Hierarchical Ensemble of Background Models for PTZ-Based Video Surveillance

Ning Liu, Hefeng Wu, and Liang Lin

***Abstract*—In this paper, we study a novel hierarchical background model for intelligent video surveillance with the pan-tilt-zoom (PTZ) camera, and give rise to an integrated system consisting of three key components: background modeling, observed frame registration, and object tracking. First, we build the hierarchical background model by separating the full range of continuous focal lengths of a PTZ camera into several discrete levels and then partitioning the wide scene at each level into many partial fixed scenes. In this way, the wide scenes captured by a PTZ camera through rotation and zoom are represented by a hierarchical collection of partial fixed scenes. A new robust feature is presented for background modeling of each partial scene. Second, we locate the partial scenes corresponding to the observed frame in the hierarchical background model. Frame registration is then achieved by feature descriptor matching via fast approximate nearest neighbor search. Afterwards, foreground objects can be detected using background subtraction. Last, we configure the hierarchical background model into a framework to facilitate existing object tracking algorithms under the PTZ camera. Foreground extraction is used to assist tracking an object of interest. The tracking outputs are fed back to the PTZ controller for adjusting the camera properly so as to maintain the tracked object in the image plane. We apply our system on several challenging scenarios and achieve promising results.**

***Index Terms*—Hierarchical background modeling, object localization, PTZ camera surveillance, video tracking.**

Manuscript received June 23, 2013; revised January 13, 2014, April 1, 2014, and April 17, 2014; accepted April 22, 2014. This work was supported in part by the National Natural Science Foundation of China under Grant 61100080, Grant 61320106008, Grant 61173082, and Grant 61370186, in part by the Program of Guangzhou Zhujiang Star of Science and Technology under Grant 2013J2200067, in part by the National Key Basic Research and Development Program of China under Grant (973) 2013CB329505, in part by NSFC-Guangdong Joint Fund under Grant U1135003, in part by Guangdong Natural Science Foundation under Grant S2013050014548, in part by Special Project on Integration of Industry, in part by Education and Research of Guangdong Province under Grant 2012B091100148, and in part by Shenzhen special funds for the development of strategic emerging industries under Grant jcyj2012061515751107. This paper was recommended by Associate Editor A. Roy-Chowdhury.

N. Liu is with the School of Software, Sun Yat-sen University, Guangzhou 510006, China (e-mail: liuning2@mail.sysu.edu.cn).

H. Wu is with the National Engineering Research Center of Digital Life, State-Province Joint Laboratory of Digital Home Interactive Applications, School of Information and Science Technology, Sun Yat-sen University, Guangzhou 510006, China, and also with Guangdong Zhicheng Champion Group Company, Ltd, Dongguan 523000, China (e-mail: wuhefeng@gmail.com).

L. Lin is with the School of Advanced Computing, Sun Yat-Sen University, Guangzhou 510006, China, and also with SYSU-CMU Shunde International Joint Research Institute, Shunde 528300, China (e-mail: linliang@ieee.org).

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/TCYB.2014.2320493

I. Introduction

VISUAL surveillance is an active research area in computer vision. It has a wide range of applications such as trajectory analysis [1], traffic monitoring [2], crowd control [3], and many other related fields. Recently, pan-tilt-zoom (PTZ) camera-based surveillance systems receive increasing research interest [4]–[7] as their flexibility on rotation and zoom offers a broad range of views and fine scene details. However, most existing work proposed for fixed cameras can not be directly applied to PTZ camera-based systems due to the presence of varying focal lengths and scene changes.

Background subtraction is a fundamental task in surveillance applications. A great number of background subtraction approaches with fixed cameras have been presented in the past several decades [8]–[16], especially for handing complex scenes with dynamic background. However, there is much less research work for PTZ camera-based background subtraction. This can be attributed to the following difficulties. First of all, a fixed camera captures a scene of the same place, and the variations in every pixel location can be modeled independently and exhaustively in the fixed field of view. A PTZ camera has a much broader field of view than a fixed camera, but it cannot capture the broad scene at one shot. It must rotate to capture different parts of the broad scene, and a pixel location in the image plane can contain information that comes from diverse locations of the scene. Furthermore, when a PTZ camera zooms in/out, it greatly affects the local feature descriptors that are commonly used for background modeling.

Certain progress in PTZ-based background modeling has been made [17]–[20]. However, there remain challenging issues in the literature presented so far, which we will discuss in detail in the related work. These existing problems motivate us to put forward a hierarchical background modeling solution for PTZ cameras in this paper. It can take full advantage of the effective background modeling algorithms developed for fixed cameras.

The rest of the paper is organized as follows. We first review the related work and present an overview of the proposed surveillance framework. Afterwards, Section II describes the hierarchical background model for PTZ cameras, and Section III discusses the observed frame registration approach. The object tracking scheme is presented in Section IV, and experiments are given in Section V. We conclude the paper in Section VI.

A. Related Work

Most of the background modeling approaches are discussed in the context of stationary cameras. Mixture of Gaussians (MoG) [8]–[10] is widely studied for modeling the real-world scenes that contain dynamic complex backgrounds, e.g., waving trees, rippling water, moving escalators, etc. Meanwhile, nonparametric methods [11], [12] are also proposed to build the background model. The feature used is another important factor in background modeling. Recently, local textures have been considered a good alternative for modeling a pixel or a small block. Heikkilä and Pietikäinen [14] use the local binary pattern (LBP) descriptor as the feature, which is more robust to illumination changes. A novel scale invariant local ternary pattern (SILTP) feature is also presented by Liao *et al*. [15] for background modeling.

PTZ-Based Background Model: In recent years, PTZ-based background modeling gains more and more research attractions. The methods presented in the literature so far can be divided into two main categories: frame-to-frame and frame-to-global. Frame-to-frame methods focus on reuse of pixel information from overlapping regions of the observed frame and its previous frames. Kang *et al*. [17] present an adaptive background generation algorithm. They use geometric transform to align consecutive frames when the PTZ camera rotates. Wu *et al*. [18] find frame-to-frame correspondences by using two extended Kalman filters to simultaneously estimate zoom and pan-tilt parameters. The Laplacian pyramid is employed to estimate the motion parameters between two frames in a coarse-to-fine manner and to align the overlapping image regions in [21]. Frame-to-global methods focus on building and maintaining a background image of the whole monitored scene. Bevilacqua *et al*. [19] propose to build a global background mosaic in real-time for detecting foreground objects under PTZ cameras. The authors stitch a panorama for each different scale of a PTZ camera, using a number of high-resolution frames at each scale in [20]. Xue *et al*. [22] propose to build a panoramic Gaussian mixture model (PGMM) covering the PTZ camera's field of view. A multilayered correspondence ensemble is used to help register newly captured frames to the panoramic model.

B. Relations to Our Work

In this subsection, we discuss the relations and differences between the proposed PTZ-based background model and previous work.

The frame-to-frame methods [17], [18] utilize neighboring frames for frame registration. This can obtain aligned images with less distortion, but the registration may be degraded by moving objects. Moreover, foreground detection in certain frame regions is intractable due to the lack of global scene information. In addition, they cannot handle the focus changes of PTZ cameras well. With a hierarchical ensemble of partial scenes to provide global information, the proposed method can avoid the disadvantages of the frame-to-frame methods, and utilize their advantages by registering the observed frame to neighboring partial scenes.

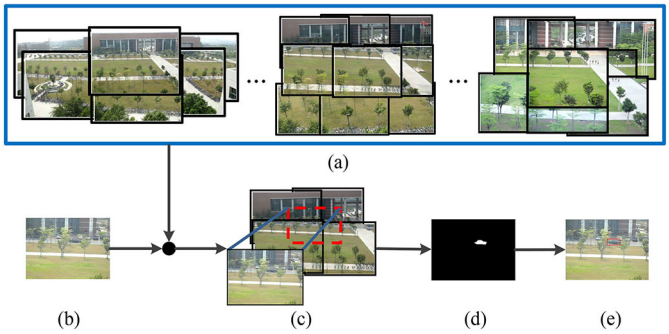


Fig. 1. Visualization of the proposed method. (a) Hierarchical background model. (b) New frame. (c) Registration. (d) Detection. (e) Tracking.

Frame-to-global methods [19], [20] use a stitched panorama to provide global spatial information for foreground detection. However, these methods may provide a panoramic background image containing moving objects. Some stitching problems, e.g., heavy image distortion and serious artifacts, may also be introduced. When building the panoramic image, the frame-to-global methods commonly follow the assumption that there is no significant motion parallax, i.e., depth variations in the scene are not apparent from the motion of the camera. This assumption may not hold in some scenarios. For example, in the situations where the camera rotates in a large vertical range or the objects in the scene are not far way from the camera (e.g., when the camera is mounted on roadsides, street corners, inside or among buildings), depth variations will be quite obvious from the motion of the camera. The proposed hierarchical background model avoids these problems by modeling an ensemble of partial scenes [Fig. 1(a)] rather than building the panoramic scene image. In addition, it is nontrivial to register a frame captured at a fine scale to the panoramic model. The registration process may fail due to few matched features. A multilayered feature correspondence propagation method is presented in [22] to generate enough feature correspondences for better registration at the cost of high computation. However, the background subtraction results may heavily be affected by image distortion. On the contrary, the proposed method register the observed frame to the partial background models at a similar scale, which can greatly alleviate these problems and also achieve computational efficiency. Building a separate panoramic background model for each scale may also provide a way for alleviating the image distortion problem in registration. But, as we discussed above, the panoramic model is more suitable for certain scenarios. Furthermore, the model updating process is more difficult and expensive for a panoramic image. Instead, the proposed method can update the background model of each partial scene independently in a more feasible way.

