Spatiotemporal Tree Filtering for Enhancing Image Change Detection

Dawei Li, *Member, IEEE*, Siyuan Yan, Mingbo Zhao, *Senior Member, IEEE*, and Tommy W. S. Chow, *Fellow, IEEE*

***Abstract*— Change detection has received extensive attention because of its realistic significance and broad application fields. However, none of the existing change detection algorithms can handle all scenarios and tasks so far. Different from the most of contributions from the research community in recent years, this paper does not work on designing new change detection algorithms. We, instead, solve the problem from another perspective by enhancing the raw detection results after change detection. As a result, the proposed method is applicable to various kinds of change detection methods, and regardless of how the results are detected. In this paper, we propose Fast Spatiotemporal Tree Filter (FSTF), a purely unsupervised detection method, to enhance coarse binary detection masks obtained by different kinds of change detection methods. In detail, the proposed FSTF has adopted a volumetric structure to effectively synthesize spatiotemporal information of the same target from the current time and history frames to enhance detection. The computational complexity analyzed in the view of graph theory also show that the fast realization of FSTF is a linear time algorithm, which is capable of handling efficient on-line detection tasks. Finally, comprehensive experiments based on qualitative and quantitative analysis verify that FSTF-based change detection enhancement is superior to several other state-of-the-art methods including fully connected Conditional Random Field (CRF), joint bilateral filter, and guided filter. It is illustrated that FSTF is versatile enough to also improve saliency detection as well as semantic image segmentation.**

***Index Terms*— Change detection, binary mask enhancement, tree filtering, spatiotemporal filtering, post-processing.**

I. Introduction

THE purpose of "change detection" is to detect areas of change in a series of images taken at different times at the same scene. It is the basis of intelligent video analysis such as target tracking and action recognition. Change detection has received extensive attention due to its practical significance and broad applications. Important applications of change detection include video surveillance [1], [2], remote sensing [3]–[5], [6], [7], medical diagnosis and treatment [8], etc.

The earliest image change detection method is frame differencing. Jain and Nagel [9] propose a frame differencing method to extract moving objects. Though the computational speed of frame differencing is fast, it is only suitable for simple scenes with static backgrounds and cameras. Then, Aach and Kaup [10] improve the frame differencing using hypothesis testing and Markov Random Field (MRF). In order to remove ghosts caused by frame differencing, background modeling for change detection later becomes popular. Stauffer and Grimson [11] use Gaussian Mixture Model (GMM) for background modeling and foreground subtraction. Despite its robustness against complex disturbances, early versions of GMM are unable to achieve satisfactory results at situations of target stagnation, serious camera jitter, sudden illumination changes, and so on. Therefore, many approaches have been proposed to improve GMM. Zivkovic [12] employ GMM with changeable number of Gaussian distributions at each pixel to improve adaptability and to simplify computation. An illumination-robust foreground detection method based on GMM with adaptive Gaussians is presented by Li *et al*. [13]. Different from typical statistical background modeling methods, unsupervised change detection algorithms often solve the detection problem in heuristic ways, and solutions emerge in an endless stream in recent years. In [14], a nonparametric kernel density estimation is proposed to detect foregrounds, and it reduces negative effect from shadows to some extent by considering the srgb color space information. ViBe [15] leverages a random background updating strategy to reduce the complexity of background modeling and obtains an accelerated detection speed. Change detection based on Robust Principal Component Analysis (RPCA) has also been progressing rapidly in recent years. Candes *et al*. [16] show that RPCA can be applied to moving object detection by modeling the background with a low-rank subspace and representing the foreground objects with a noise component. Zhou *et al*. [17] propose a batch algorithm—DECOLOR that combines RPCA and MRF to separate foreground objects from images. Gao *et al*. [18] impose spatial coherence of foreground components to refine the mechanism of change detection using decomposition. Liu *et al.* [19] take group properties of foregrounds into account on both spatial and temporal domains for sparsity recovery in background subtraction. Though RPCA-type methods boast solid theoretical basis and effective change detection results, the computational speed is

Manuscript received January 20, 2020; revised July 22, 2020; accepted August 13, 2020. Date of publication August 24, 2020; date of current version September 9, 2020. This work was supported in part by the Natural Science Foundation of Shanghai under Grant 20ZR1400800, in part by the National Natural Science Foundation of China under Grants 11972115, 61971121, and 61806051; and also in part by Funds for the Central Universities and DHU Distinguished Young Professor Program. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Lisimachos P. Kondi. *(Corresponding author: Mingbo Zhao.)*

Dawei Li, Siyuan Yan, and Mingbo Zhao are with the College of Information Sciences and Technology, Donghua University, Shanghai 201620, China, and also with the Engineering Research Center of Digitized Textile and Fashion Technology, Ministry of Education, Donghua University, Shanghai 201620, China (e-mail: daweili@dhu.edu.cn; mzhao4@dhu.edu.cn).

