of feature points are first extracted from the face region through an uniform sampling. Then the pyramidal KLT algorithm [3], [4] is applied to track these feature points. Next, we filter out the inaccurately tracked points. Specifically, suppose $x$ is a feature point extracted from the previous frame and $x'$ the corresponding tracked one in the current frame, we track $x'$ back to $x''$ in the previous frame, and the distance between $x$ and $x''$ is regarded as the forward-backward error (FBE) [13]. If the FBE of one tracked point is larger than the median FBE of all tracked points, it will be filtered out as inaccurate tracking. The remaining feature points can be denoted as

$$\{(x_i, x'_i) | i = 1, 2, ..., M\}$$

where $M$ is the number of accurately tracked feature pairs.

Based on these $M$ feature pairs, the displacement and scale change of the face region can be calculated as

$$\delta = \text{median}(x'_i - x_i), \quad i \in \{1, 2, ..., M\}$$

$$\alpha = \text{median}(\frac{||x'_i - x'_j||}{||x_i - x_j||}), \quad i, j \in \{1, 2, ..., M\}$$

Then the location and size of the new face region can be obtained through the previous one.

After we obtain the new face region, two criteria are used to determine whether the tracklet fails. One is the median FBE of all accurately tracked points. If the value is larger than a given threshold $T_2$, the tracklet will be regarded as failed. The other is the color histogram of the whole face region, as the feature points only represent partial information of the face. If the color histogram of the new face region is significantly different from the one updated by the latest detection result, there is a large probability that the tracklet has drifted to the background and should be regarded as failed. The distance between two histograms $H_1$ and $H_2$ is calculated as

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{H_1H_2L^2}} \sum_{i=1}^{L} \sqrt{H_1(i) \cdot H_2(i)}}$$

where $L$ is the number of bins, $\bar{H}_1$ and $\bar{H}_2$ are the mean value of $H_1$ and $H_2$, respectively. Correspondingly, there is a given threshold $T_3$ for evaluating the histogram similarity.
Fig. 2: Examples of updating tracklets. (a) Two active tracklets are both updated by the latest detected faces. (b) A buffered tracklet is activated by the latest detected face. (c) The latest detected face (of the girl) is assigned a new tracklet. Red rectangles denote the latest detected faces, black ones denote the buffered tracklets, and other colors denote different active tracklets.

Fig. 3: Tracking with HAKLT.

Table I: Parameters empirically set in the experiments.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N_1$</td>
<td>50</td>
<td>$T_1$</td>
<td>0.3</td>
</tr>
<tr>
<td>$N_2$</td>
<td>55</td>
<td>$T_2$</td>
<td>10</td>
</tr>
<tr>
<td>$N_3$</td>
<td>100</td>
<td>$T_3$</td>
<td>0.5</td>
</tr>
</tbody>
</table>

IV. Experiments

To evaluate the performance and efficiency of the proposed system, we conduct experiments on a variety of surveillance videos. Here we present the results on four test sequences: Seq1 (“Girl”) and Seq2 (“OccludedFace2”) are from the public Babenko tracking dataset [14]. Seq3 (“P1E_S1_C1”) is a public indoor surveillance video from [15], and Seq4 is an indoor surveillance video captured by our own webcam (typical frames are shown in Fig. 2 where faces come in and out frequently). We obtain the anchor face regions through frame-by-frame detection [11] (the frames without detected faces are not counted). Two common metrics, the success plot and the precision plot, are used to measure the bounding box overlap and the center location error under different thresholds [16]. The parameters used in the experiments are given in Table I. Here we choose a uniform setting that should be well generalized to other sequences. Note that, to allow for finer trade-off between complexity and accuracy, the optimal parameters may vary according to the content of surveillance videos.

For the first experiment, we integrate two state-of-the-art trackers, SFCT [7] and Struck [8] under the proposed regular update framework and make a comparison with one-time initialization. The scale number of SFCT is set to 5 and the feature number of Struck is set to 100. The area under curve score of the success plot (S-Score) and the score at an error threshold of 20 of the precision plot (P-Score) are shown in Table II. It can been seen that the proposed framework significantly outperforms one-time initialization, which demonstrates the effectiveness of the regular update mechanism for long-term tracking.

For the second experiment, we compare SFCT, Struck, and HAKLT under the proposed regular update framework. Since different trackers may fail at different instants, we update all trackers at the same time for a fair comparison. The results are shown in Fig. 4 and Fig. 5. For all test sequences, our designed HAKLT consistently outperforms SFCT and Struck. The main reason is that HAKLT is more robust to drifting and scale change and not sensitive to update. Therefore, it is validated that HAKLT best fits the proposed framework for robust online face tracking.

The average time for processing one frame with different trackers is shown in Table III, which is calculated on an Intel Core i5 3.30GHz CPU. As can be seen, the processing speed with HAKLT exceeds 100fps at a 640×480 resolution under the proposed framework. We believe this is by far the highest speed reported in the literature for robust online face tracking.

Table III: Statistics of one frame processing time (ms).

<table>
<thead>
<tr>
<th>Tracker</th>
<th>SFCT</th>
<th>Struck</th>
<th>HAKLT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seq1 (320×240)</td>
<td>14.19</td>
<td>51.42</td>
<td>5.09</td>
</tr>
<tr>
<td>Seq2 (320×240)</td>
<td>14.45</td>
<td>51.43</td>
<td>5.53</td>
</tr>
<tr>
<td>Seq3 (640×480)</td>
<td>19.73</td>
<td>60.56</td>
<td>9.82</td>
</tr>
<tr>
<td>Seq4 (640×480)</td>
<td>17.50</td>
<td>56.55</td>
<td>9.61</td>
</tr>
</tbody>
</table>
TABLE II: Performance comparison of one-time initialization and the proposed regular update framework.

<table>
<thead>
<tr>
<th>Framework</th>
<th>Seq1 (SFCT)</th>
<th>Seq1 (Struck)</th>
<th>Seq2 (SFCT)</th>
<th>Seq2 (Struck)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>S-Score</td>
<td>P-Score</td>
<td>S-Score</td>
<td>P-Score</td>
</tr>
<tr>
<td>One-time initialization</td>
<td>0.252</td>
<td>0.276</td>
<td>0.641</td>
<td>0.880</td>
</tr>
<tr>
<td>Regular update (proposed)</td>
<td>0.485</td>
<td>0.640</td>
<td>0.709</td>
<td>0.934</td>
</tr>
</tbody>
</table>

Fig. 4: Performance comparison (success plots) of SFCT, Struck, and HAKLT under the proposed framework.

Fig. 5: Performance comparison (precision plots) of SFCT, Struck, and HAKLT under the proposed framework.

V. CONCLUSIONS

In this paper, we have presented a novel and practical system for robust online face tracking in surveillance videos. Our system adopts a regular update framework, which guarantees sustained high performance for long-term tracking even faces come in and out frequently. To maintain extremely low computational complexity, we further design a simple yet effective HAKLT tracker that is robust to drifting and scale change. Experimental results validate the superior and superfast performance of our system in practice, which is highly desired for intelligent video analysis on various platforms.

REFERENCES