Guillot *et al*. [23] propose a background subtraction algorithm suitable for a PTZ camera performing a guard tour. Although the problem addressed by them is different from this paper, their method and ours share similarities in using a predefined set of positions. Guillot *et al*. maintain a background image for each position and obtain the foreground from the image regions covered by nonmatching keypoints. Whilst the

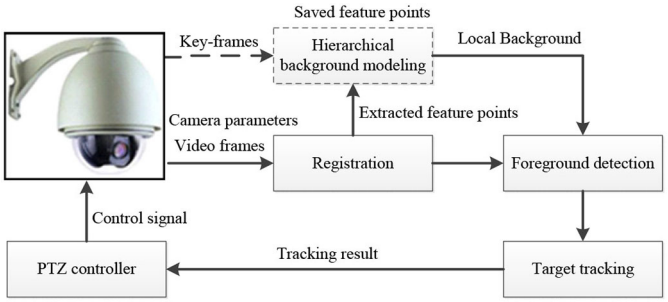


Fig. 2. Flowchart of the proposed framework. The hierarchical background model (shown in dashed lines) is built in advance. An observed frame newly captured by the PTZ camera is registered to the background model, and foreground detection is done locally. The detection result can be used to assist tracking the target. In addition, the tracking result will serve as feedback to the PTZ controller for camera adjustment.

proposed method builds a hierarchical background model to cover the whole scene monitored by the PTZ camera, it explicitly addresses the scale changes of the PTZ camera and the relations of partial scenes, which are not addressed in [23]. In addition, we also utilize more robust methods for background modeling and foreground detection.

The proposed background model is configured into a framework coupled with existing tracking methods for robust object tracking via the PTZ camera. Most of the state-of-the-art object tracking algorithms [24]–[29] rely on the assumptions that the objects being tracked are not undergoing fast changes of scales. They usually track the object with fixed-size bounding boxes. However, it may not hold with respect to object tracking under the PTZ camera. The camera adjustments (CAs) may cause the target to change rapidly in location and scale, which easily induces a tracking algorithm to fail. With some recent tracking-by-detection methods [25], [28] or object recognition methods [30], [31], it seems that one can detect and track the object without the need of a background model. However, a detector only fits for a specific kind of object (e.g., human), it cannot find other kinds of foreground objects. These methods could not yet handle the scale changes of objects well, and they have to find the target through exhaustive detection. The proposed background model can provide a helpful complement for existing tracking methods in PTZ-based surveillance.

C. Overview

Fig. 1 visualizes the framework of the proposed surveillance system, while a flowchart of the framework is provided by Fig. 2. The system framework is mainly composed of three parts: hierarchical background modeling, observed frame registration, and object tracking.

1) Hierarchical Background Modeling: As shown in Fig. 1(a), we divide the scale range of the PTZ camera into several discrete layers. We further partition the wide scene at each layer, which can be captured by a PTZ camera when it rotates, into a few partial scenes with some overlaps. A partial scene can be captured by the PTZ camera at a certain view. All the partial scenes together can cover the whole field of view of the PTZ camera. A frame captured at a partial scene is termed as a key-frame. Each partial scene is then directly applied with background modeling methods developed for fixed cameras.

TABLE I
IMPORTANT NOTATIONS USED IN THE PAPER

Notation	Definition
α	the pan angle of the camera
β	the tilt angle of the camera
f	the focus length of the camera
I^t	the frame captured at time t
W, H	the with and height of a frame
(x, y)	a pixel location in the image plane
N	the number of hierarchical scene layers
$\mathcal{K}$	the camera parameters (α, β, f) of a key-frame
$\mathcal{H}$	the homography matrix
p, q	local pattern descriptors
$d(p, q)$	the Hamming distance between binary patterns
$f_p(q)$	the kernel function of a patten with the mean p
w	weighting coefficients
$\Phi(q)$	the pattern probability density function
$\mathcal{P}(q)$	the probability of the pattern q being background
u	feature points
$\mathcal{M}, \mathcal{N}$	the row and column of image subregion division
$\Re$	the bounding rectangle of the tracked object
x_l, x_r	the left and right sides of the object rectangle
y_u, y_b	the upper and bottom sides of the object rectangle
(v_x, v_y)	the motion velocity of the object
r_z	the ratio of the size of the object to that of the frame

2) Frame Registration for Foreground Detection: Given an observed frame [Fig. 1(b)] newly captured by the PTZ camera, we need to register it to the online maintained background model, and find its corresponding partial scenes [Fig. 1(c)]. The camera's parameters are used to limit the search in a small set, and accurate alignment is achieved by feature descriptor matching. After the new frame is registered, the corresponding local background models are used to detect foreground objects [Fig. 1(d)].

3) Object Tracking With the Proposed Background Model: We combine the hierarchical background model with existing tracking algorithms to track the specified target detected from the foreground [Fig. 1(e)]. As depicted in Fig. 2, the tracking results that reflect the positions and scales of the target will be used for controlling the PTZ camera. The controller sends signal to maintain appropriate position and scale of the target in the camera screen.

The key contributions of this paper are three-fold. 1) We present a hierarchical background model for PTZ camera-based surveillance, and this model integrates the advantages of existing background modeling algorithms. 2) A novel local texture descriptor is proposed for background modeling, which encodes spatio-temporal information and is more robust for real-world challenges. 3) The proposed hierarchical background model is configured into a framework to couple with existing tracking methods for object tracking. Moreover, we conduct several scenarios to validate the proposed method. We compare and analyze the proposed PTZ-based background model in both qualitative and quantitative ways, and verify the effectiveness of the proposed tracking framework in challenging scenes.

In Table I, we list the main notations used in the paper.

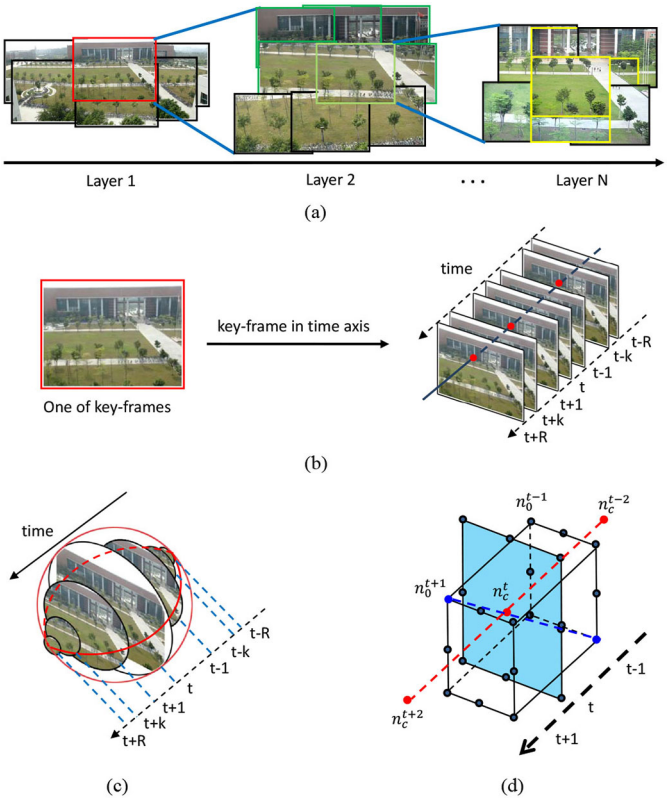


Fig. 3. Illustration of hierarchical scene representation and the extraction of SCS-SILTP descriptor. (a) Hierarchial scene partitioning and correspondences. (b) Background modeling at a partial fixed scene. (c) Local spatio-temporal information of a pixel location. (d) Example structure for the extraction of SCS-SILTP texture descriptor.

II. PTZ Background Model

In this section, we first discuss the process of building hierarchical scene model, and then present a novel texture descriptor for modeling the background in a partial scene.

A. Hierarchical Ensemble Scene Representation

Assume that the scale range of the PTZ camera is discretized into N layers. The coarsest layer (i.e., with a small focal length) is Layer 1, and the finest layer is Layer N, as shown in Fig. 3(a). We model the whole scene at each layer, respectively, and build correspondences between consecutive layers. As mentioned above, the whole monitoring scene of the PTZ camera at a specific layer can be represented as an ensemble of partial scenes. The key-frame of a partial scene can be parameterized by a vector $\mathcal{K} = (\alpha, \beta, f)$, where α is the pan angle, β is the tilt angle, and f is the focal length. The corresponding partial scene is denoted as $Scene(\mathcal{K})$. Assume that the number of partial scenes at Layer $n \in \{1...N\}$ is M_n. We denote the whole scene of Layer n by $Scene_n = \{Scene(\mathcal{K}_{n,m})\}_{m=1}^{M_n}$.

Different schemes can be designed to find the key-frame ensemble of a specific layer. For example, we can have the PTZ camera randomly capture a set of overlapping frames with different pan and tilt angles, and design a greedy algorithm to find the proper set of key-frames from them. However, deciding the key-frame ensemble manually may yield more benefits. For a narrow scene like street corners in between buildings, we can use a sparse set of partial scenes to cover it. In a wide scene, we can use a regular way to construct the ensemble of partial scenes at each layer. More specifically, let $[0, \Phi_{pan}]$ and $[0, \Phi_{tilt}]$, respectively, be the the ranges of pan and tilt angles of the PTZ camera. For Layer n, we divide the pan and tilt ranges, respectively, into $\bar{P}_n$ and $\bar{Q}_n$ discrete bins so that neighboring partial scenes overlap and the union cover the whole scene at this layer.

We build a background model for each partial scene when the hierarchical scene representation is completed. As shown in Fig. 3(b), a time series of frames are captured for initializing the background model of the partial scene denoted by the key-frame marked in red in Fig. 3(a). In Section II-B, we will describe this process in detail.