Tommy W. S. Chow is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong (e-mail: eetchow@cityu.edu.hk).

Digital Object Identifier 10.1109/TIP.2020.3017339

usually slow. With the rapid development of deep learning for computer vision tasks, semi-supervised and supervised foreground segmentation algorithms based on deep learning [20], [21] are sprouting up, and the segmented foreground masks can serve directly for the purpose of detection. In order to provide a large amount of data for various supervised detection algorithms and a benchmark for comparison of various algorithms, large change detection datasets emerged. The most popular one is the CDnet2012 dataset [22]. It contains 6 video categories with 4 to 6 videos sequences in each category. BMS-26 dataset [23] consists of 26 video sequences with pixel-level annotated moving objects. Daimler Pedestrian Benchmark dataset [24] consists of many pedestrian images captured from a vehicle-mounted calibrated stereo camera rig in urban environments.

In practice, many change detection algorithms cannot deliver satisfactory binary results under heavy noise, and others tend to generate under-detected or over-smoothed foreground contours. To date, none of the existing change detection algorithms can deliver satisfactory results under all scenarios and tasks. On one hand, we can always design and improve change detection algorithms. But on the other hand, we can also solve the problem by enhancing the raw detection results after change detection, regardless of how they are detected. The earliest enhancement method is perhaps the morphological operation [25], [26]. It exploits the direct relationship between a pixel and its neighboring pixels to improve detection masks. However, this method needs to design different combinations of morphological operators for different binary masks. For example, one must have prior knowledge that whether the detection is over-smoothed or under-detected so as to apply an erosion operator or a dilation operator respectively. Also, the iteration times of morphological operation are usually determined by experience and observation. Redundant dilations amplify detection noise and bring about the problem of false-positive targets. Redundant erosions easily remove under-detected targets, leading to false-negatives. With the rising popularity of Markov Random Fields (MRFs) during the last few decades, image denoising, restoration, and other image processing operations have found harder theoretical basis. MRFs are generally used to calculate the *a priori* probability of the pixel to be affected by its surrounding pixels, and the maximum *a posteriori* (MAP) probability is then iteratively calculated to improve the detection masks. By using the simulated annealing method with a decreasing temperature parameter of the Gibbs distribution, the framework of MAP-MRF can reach rapid convergence [13], [27], [28]. MRFs are especially suitable for improving results obtained from statistical/probabilistic foreground detection methods. Limited by the high computation burden, MRFs usually perform in a local window, which restricts its smoothing effort to a local scale. Conditional Random Field (CRF) considers both unary potentials and pairwise potentials of pixels. The traditional CRF is also applied locally to improve image segmentation results due to the huge computation burden [29], [30]. However, reference [31] proposes a fully connected CRF for global enhancement based on mean field approximation and high-dimensional filtering, which significantly reduces the computation cost. Since then, CRFs are extended to full size images. A fully-connected CRFs are currently widely used for post-processing of image detection or segmentation results [32], [33].

Recently, image filtering becomes a popular and important research topic for their performances on image/video denoising and enhancement. Bilateral filter [34] is a local image filter that considers both spatial relation and color similarity among pixels. Given enough computing power, the bilateral filtering sometimes generates significant improvements. Several accelerated versions have emerged to improve the filtering efficiency [35]–[37]. Since bilateral filter, new image filters have been emerging rapidly. He *et al.* [38] present the guided filter, a local linear filter utilizing a guidance image for enhancing the input image. This kind of filter exhibits excellent edge-preserving capability with low computational complexity. However, if an input image is highly noisy, this method tends to amplify the noise because the filter runs with local information. Yang [39], thus, propose a tree filter to handle the cost aggregation process in stereo matching in a global way. Unlike other local image filters, the tree filter aggregates information on the minimum spanning tree generated from the whole guidance image. Every pixel of the image can transfer information to any other pixels through paths of the minimum spanning tree. Local filters have limited filtering effects for the reason that they only consider the relationship among the pixels within a window; while a global filter considers long-range correlation between any pair of pixels, making full use of the domain information.

For the case of videos, the same foreground object apparently might appear in a series of adjacent frames. Over the timeline, the same object has self-similarity in both appearance and form. Thus, it is reasonable to assume that the information obtained from the temporal domain and spatial domain can aggregate to enhance the change detection results, e.g., with more accurate foreground contours. Motivated by the above mentioned, in this paper we propose a Fast Spatiotemporal Tree Filter (FSTF) to enhance coarse binary detection masks. The contributions are summarized as follows:

1) We extend the original tree filter to the spatiotemporal domain and propose Fast Spatiotemporal Tree Filter (FSTF) to enhance coarse change detection masks for image sequences. FSTF effectively synthesizes information of the same target from the current time and history frames to enhance detection. The proposed filter is purely unsupervised, and can be applied after various kinds of change detection algorithms.