In addition, we will build key-frame correspondences between consecutive layers, which is useful for object tracking when using zoom in/out with the PTZ camera. The correspondences between two key-frames can be estimated through their shared image regions if the two partial scenes have overlaps. Specifically, given two corresponding pixel locations u_i and u_j in two partial scenes $\mathcal{K}_i$ and $\mathcal{K}_j$, their relationship can be described by a homography transform, defined as

$$u_i = \mathcal{H}_{ij} u_j \tag{1}$$

where $\mathcal{H}_{ij}$ is a 3×3 matrix. The estimation of homography correspondences will be introduced in Section III.

After the corresponding matrix between two key-frames in the consecutive layers are found, we project the coordinates of their four image corners into each other's image plane. The projected coordinates are what we stored for later use. As exhibited in Fig. 3(a), we will build the correspondences between the key-frames that share part of scene from neighboring layers (e.g., the key-frame marked in red in Layer 1 and the key-frames marked in green in Layer 2, the key-frame marked in light green and the key-frames marked in yellow in the next layer). Algorithm 1 describes the process of building key-frame correspondences.

Algorithm 1 Building Key-Frame Correspondences Between Consecutive Layers

Require: Hierarchical ensemble of key-frames $\{\mathcal{K}_{n,m}\}$, $n \in \{1...N\}$, $m \in \{1...M_n\}$.
Output: Correspondences of every pair of key-frames that share scenes at consecutive layers.

```
begin
    Extract feature points for each key-frame;
    for each key-frame K_{n,m}, n ∈ {1, ..., N − 1}
        for each neighboring key-frame K_{n+1,i}, which has adjacent pan
        and tilt angles, at Layer (n + 1)
            if no sufficient corresponding locations are found then
                Continue to process next neighboring key-frame;
            end
            Estimate the homography matrix H between K_{n,m} and K_{n+1,i}
            from the set of corresponding locations;
            Store the homography matrix of the two key-frames;
            Project their image corners into each other's image plane and
            store them;
        end
    end
end
```

B. Modeling Pixel Process of Partial Scene

Each partial scene can be treated as being captured by a fixed camera. Motivated by SILTP [15] and [32], in this paper, we propose a novel texture descriptor termed spherical center-symmetric scale invariant local ternary pattern (SCS-SILTP) for background modeling. This descriptor has two important advantages compared to original features: 1) integration of spatio-temporal statistics and 2) better robustness to noises and shadows based on the scale invariance property. We describe image pixels with this pattern descriptor, and model the background scene by estimating and updating the kernel density of each pixel.

1) SCS-SILTP Descriptor: SILTP [15] is calculated by comparing the values of a center pixel and its neighboring pixels, which improves from LTP [33] with a scale factor to achieve gray-scale invariance. Given a pixel location (x_c, y_c), SILTP encodes it as

$$\mathtt{SILTP}^{\tau}_{\bar{N},R}(x_c, y_c) = \bigoplus_{k=0}^{\bar{N}-1} s_\tau(I_c, I_k) \tag{2}$$

where I_c is value of the center pixel (x_c, y_c), and $\{I_k\}_{k=0}^{\bar{N}-1}$ is the values of its $\bar{N}$ neighboring pixels which are equally spaced on a cycle of radius R, $\bigoplus$ denotes concatenation operator of binary strings, τ is scale factor indicating the comparing range, and s_τ is defined as

$$s_\tau(I_c, I_k) = \begin{cases} 01, & \text{if } I_k > (1+\tau)I_c, \\ 10, & \text{if } I_k < (1-\tau)I_c, \\ 00, & \text{otherwise.} \end{cases} \tag{3}$$

We propose to take advantage of temporal information, and encode the spatio-temporal information in a center-symmetric way. It can be imagined that the circle with radius R surrounding the center pixel in the 2-D image plane is extended to a sphere with radius R in the 3-D spatio-time domain, as illustrated in Fig. 3(c). An example of discretization is shown in Fig. 3(d). The SCS-SILTP operator is formulated as

$$\begin{aligned} \mathtt{SCS\text{-}SILTP}^{\tau}_{R}(x_c, y_c) = & \bigoplus_{k=0}^{(\frac{\bar{N}_0}{2})-1} s_\tau\left(I_k^{t+0}, I_{CS(k)}^{t-0}\right) \\ & \bigoplus_{r=1}^{R} \bigoplus_{k=0}^{\bar{N}_r} s_\tau\left(I_k^{t+r}, I_{CS(k)}^{t-r}\right) \end{aligned} \tag{4}$$

where I^t denotes the frame captured at time instant t, I_k^{t+r} and $I_{CS(k)}^{t-r}$ are the center-symmetric pixel locations lying on the spherical surface, and $\bar{N}_r$ is the number of pixel locations used in the images I^{t+r} and I^{t-r}. The locations on the frame I^t are addressed separately in (4), for their center-symmetric locations are on the same frame.

2) Background Modeling Using Kernel Density Estimation of Local Patterns: We utilize the pattern kernel density estimation method from [15] to maintain an approximate distribution for each pixel location in the scene.

As in [15], we define a distance function $d(p, q)$ between two local patterns p and q as the number of different bits between them, which can be fast computed via XOR operation. Then we obtain the local pattern kernel $f_p(q) = g(d(p, q))$, where g is a Gaussian-like weighting function.

Given a pixel location (x, y), we maintain K most frequently occurred local patterns $\{p_i\}_{i=1}^{K}$, and the pattern probability density function can be approximated smoothly by

$$\Phi(q) = \sum_{i=1}^{K} w_i f_{p_i}(q) \tag{5}$$

where w_i are weighting coefficients, and $\sum w_i = 1$. We sort the K patterns with the corresponding weights in descending order.

Given a new pattern p_t extracted at the pixel location (x, y), we can update the estimated distribution $\Phi(q)$ accordingly. The pattern p_t is matched to the K sorted patterns $\{p_i\}$ in turn, and a match is found if $f_{p_i}(p_t) > T_m$, where T_m is a constant threshold controlling the matching. The weight of each pattern p_i is updated as

$$w_i = (1-\rho)w_i + \rho\Gamma(p_i, p_t) \tag{6}$$

where ρ is a learning rate, and $\Gamma(p_i, p_t)$ is an indicator variable that takes 1 if p_i and p_t are matched and 0 otherwise. If none of the K patterns matches the current pattern p_t, the one with the lowest weight is replaced with p_t, and a low initial weight.

3) Foreground Detection: Given a pattern p_t newly extracted at the location (x, y), we can determine whether it is background or not. The first Q patterns from the K maintained patterns are used to make the decision, and Q is obtained by

$$Q = \underset{k}{\operatorname{argmin}}\left(\sum_{i=1}^{k} w_i > T_b\right) \tag{7}$$

where $T_b \in [0, 1]$ is a threshold indicating how many data should be considered as background, so that some newly added patterns (likely to be foreground) are excluded. Afterwards, the probability of pattern p_t being background is estimated as

$$\mathcal{P}(p_t) = \frac{1}{\sum_{i=1}^{Q} w_i} \sum_{i=1}^{Q} w_i f_{p_i}(p_t). \tag{8}$$

The pixel location (x, y) is marked as foreground if $\mathcal{P}(p_t)$ is less than T_{bg}, a predefined constant parameter.

4) Analysis and Discussion: When the PTZ camera is not activated for foreground detection or object tracking, it can scan the monitored area and stop at each position of partial scene to update the partial background models.

Several frames of the same position need to be captured for foreground detection when the SCS-SILTP feature is used in background modeling, since it contains spatio-temporal information. Therefore, it is not suitable for continuous adjustments of the PTZ camera. However, we can make intermittent CAs in video surveillance, especially for tracking an object of interest. In this way, we can reduce the computational cost for frame registration, since not every frame needs being registration. In addition, the proposed hierarchical background model is suitable for the PTZ camera performing continuous adjustments when using features that do not contain temporal information (e.g., color and SILTP).

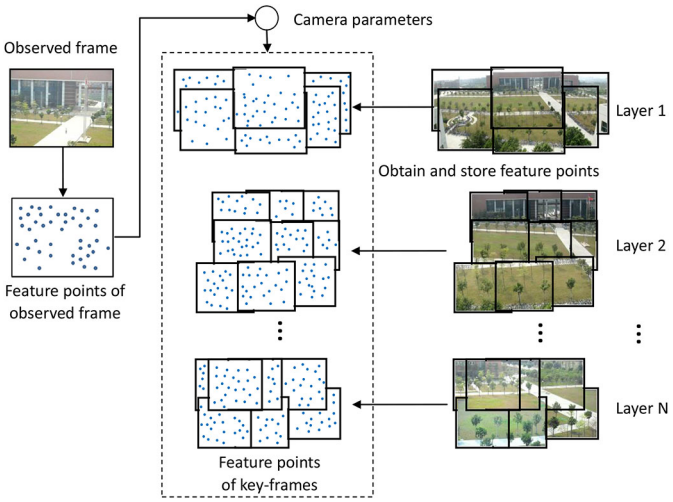


Fig. 4. Illustration of the frame registration process: feature point correspondences are used for homography estimation between the observed frame and the key-frames picked out by camera parameters.

In foreground detection, the features of the observed frame are extracted after registration. The pixel values after registration may be interpolated, but they are quite similar with the neighboring ones before registration. The pattern features, e.g., SILTP and SCS-SILTP, just make use of the comparisons of pixel values, which makes them robust. In addition, the noise-resisted parameter τ also contributes to the robustness of the pattern features. In experiments, we found that the effect of the registration error on pattern representation and further on background subtraction is tolerable.

III. Frame Registration

Fig. 4 visualizes the process of frame registration. As you can see, feature point matching plays an important role in the registration process. In initialization, feature points from key-frames are extracted and stored in advance. However, problems arise if a "clean" key-frame, in which no moving objects exist, cannot be captured. Moving objects can badly corrupt the registration process because they provide incorrect information. In such situations, we follow the technique used in [34] to generate a clean key-frame.

In the monitoring process, when a new frame is captured by the PTZ camera, we need to register it into the hierarchical ensemble of partial scenes, and detect foreground objects using corresponding local background models. First, we retrieve the camera parameters of the current frame I^t from the PTZ controller, i.e., the pan, tilt angles, and the focal length (α^t, β^t, f^t). Since the parameters of any key-frame are designed offline and thus known in advance, we can pick out the neighboring key-frames with the parameters surrounding (α^t, β^t, f^t). We define the set of neighboring key-frames by

$$Set(I^t) = \{\ (\alpha, \beta, f)\ |\ f = rnd\left(f^t\right) \|\alpha - \alpha^t\| < T_{pan}, \|\beta - \beta^t\| < T_{tilt}\ \}. \quad (9)$$

The function $rnd(\cdot)$ rounds the focal length to the nearest layer. In order to make the registration more accurate, we may constrain the PTZ controller to just use focal lengths around those of the discretized layers. T_{pan} and T_{tilt} are thresholds for the absolute differences of pan and tilt angles. By choosing appropriate values, about four related key-frames will be picked out for further processing.

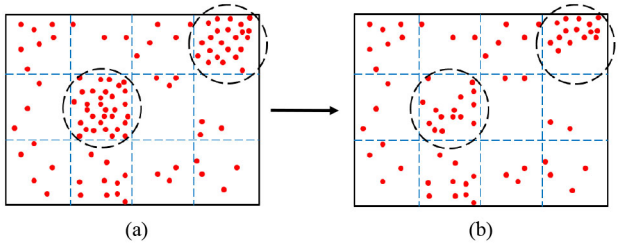


Fig. 5. Feature point extraction by constraining subregion thresholds. (a) A lot of feature points come from a small portion of the image. (b) Feature points come nearly uniformly from the whole image with subregion constraints.

Afterwards we will find a set of location correspondences for estimating the homography transform between the observed frame and the selected key-frames, as shown in Fig. 4. Generally speaking, the feature points used should be robust enough for matching two images captured by the PTZ camera at different views. We can take into consideration some good feature point descriptors proposed in recent years, such as SIFT [35], SURF [36], and ORB [37]. The feature descriptors of each key-frame are extracted and stored when initialization, so we only need to extract the descriptors of the new frame and match them against the stored descriptors. With the feature points of key-frames extracted offline, we can use some fast approximate nearest neighbor algorithms [38], [39] to speed up the online search. After a set of location correspondences is found by feature matching, we further estimate the homography matrix $\mathcal{H}_{ij}$ between two frames $\mathcal{K}_i$ and $\mathcal{K}_j$, as previously described by (1). The RANSAC algorithm [40] is employed to find a good solution.

There may exist another problem when extracting feature points, that is, a lot of feature points may come from a small portion of the image, as illustrated in Fig. 5(a). It is not suitable for aligning the observed frame with different key-frames. To address this problem, we propose a novel method to extract feature points by applying subregion constraints.

We divide the image into $\mathcal{M}$ rows and $\mathcal{N}$ columns, as shown in Fig. 5, with a total of $\mathcal{M} \times \mathcal{N}$ subregions. A threshold T_s is set for the number of feature points extracted from every subregion. With this extraction constraint, it is unable to make the descriptors be extracted uniformly from a subregion. However, with appropriate $\mathcal{M}$ and $\mathcal{N}$, we can make the feature points come nearly uniformly from the whole image. In addition, with a constrained number of feature points, we can estimate the homography matrix properly while reducing the time for matching.

After the homography between the observed frame and a key-frame is found, we can project the observed frame into the the image plane of the key-frame. Background subtraction is performed using the local background model of the corresponding partial scene, which is described in Section II-B3. The foreground detection result is then projected back. We can fuse the foreground detection results from

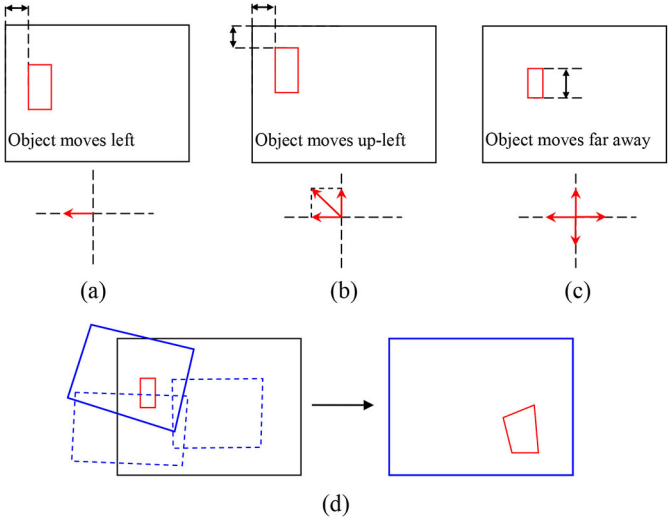


Fig. 6. Camera adjustment from tracking feedback. (a) Pan. (b) Pan and tilt. (c) Zoom. (d) Selection from key-frame correspondences across consecutive layers.

multiple key-frames. Specifically, we note that, when a pixel location has several probability values $\{\mathcal{P}_i\}_{i=1}^{n}$ of being background from related key-frames, we use the geometric average of them as the final probability $\sqrt[n]{\prod \mathcal{P}_i}$.

IV. Object Tracking With Background Subtraction

When a detected foreground object is suspicious, we would require that the PTZ camera can automatically track it and keep it in the monitoring screen. In this section, we present a general object tracking framework using the PTZ camera, by combining a tracking algorithm with the proposed hierarchical background model. In fact, any state-of-the-art object tracking algorithm can be incorporated into the presented framework. This framework relies on the tracking algorithm that we incorporate to build and update the model of the tracked object. With the proposed hierarchical background model, we are able to properly extract the foreground of the observed frame captured by the PTZ camera, and use it to refine the outputs of the incorporated tracking algorithm and thus improve the tracking performance.

Let $\mathcal{T}$ denote the incorporated tracker. It outputs the object rectangle $\mathfrak{R}_t$ that contains the tracked object in the current frame. We can refine the tracking result by locally adjusting the position and size of $\mathfrak{R}_t$, using the foreground blob that has intersection with $\mathfrak{R}_t$. Moreover, we can recover from tracking failure when the tracker drifts away due to clutter background distraction. In this situation, we can search, from the detected foreground objects nearby, for the most similar one to the target model.

We can determine whether a CA is needed according to the object rectangle $\mathfrak{R}_t$. Some CAs are illustrated in Fig. 6 as examples. The PTZ camera will be signaled to pan left when the object moves left and is close to the left side of the image rectangle (less than a predefined threshold $T_{horizon}$), as depicted by Fig. 6(a). It will cause the camera to rotate up-left if the object moves up-left [Fig. 6(b)] and triggers both the horizontal and vertical constraints ($T_{horizon}$ and $T_{vertical}$). The zooming of the camera is related to the ratio r_z^t of the size of the object to that of the image. The camera would zoom in if the ratio is less than T_{in} [Fig. 6(c)], while it would zoom out if the ratio greater than T_{out}.

Assume the image rectangle is parametrized by $[0, 0, W, H]$, where W and H are, respectively, the width and height of the image. The object rectangle is accordingly parametrized by $[x_l^t, y_u^t, x_r^t, y_b^t]$, the coordinates of its left, upper, right, and bottom sides in the current image plane. The motion vector (v_x^t, v_y^t) of the object can be roughly estimated using the position information of the last two frames. The CA can be formulated as

$$\mathtt{CA} = \begin{cases} \mathtt{pan\ left}, & \text{if } x_l^t < T_{horizon} \wedge v_x^t < 0, \\ \mathtt{pan\ right}, & \text{if } W - x_r^t < T_{horizon} \wedge v_x^t > 0, \\ \mathtt{tilt\ up}, & \text{if } y_u^t < T_{vertical} \wedge v_y^t < 0, \\ \mathtt{tilt\ down}, & \text{if } H - y_b^t < T_{vertical} \wedge v_y^t > 0, \\ \mathtt{zoom\ in}, & \text{if } r_z^t < T_{in}, \\ \mathtt{zoom\ out}, & \text{if } r_z^t > T_{out}. \end{cases} \tag{10}$$

It will reduce the cost for frame registration, if we adjust the PTZ camera to the position of some partial scene when tracking the target. When only rotation of the PTZ camera is needed, it is quite straightforward, since we can try to round the pan and tilt angles to that of an appropriate neighboring partial scene. However, when zooming is involved, additional computational effort is needed: we have to estimate the homography matrix between the observed frame and a key-frame and project the object into the coordinate plane of that key-frame to see whether the object is fully contained by it. In this situation, it would be better to use simple zooming and then register the new frame.

When the PTZ camera is currently set to the position of a partial scene, it would be much easier to find an appropriate key-frame in the neighboring layer, by using the key-frame correspondences that we store in advance. Since the image rectangles of the related key-frames in the neighboring layer are already projected to the plane of the observed frame, we just need to find the most suitable key-frame that contains the object appropriately, as depicted in Fig. 6(d).

The general object tracking framework for PTZ-based surveillance is described by Algorithm 2.

V. Experiments

Extensive experiments are carried out to verify the effectiveness of our system. First, we present the experimental results on foreground detection using our hierarchical background model for the PTZ camera, and then the performance of the proposed object tracking framework is demonstrated.