2) We analyze the computational complexity of FSTF from the perspective of graph theory, and we theoretically prove that the fast realization of FSTF holds the same complexity with the original tree filter proposed in [39], which is a linear time algorithm.

3) Comprehensive qualitative and quantitative experiments prove that FSTF-based change detection enhancement is superior to several other state-of-the-art enhancing methods. We also demonstrate that FSTF has a broad applicability. It is not only applicable to enhancing change detection results, but

also able to improve saliency detection as well as semantic image segmentation on a single image.

The paper is organized as follows. The principles of linear image filtering and tree filtering are briefly introduced in Section II. Section III presents the framework of the foreground enhancement algorithm based on FSTF in details. In Section IV, comprehensive qualitative and quantitative experiments are performed to prove the superiority of FSTF. In section V, we discuss the robustness of FSTF against disturbances, and show the versatility of FSTF for improving different detections and segmentations. Finally, conclusions are drawn in Section VI.

II. Spatiotemporal Tree Filtering

In this section, we first elaborate the basic concept of using an image filter to enhance change detection results. Afterwards, we introduce the form of linear image filtering and several popular filters that belong to this form. We then extend the original tree filter that works spatially to a spatiotemporal domain, and propose an $O(n)$ Fast Spatiotemporal Tree Filter.

A. Change Detection and Image Filtering

An image filtering process involves an input image p, a guidance image I used for aggregating information from other domains, and an output image q. A change detection or a foreground detection result is usually represented by a binary image in which the foreground is labeled with white pixels, while the background is labeled in black. The objective of enhancing the change detection is to improve the input binary image p. The guidance image I, e.g., the grayscale original image or the depth image corresponding to p, contains rich texture information and color/spatial smoothness coherence that p does not hold. Moreover, due to noise and detection limits of those algorithms, dispersed false detections are common in the resultant binary image and the foreground mask may not cover the real object area. Therefore, it is natural to employ I as the reference to produce a filtered detection image q for enhancement. It should be noted that after enhancing the input binary image p, the output image q becomes a grayscale image in which a brighter pixel implies a higher possibility of being foreground.

B. Linear Image Filters

The general linear filtering process can be defined as a weighted sum on a pixel support region centered at pixel i

$$q_i = \sum\nolimits_j \omega_{ij}(I) p_j, \tag{1}$$

where j is the pixel index in the support region, and $\omega_{ij}(I)$ is the weight which can be regarded as the coherence between the center pixel i and j on the guidance image.

Joint bilateral filter [40], a very typical image filter in the above linear weighted form, can be represented as a joint filtering process upon the spatial and the color kernels

$$q_i = \frac{1}{N_i} \sum\nolimits_j \exp\left(-\frac{\left\|\boldsymbol{x}_i - \boldsymbol{x}_j\right\|^2}{\sigma_s^2}\right) \exp\left(-\frac{\left\|I_i - I_j\right\|^2}{\sigma_r^2}\right) p_j, \tag{2}$$

where $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ are the pixel coordinates, σ_s and σ_r are two parameters used to adjust the spatial similarity and the range similarity respectively, and N_i is a normalizing factor. If the guidance image is the input image itself, the joint bilateral filter degrades to the original bilateral filter proposed in [34].

The guided image filter [38], another effective local-based filter is also in the form of (1). In guided filtering, the kernel weight can be expressed as

$$\omega_{ij}(I) = \frac{1}{|\omega|^2} \sum_{k:(i,j)\in\omega_k} \left[1 + \frac{(I_i - \mu_k)(I_j - \mu_k)}{\sigma_k^2 + \varepsilon}\right], \tag{3}$$

where $|\omega|$ is the number of pixels in the support window ω_k with a radius r, and ε is a regularization parameter to control the edge-preserving extent. Parameters μ_k and σ_k^2 are the mean and variance of I in ω_k, respectively. There is no need to normalize the weights because $\sum_j \omega_{ij}(I) = 1$. A correspondence exists between the guided filter and the bilateral filter: $r \leftrightarrow \sigma_s^2$ and $\varepsilon \leftrightarrow \sigma_r^2$ [38].

C. Tree Filtering

Many linear image filters can be applied to improve change detection results; however, only in a local manner. The tree filter is a global filter because it aggregates information from all pixels along branches of the Minimum Spanning Tree of the guidance image.