A. Implementation Details

We use a PTZ camera with the focal length ranging from 4.6 to 82.8 mm, which allows a 18× optical zoom. The ranges of its pan and tilt angles are, respectively, 350 and 120 degrees. In the experiments, a frame size of 480 × 360 is used. We

Algorithm 2 Online PTZ-Based Object Tracking With Background Subtraction

Require: Frames $\{I^t\}$ captured online by the PTZ camera, the tracker $\mathcal{T}$, the initial rectangle $\Re_1$ that contains the tracked object, and the hierarchical background model.
Output: The object rectangles $\{\Re_t\}$.
begin
  Initialization ($t = 1$): Use I^1 and $\Re_1$ to initialize the tracker $\mathcal{T}$;
  for each new frame I^t ($t > 1$) captured by the PTZ camera
    Use I^t as input to the tracker $\mathcal{T}$ and obtain $\Re_t$;
    Do frame registration and background subtraction to get the foreground mask $\mathcal{F}$;
    Use $\mathcal{F}$ to refine the tracking result or recover from tracking drift, and update $\Re_t$;
    Update the tracker $\mathcal{T}$;
    Find out whether camera adjustment is necessary;
    if zoom is needed **then**
      Adopt simple zoom or use the key-frame correspondences to find out the most suitable partial scene, and adjust the camera;
    else if rotation is needed **then**
      Determine the rotation parameters and adjust the camera;
    end
    Use the camera parameters to project the object location into the new image plane if camera adjustment happens;
  end
end

found that $N = 3$ layers is enough for our experiments. We ran the experiments on a PC with Intel i5-3570K processor (four 3.4 GHz cores) and 16 GB memory.

We implement the proposed method in C++ with the OpenCV library. Besides the proposed SCS-SILTP descriptor, three state-of-the-art background subtraction methods used for stationary cameras, including MoG [8], Bayes decision ("Bayes") for complex scenes [13], "SILTP" [15], are also incorporated into our hierarchical background model for comparison. We use both the MoG and Bayes algorithms implemented in OpenCV2.4.5 with default parameters. For the SILTP operator, we use $\bar{N} = 8$, $R = 1$, and $\tau = 0.05$. For the SCS-SILTP operator, we set $R = 2$ and $\tau = 0.05$, and use three adjacent frames, with $\bar{N}_0 = 8$ and $\bar{N}_1 = 4$. In kernel density estimation, both SILTP and SCS-SILTP use the same set of parameters: $K = 5$, $T_m = 0.08$, $T_b = 0.8$, $T_{bg} = 0.01$, and $\rho = 0.005$. In addition, we use a Gaussian with standard deviation 1.2 for $g(x)$. Since the Hamming distance $d(p, q)$ is an integer, we can precalculate $g(x)$ taking several integer values and store them. Then, in later calculation, we can efficiently obtain the result via a look-up table. When extracting feature points from the observed frame, we set $\mathcal{M} = 2$, $\mathcal{N} = 2$, and $T_s = 200$.

Regarding the feature points, we tested three features, i.e., SIFT [35], SURF [36], and ORB [37]. In our experiments, we observe that SIFT is most robust and distinctive among the three, but at the cost of highest computation time. The extraction of ORB is at least an order faster than SIFT and is also several times faster than SURF, though ORB seems to perform worse than SIFT and SURF in large-scale variations. However, we observe that ORB works well in the frame registration process, for we register the observed frame to the key-frames with a similar scale. As mentioned above, we can use fast approximate nearest neighbor search to speed up the feature matching process. Multiprobe locality sensitive hashing (LSH) [39] can be used for ORB matching, while randomized KD-trees can be used for SIFT and SURF. These

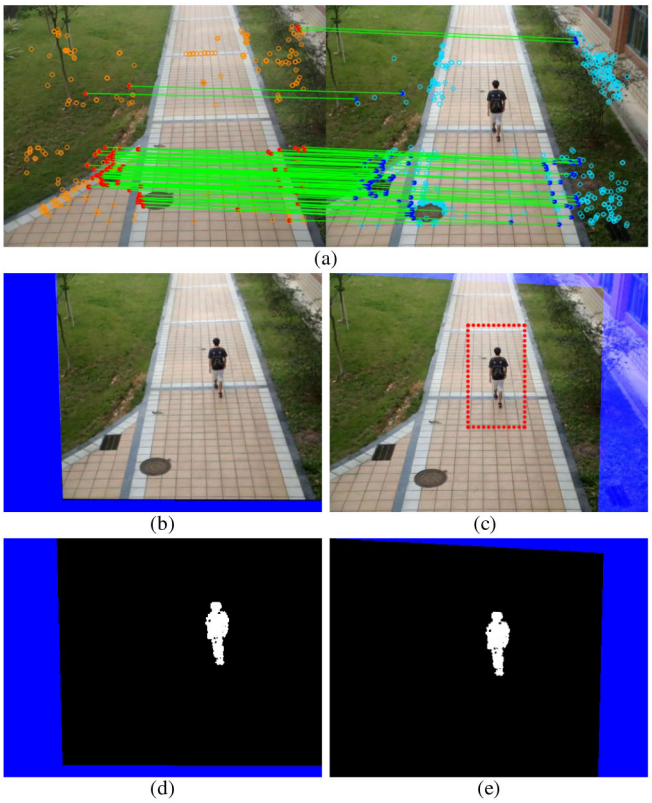


Fig. 7. Example of homography transform and foreground detection. (a) The estimated homography between a key-frame and the observed frame. (b) Observed frame projected to the key-frame. (c) The corresponding shared region of observed frame. (d) Foreground detection. (e) Foreground result projected back to the observed frame.

fast approximate nearest neighbor algorithms are also provided by OpenCV2.4.5 with the FLANN library [38], and LSH is faster than the KD-trees. Therefore, we use ORB features in the experiments, and use 16 hashing tables for each key-frame to store the feature points. We adopt the measure used in [37] to select the required number of ORB feature points.

An example of homography transform and foreground detection is shown in Fig. 7. We show some of the extracted feature point locations in Fig. 7(a), and connect with green lines the correct location correspondences found by RANSAC. The projected regions shared by the observed frame and the key-frame are depicted in Fig. 7(b) and (c), in which unshared regions are marked in blue. Fig. 7(d) is the foreground detection result by subtracting the local background model, while the projected-back result into the observed frame is shown in Fig. 7(e). It can be seen from Fig. 7(e) that some region (in blue) of the observed frame is unprocessed when using one key-frame. It can be solved by using multiple key-frames if we want to cover the whole observed frame and detect all the foreground objects. However, for tracking an object of interest [e.g., the boy in Fig. 7(c)], we are concerned the image regions surrounding the object, as illustrated by the dashed-line rectangle, have been processed so as to assist tracking. Therefore, in this case, foreground extraction can be achieved using only one key-frame, as revealed in Fig. 7(e). In this way, we can achieve better computational

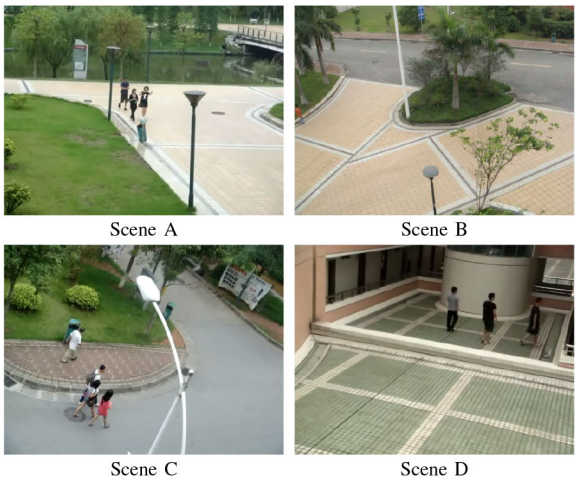


Fig. 8. Scenes used for evaluation of foreground detection. Depth variations in the four scenes, except Scene B, are quite apparent from the motion of the camera.

efficiency. As a result, foreground objects that fall into the unprocessed image region will not be detected. In general, this key-frame can be picked out from the set of neighboring key-frames $Set(I^t)$ by the minimum sum of $\|\alpha - \alpha^t\|$ and $\|\beta - \beta^t\|$ in (9). However, this selection step will be iteratively carried out until the selected key-frame guarantee that the predefined surrounding region of the tracked object is processed.

We implement the PGM background model [22] for comparison. Some of the key-frames are manually chosen to generate the panoramic frame as described in [22], and 100 panoramic frames are used to initialize the PGMM. In addition, layered correspondence ensemble and SURF features are used for registering the observed frame to the panoramic background image.

B. Experiment Results on PTZ Foreground Detection

Four scenes are used to evaluate the foreground detection performance under the PTZ camera, as shown in Fig. 8. We note that the four scenes, except Scene B, exhibit obvious depth variations when the PTZ camera rotates. Scene B is a wide crossing. The objects in the scene are a distance away from the position of the camera, and the camera rotates in a relatively small range. While for the other three scenes, when the camera rotates, the distance of the objects in the scene to the position of camera would vary from several meters to tens of meters away. Two test videos are captured for each scene, with the PTZ camera rotating and zooming. The test videos varies in length, each with 20 frames manually labeled by us as the ground truth.

In PTZ-based surveillance, it is more critical to localize positions of foreground objects rather than obtaining accurate segmentation, so we evaluate the background subtraction algorithms in the object-level, which can also avoid pixel-level inaccuracy caused by manual labeling and image projection. A connected foreground blob is considered to be a foreground object. We eliminate small blobs less than 20 pixels. If a detected object has intersection with a foreground object annotated in the ground truth and satisfies the following formula, it is considered to be correctly detected:

$$\frac{S_{DT} \cap S_{GT}}{S_{DT} \cup S_{GT}} > T_{DT} \tag{11}$$

where S_{DT} and S_{GT} are, respectively, the areas of the detected object and the ground truth object, and T_{DT} is constant threshold that we set to 0.5 in the experiments.

By ground truth annotation, we can obtain a binary image for a frame, in which pixels belonging to foreground objects are marked as white and background pixels as black. Some foreground objects may weakly connect to each other and together form a big foreground blob in the ground truth image. In the experiments, we further annotate a mask image, in which curves are drawn to separate the weakly connected foreground objects. In this way, the foreground detection image, generated by a background subtraction algorithm, and the ground truth image will be imposed with the mask before they are further processed to find correctly detected objects.

The precision (the number of correctly detected objects to that of all the detected objects) and the recall (the number of correctly detected objects to that of the annotated ground truth objects) are used as evaluation measures. Table II shows the precision and recall evaluation of the compared background subtraction methods. The best precisions are marked in red bold fonts, and the second best are marked in red fonts with underlines. The best recalls are marked in blue bold fonts, while the second best are marked in blue fonts with underlines. As can be seen from the table, the proposed hierarchical background model with SCS-SILTP does best in most of the eight tested videos. Among the four background subtraction methods used in the hierarchical model, SILTP ranks second, followed by Bayes, and MoG comes last. Regarding PGMM, it suffers much from image distortion and is greatly affected by the panoramic image generation results. It can be seen for Table II that PGMM gives a low recall in the eight tested videos. In addition, we notice that PGMM does better in Scene B than the other three scenes. In fact, we do not apply large rotation angles in Scene B and there is no obvious depth variations in the scene. As a result, the resulted panoramic image is more smooth, with less image distortion introduced by stitching. In summary, compared with PGMM that uses a panoramic background model, the proposed hierarchical background model works better under the PTZ camera, especially for scenes with obvious depth variations.

In addition, example foreground detection results of different background subtraction methods are demonstrated in Fig. 9, with a representative frame from each test video. One of the related key-frames of the selected frame is also shown. For the detection results, we mark the classified backgrounds with green color.

In the experiments, it takes roughly 34 ms for the proposed hierarchical model to register the observed frame to one key-frame. For foreground detection after registration, it takes 17, 26, 65, 67 ms, respectively, for our model with MoG, Bayes, SILTP, and SCS-SILTP. While it takes 126 ms for PGMM to register the observed frame to the panoramic image, and the foreground detection stage takes 24 ms.

TABLE II
OBJECT-LEVEL EVALUATION OF FOREGROUND DETECTION (%)

Videos	PGMM		MoG		Bayes		SILTP		SCS-SILTP	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
A1	68.1	45.8	80.6	80.6	90.3	86.7	86.1	88.6	**97.2**	**92.1**
A2	62.5	40.2	82.1	70.8	60.7	85.0	89.3	80.6	**92.9**	**86.7**
B1	83.7	58.1	88.4	76.0	90.7	78.0	93.0	80.0	**95.3**	**83.7**
B2	74.1	62.4	70.6	70.6	77.6	71.7	88.2	79.8	**91.8**	**82.1**
C1	67.4	53.6	80.9	70.6	78.7	75.3	85.4	76.8	**89.9**	**79.2**
C2	72.9	48.3	88.1	64.2	84.7	**74.6**	89.8	65.4	**91.5**	73.0
D1	61.1	47.3	**86.1**	75.6	77.8	78.9	79.2	75.0	84.7	**79.2**
D2	64.8	42.2	87.0	65.3	88.9	67.6	88.9	68.6	**90.7**	**73.1**
Total	68.9	49.7	82.1	71.4	80.8	76.6	87.0	76.6	**91.5**	**81.0**

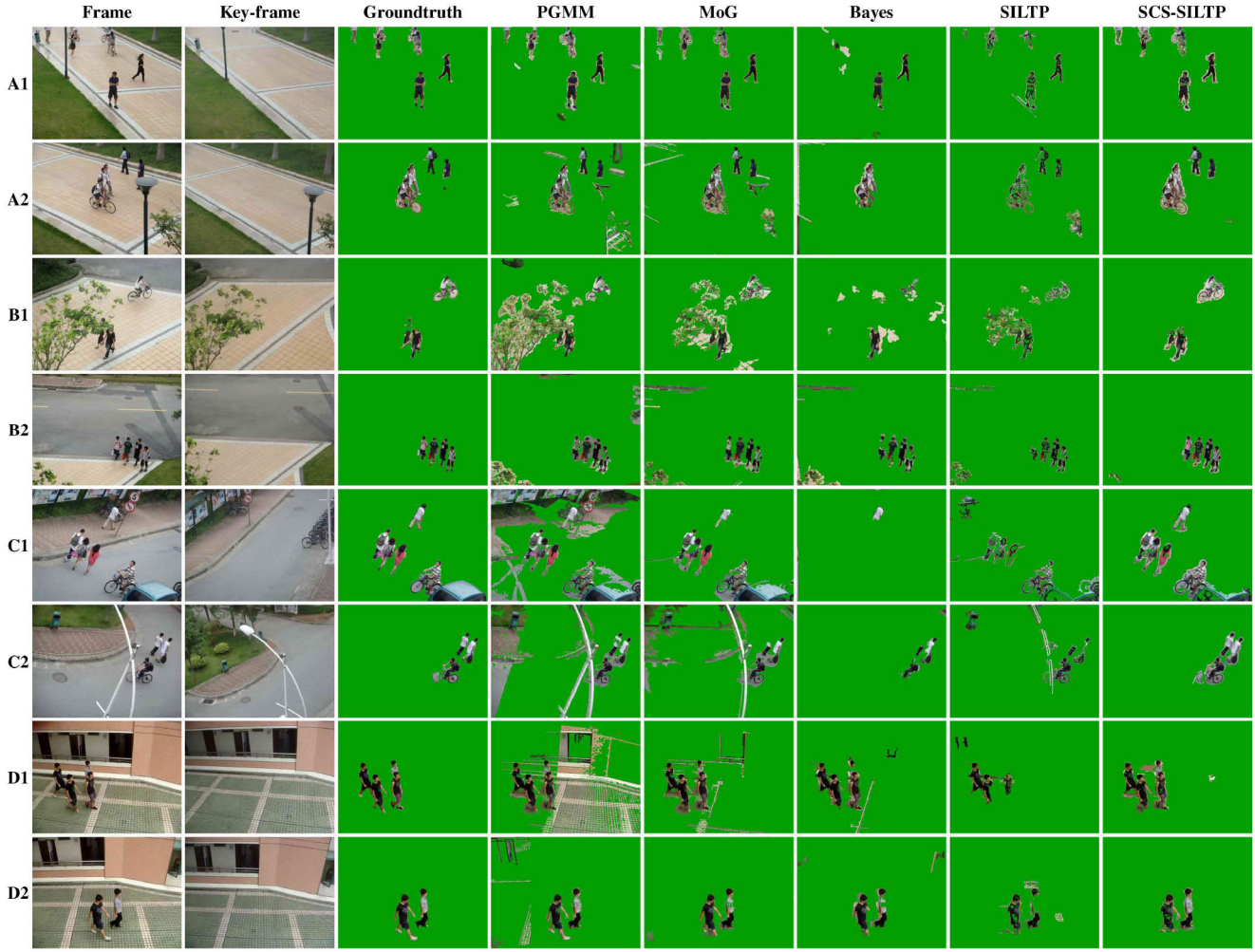


Fig. 9. Foreground detection results of different background subtraction methods.

Evaluation on Parameter Selection: We evaluate the effect of important parameters on the system performance quantitatively. The overall precision and recall measures mentioned above are adopted. The important parameters defined in Section II-B are evaluated. We fix the values of the other parameters as set in Section V-A when changing the value of a specific parameter. The evaluation results of the parameters τ, T_{bg}, and T_m are plotted in Fig. 10. The precision is at its peaks and is relatively stable when the scale factor τ is between 0.04 and 0.06. More pixels are classified as foreground when τ gets smaller, but these pixels do not connect into correct foreground objects and the precision decreases. The recall also decreases because more background pixels are falsely classified as foreground. On the contrary, fewer pixels are classified as foreground when τ gets larger, causing the precision to decrease. However, the recall increases instead since fewer background pixels are falsely classified. The precision and recall almost stay the same when the background threshold T_{bg} is around 0.01. They decrease slowly when T_{bg} gets larger. Quick drops may sometimes happen (e.g., when T_{bg} is in [0.07, 0.08]), which can be attributed to that the pattern distance $d(p, q)$ only outputs discrete integer

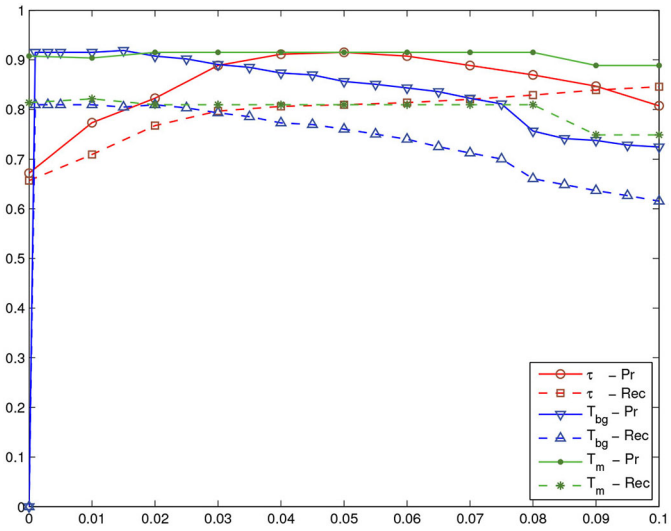


Fig. 10. Precision (Pr) and recall (Rec) plots for different parameters of the proposed background model.

values. For the same reason, the precision and recall are nearly piecewise constant with respect to the matching threshold T_m, and they have some sudden changes when T_m takes specific values (e.g., at around 0.01 and 0.09). The precision is at its peak when T_m is in [0.02, 0.08]. For the threshold T_b, we find that the system performance is the best and is relatively stable when it is in [0.7, 0.8]. We also note that it is most appropriate for the number of maintained patterns K to take 3–5, because fewer maintained patterns are not able to model dynamic background, while more maintained patterns would result in little performance gain at high computational cost.

C. Experiment Results on PTZ Tracking

We incorporate two state-of-the-art tracking methods into the proposed PTZ object tracking framework, and evaluate the improved tracking performance achieved by combining the proposed hierarchical background model.

The ensemble tracking (ET) algorithm [24] and a recent tracking-by-detection algorithm (Struck [28]) are incorporated into the proposed framework for evaluation. For the ET tracker [24], we follow the parameter settings of [24], and use a simple particle filter to handle occlusion as in [24]. The target is manually marked with a bounding box in the first frame. The ET tracker does not handle scale changes of the target. In the experiments, we do scale adaption of the target with respect to the foreground detection result and the confidence map generated by the ET tracker. The confidence map provides the probability of pixels belonging to the tracked object. A pixel is said to belong to the target if it is foreground with the probability higher than 0.4. Assume δ is the difference of the newly detected width and the original width of the target. We will change the width by half of δ to avoid over-adaptation. The height of the target is dealt with likewise. For the struck tracker [28] that is built on patched-based detection, we use the source codes provided by the authors. Scale adaptation of the target is performed via the foreground detection results, similar to the case of the ET tracker. The patches are scaled to the size of the target model.

In the experiments, we evaluate the proposed object tracking framework both offline and online, where offline means that the tracking results are not fed back to the PTZ controller for CAs. Four videos are captured in advance to do the offline evaluation. As you can see from Fig. 11, three of them are captured in the four scenes shown in Fig. 8. The 4th video captures a boy going away from the camera, and the scene is quite simple, which can be captured by the camera with a small focus length. For comparison, we run the trackers alone on the four videos, and then with PGMM and with the proposed framework, respectively. The foreground detection is performed every three frames when no CA happens. To save computation time, we run the tracker and foreground detection on a downsampled image with half of the frame size (i.e., 240×180). On average, the ET tracker alone, with PGMM and with our model runs, respectively, 23.0, 14.7, and 11.6 frames/s, while the Struck tracker alone, with PGMM and with our model runs, respectively, 17.8, 11.4, and 9.2 frames/s. Example frames from the tracking results are shown in Fig. 11, also with the foreground detection results. As you can observe, our foreground detection results are much better than that of PGMM. We also note that some bad detection results of PGMM in Fig. 11(d) are caused by bad registration, where few feature point correspondences between the observed frame and the panorama are found. The proposed method works much better by registering the observed frame to the neighboring key-frames.

To quantitatively evaluate the object tracking performance, we apply a tracking precision measure, which is the ratio between the numbers of correctly tracking frames and the total frames for each video, as reported in Table III. A frame is counted as correctly tracked only if the ratio of the overlap between the detected and annotated boxes to the union of them is over 0.5. As can been seen, the tracking performance has been greatly improved when the trackers with our method than themselves alone, and it is also better than with PGMM.

We further combine the ET tracker to evaluate the proposed object tracking framework online, where the tracking results are fed back to control the PTZ camera. A separate thread is used to capture the frames from the PTZ camera and control it. In addition, we perform foreground detection every five frames or when CA happens. The tracked object does not move before we mark it in the screen and finish the initialization. An example is shown in Fig. 12, which run at 15.4 frames/s. As you can notice, the scene is very large. We also run the ET tracker with PGMM on the video offline, but it fails at the first few frames due to bad registration and only works sightly better than the ET tracker alone, which can be inferred from Table III. The tracking precisions of the Struck tracker on the video are also provided. The proposed framework improves its performance greatly.

In Fig. 12, the resulting trajectory obtained by running the ET tracker with the proposed framework online is demonstrated in the hierarchical background model. The trajectory reflects the movement of the bottom middle point of the tracked box. Some key-frames from different layers are shown,

Fig. 11. Tracking results of Videos 1–4. The background subtraction results of PGMM (2nd row) and Ours (3rd row) are also shown. (a) Video 1. (b) Video 2. (c) Video 3. (d) Video 4.

TABLE III
PRECISION COMPARISON. (A) TRACKER ALONE. (B) WITH PGMM [22]. (C) WITH THE PROPOSED METHOD

Methods		Videos					Avg.
		1	2	3	4	5	
ET tracker [24]	(A)	0.4465	0.1642	0.8290	0.1446	0.1944	0.2726
	(B)	0.8582	0.1642	0.7120	0.9703	0.2289	0.4662
	(C)	0.9065	0.8189	0.9250	0.9822	0.9550	0.9392
Struck tracker [28]	(A)	0.1719	0.4642	0.7417	0.1446	0.1345	0.2285
	(B)	0.4766	0.4642	0.6789	0.9515	0.1842	0.4202
	(C)	0.6501	0.8887	0.9127	0.9762	0.9104	0.8908

and the generated trajectory of our method is mapped onto the key-frames in red. Meanwhile, the ground truth trajectory is also generated in dashed black lines for comparison.

Discussion: In this paper, some simple criteria are provided for adjusting the PTZ camera. They work well for some cases, but better criteria can be introduced. Considering that neither the foreground detection result from the background model NOR the performance of existing tracking methods is reliable enough, it would be better measurements to fuse the two more properly. In addition, more robust CAs should be studied. The CAs can be affected by the mechanical imprecision of the PTZ controller. A large adjustment may cause the tracked object to fall out of the screen, while a small adjustment would lead to frequent CAs. However, these problems go beyond the scope of this paper and will be addressed by future work.

VI. CONCLUSION

In this paper, a solution for the PTZ camera-based surveillance system has been proposed. The system is divided into three parts: hierarchical background modeling, frame registration, and object tracking. The hierarchical background model and frame registration can detect moving foreground objects, which is a basic and important step for applications with the PTZ camera. Furthermore, a general object

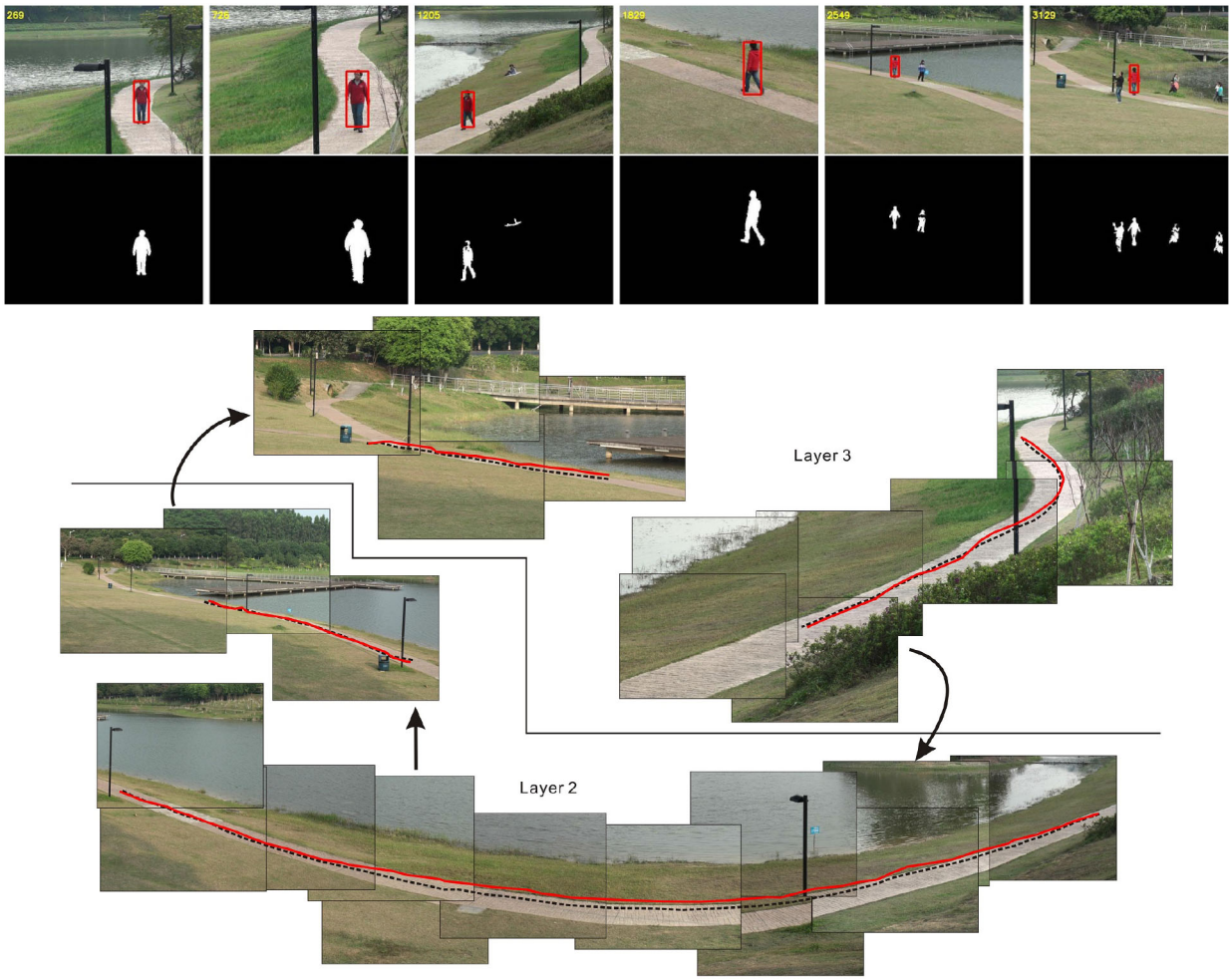


Fig. 12. Online tracking results of Video 5 and the resulted trajectory in the hierarchical background model.

tracking framework that combines the proposed PTZ background model and existing object tracking algorithms is put forward. The effectiveness of the proposed method is verified through extensive experiments.

We will focus on several issues of the proposed framework in the future work. First, in order to decide conveniently, for different scenes, the key-frame ensemble of the hierarchical model, a friendly interactive user interface or a good automatic algorithm should be developed. Second, the tracking feedback and CA should be made feasible and flexible with the parameters of different PTZ cameras. Finally, more background subtraction algorithms will be incorporated into the background model for evaluation, and we would also like to make the proposed surveillance system easier to compatible with different tracking methods.

ACKNOWLEDGMENT

The authors would like to thank L. Chen and J. Su who partly joined this paper when they were postgraduate students at Sun Yat-sen University, and they also thank K. Chen for assistance in the experimental evaluation. H. Wu would like to thank Prof. Q. Chen from Guangdong University of Education for support while doing this paper. They would also like to thank the Editors and Reviewers for their valuable suggestions on improving the quality of the paper.

REFERENCES

[1] L. Lin, Y. Lu, Y. Pan, and X. Chen, "Integrating graph partitioning and matching for trajectory analysis in video surveillance," *IEEE Trans. Image Process.*, vol. 21, no. 12, pp. 4844–4857, Dec. 2012.

[2] S. Kamijo, Y. Matsushita, K. Ikeuchi, and M. Sakauchi, "Traffic monitoring and accident detection at intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 2, pp. 108–118, Jun. 2000.

[3] Y. Xu and D. Song, "Systems and algorithms for autonomous and scalable crowd surveillance using robotic PTZ cameras assisted by a wide-angle camera," *Auton. Robot.*, vol. 29, no. 1, pp. 53–66, Jul. 2010.

[4] K.-T. Song and J.-C. Tai, "Dynamic calibration of pan–tilt–zoom cameras for traffic monitoring," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 5, pp. 1091–1103, Oct. 2006.

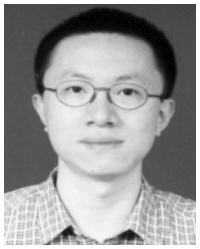
[5] P. D. Z. Varcheie and G.-A. Bilodeau, "Adaptive fuzzy particle filter tracker for a PTZ camera in an IP surveillance system," *IEEE Trans. Instrum. Meas.*, vol. 60, no. 2, pp. 354–371, Feb. 2011.

[6] Y. Yao *et al.*, "Can you see me now? Sensor positioning for automated and persistent surveillance," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 1, pp. 101–115, Feb. 2010.

[7] C. Ding, B. Song, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury, "Collaborative sensing in a distributed PTZ camera network," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3282–3295, Jul. 2012.

[8] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Conf. CVPR*, Fort Collins, CO, USA, 1999, pp. 246–252.

[9] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proc. 2nd Eur. Workshop AVBS*, 2001, pp. 1–5.
[10] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proc. IEEE ICPR*, Washington, DC, USA, 2004, pp. 28–31.
[11] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, "Background and foreground modeling using nonparametric Kernel density estimation for visual surveillance," *Proc. IEEE*, vol. 90, no. 7, pp. 1151–1163, Jul. 2002.
[12] K. Kim, T. Chalidabhongse, D. Harwood, and L. Davis, "Real-time foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, 2005.
[13] L. Li, W. Huang, I. Gu, and Q. Tian, "Foreground object detection from videos containing complex background," in *Proc. 11th ACM Int. Conf. Multimedia*, Berkeley, CA, USA, 2003, pp. 2–10.
[14] M. Heikkilä and M. Pietikäinen, "A texture-based method for modeling the background and detecting moving objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 657–662, Apr. 2006.
[15] S. Liao, G. Zhao, V. Kellokumpu, M. Pietikainen, and S. Z. Li, "Modeling pixel process with scale invariant local patterns for background subtraction in complex scenes," in *Proc. IEEE Conf. CVPR*, San Francisco, CA, USA, 2010, pp. 1301–1306.
[16] O. Barnich and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences," *IEEE Trans. Image Process.*, vol. 20, no. 6, pp. 1709–1724, Jun. 2011.
[17] S. Kang, J. Paik, A. Koschan, B. Abidi, and M. A. Abidi, "Real-time video tracking using PTZ cameras," in *Proc. Int. Conf. QCAV*, 2003, pp. 103–111.
[18] S. Wu, T. Zhao, C. Broaddus, C. Yang, and M. Aggarwal, "Robust pan, tilt and zoom estimation for PTZ camera by using meta data and/or frame-to-frame correspondences," in *Proc. ICARCV*, Singapore, 2006, pp. 1–7.
[19] A. Bevilacqua, L. D. Stefano, and P. Azzari, "An effective real-time mosaicing algorithm apt to detect motion through background subtraction using a PTZ camera," in *Proc. IEEE Int. Conf. AVSS*, 2005, pp. 511–516.
[20] S. N. Sinha and M. Pollefeys, "Pan-tilt-zoom camera calibration and high-resolution mosaic generation," *Comput. Vis. Image Und.*, vol. 103, no. 3, pp. 170–183, Sep. 2006.
[21] M. Irani and P. Anandan, "A unified approach to moving object detection in 2D and 3D scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 6, pp. 577–589, Jun. 1998.
[22] K. Xue, Y. Liu, G. Ogunmakin, J. Chen, and J. Zhang, "Panoramic gaussian mixture model and large-scale range background substraction method for PTZ camera-based surveillance systems," *Mach. Vis. Appl.*, vol. 24, no. 3, pp. 477–492, Apr. 2013.
[23] C. Guillot *et al*., "Background subtraction adapted to PTZ cameras by keypoint density estimation," in *Proc. BMVC*, 2010, pp. 34.1–34.10.
[24] S. Avidan, "Ensemble tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 261–271, Feb. 2007.
[25] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Proc. IEEE ICCV*, Kyoto, Japan, 2009, pp. 1515–1522.
[26] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. IEEE Conf. CVPR*, Miami, FL, USA, 2009, pp. 983–990.
[27] X. Liu, L. Lin, S. Yan, H. Jin, and W. Jiang, "Adaptive object tracking by learning hybrid template online," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 11, pp. 1588–1599, Nov. 2011.
[28] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE ICCV*, Barcelona, Spain, 2011, pp. 263–270.
[29] H. Wu, G. Li, and X. Luo, "Weighted attentional blocks for probabilistic object tracking," *Visual Comput.*, vol. 30, no. 2, pp. 229–243, 2014.
[30] L. Lin, T. Wu, J. Porway, and Z. Xu, "A stochastic graph grammar for compositional object representation and recognition," *Pattern Recognit.*, vol. 42, no. 7, pp. 1297–1307, 2009.
[31] L. Lin, X. Liu, and S.-C. Zhu, "Layered graph matching with composite cluster sampling," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1426–1442, Aug. 2010.
[32] M. Heikkilä, M. Pietikäinen, and C. Schmid, "Description of interest regions with local binary patterns," *Pattern Recognit.*, vol. 42, no. 3, pp. 425–436, 2009.
[33] X. Tan and B. Triggs, "Enhanced local texture feature sets for face recognition under difficult lighting conditions," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1635–1650, Jun. 2010.
[34] A. Mittal and D. Huttenlocher, "Scene modeling for wide area surveillance and image synthesis," in *Proc. IEEE Conf. CVPR*, Hilton Head Island, SC, USA, 2000, pp. 160–167.
[35] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proc. IEEE ICCV*, Kerkira, Greece, 1999, pp. 1150–1157.
[36] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. ECCV*, Graz, Austria, 2006, pp. 404–417.
[37] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE ICCV*, Barcelona, Spain, 2011, pp. 2564–2571.
[38] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proc. Int. Conf. Comput. Vis. Theory Appl.*, 2009, pp. 331–340.
[39] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, "Multi-probe LSH: Efficient indexing for high-dimensional similarity search," in *Proc. 33rd Int. Conf. VLDB*, Vienna, Austria, 2007, pp. 950–961.
[40] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981.



Ning Liu received the B.S. degree in computational mathematics from Northwest University, Xi'an, China, and the Ph.D. degree in computer science from Sun Yat-sen University (SYSU), Guangzhou, China, in 1996 and 2004, respectively.

He is currently an Associate Professor with the School of Software, SYSU. He has served as a Reviewer for several important conferences and journals. His current research interests include computer vision and machine learning algorithms.

Hefeng Wu received the B.S. and Ph.D. degrees in computer science and technology from Sun Yat-sen University (SYSU), Guangzhou, China, in 2008 and 2013, respectively.

He is currently a Post-Doctoral Research Scholar with the School of Information Science and Technology, SYSU, and is also a member of National Engineering Research Center of Digital Life, SYSU. His current research interests include image/video analysis, machine learning, and computer vision.

Liang Lin received the B.S. and Ph.D. degrees from the Beijing Institute of Technology, Beijing, China, in 1999 and 2008, respectively.

He is a Full Professor with the School of Advanced Computing, Sun Yat-Sen University (SYSU), Guangzhou, China. From 2006 to 2007, he was a joint Ph.D. student with the Department of Statistics, University of California, Los Angeles (UCLA), Los Angeles, CA, USA. He served as a Post-Doctoral Research Fellow at the Center for Vision, Cognition, Learning, and Art of UCLA. His current research interests include new models, algorithms, and systems for intelligent processing and understanding of visual data such as images and videos. He has published over 50 papers in top tier academic journals and conferences including PROCEEDINGS OF THE IEEE, TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, TRANSACTIONS ON IMAGE PROCESSING, TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, TRANSACTIONS ON MULTIMEDIA, *Pattern Recognition, CVPR, ICCV, ECCV, ACM MM,* and *NIPS*.

Prof. Lin was supported by several promotive programs or funds for his works, such as the Program for New Century Excellent Talents of Ministry of Education, China, in 2012, the Program of Guangzhou Zhujiang Star of Science and Technology in 2012, and the Guangdong Natural Science Funds for Distinguished Young Scholars in 2013. He received the Best Paper Runners-Up Award in ACM NPAR 2010, the China National Excellent Ph.D. Thesis Award Nomination in 2010, for the Ph.D. thesis, and the Google Faculty Award in 2012.