In tree filtering, a guidance image I is regarded as a 4-connected, undirected graph $G = (V, E)$. The vertex set V represents all the pixels in the guidance image and the edge set E contains all edges between connected pixels. The weight of an edge connecting two pixels u and v can be represented as

$$e(u, v) = e(v, u) = |I_u - I_v|. \tag{4}$$

A minimum spanning tree (MST), which connects all the nodes and possesses the smallest sum of weights in all possible spanning trees, is generated from the graph G defined above. The similarity between any two nodes u and v on the guidance image is defined as

$$S_I(u, v) = S_I(v, u) = \exp\left(-\frac{L_I(u, v)}{\sigma}\right), \tag{5}$$

where $L_I(u, v)$ denotes the length between u and v—i.e., the sum of the weights of the edges on the path from u and v on the MST. The parameter σ adjusts the sensitivity of the similarity between u and v. For the guidance image I, the input p and the output image q, the tree filtering is defined as

$$q_i = \frac{\sum_j S_I(i, j)\, p_j}{\sum_j S_I(i, j)}. \tag{6}$$

The denominator of (6) depends only on the position of pixel i, which can then be viewed as a normalization factor N_i. The similarity S_I in (6) can also be viewed as a support weight received by the pixel i in filtering. It is easy to observe that the tree filtering has the weighted linear form just as (1).

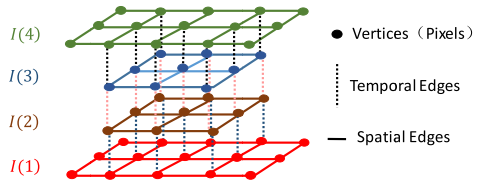


Fig. 1. Spatiotemporally connected, undirected graph $\mathcal{I} = (V, E)$, with $k = 4$.

D. Spatiotemporal Tree Filtering

The same moving object in several adjacent video frames have similar appearance, therefore it seems reasonable to aggregate the detection masks and guidance images of the same object with time. Sometimes due to restrictions on imaging, detection, as well as interference from background, the object in a frame can only be partially detected. By utilizing the spatiotemporally aggregated image filtering of the object, we can improve the current binary mask by exploiting past detections, even though they are coarse and imperfect.

Following the above enlightenment, a spatiotemporal tree filter builds a guidance image set $\mathcal{I}$ consisting of local image regions that contain the same object in k consecutive video frames. If $I(t)$ denotes the local image area of the moving object in the video frame at time t, then we have $\mathcal{I} = \{I(t)|t \in \{1, \ldots, k\}\}$. To balance the processing speed and the filtering effect, k should be set to a proper value (usually smaller than 5). The tuning of k will be further discussed in Section IV-B. By stacking all guidance images in $\mathcal{I}$ with their centers aligned together, we can construct a spatiotemporal undirected graph $\mathcal{I} = (V, E)$ shown in Fig. 1. The vertex set V contains all pixels of $\mathcal{I}$, and E consists of two parts which are the spatial edges and the temporal edges. The spatial edges are indicated by solid lines in Fig. 1, and the temporal edges are indicated by the dashed lines. All edges are defined exactly the same as (4). In order to effectively connect all images of the same target, we align the central pixels of all $I(t)$ for k times, and the latest image layers are placed on top of the stack. The pixels of target image $I(t)$ are connected to pixels from $I(t+1)$ and $I(t-1)$. As all edge weights are calculated using (4), a spatiotemporal minimum spanning tree (MST) can always be obtained from $\mathcal{I} = (V, E)$. The coarse detection masks corresponding to the guidance images can also form a set $\mathcal{P} = \{p(t)|t \in \{1, \ldots, k\}\}$, which is called the input set. Via the spatiotemporal tree filtering of (7), we acquire an enhancement foreground image set $\mathcal{Q} = \{q(t)|t \in \{1, \ldots, k\}\}$. The value of σ in (7) is related to the complexity of the video scene

$$\mathcal{Q}_i = \frac{\sum_j \exp\left(-\frac{L_{\mathcal{I}}(i,j)}{\sigma}\right)\mathcal{P}_j}{\sum_j \exp\left(-\frac{L_{\mathcal{I}}(i,j)}{\sigma}\right)}. \tag{7}$$

The biggest advantage of the spatiotemporal tree filter is that it can aggregate information in both time and space domains. In addition, compared with other local filters, it provides a natural measure of global pixel similarity; each pixel can effectively support all other pixels by delivering its information on the MST. Fig. 2(a) shows grayscale guidance images of

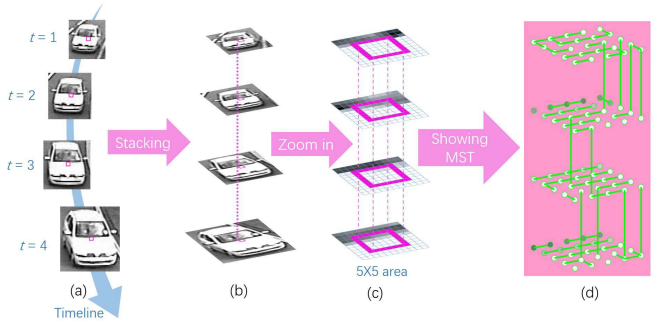


Fig. 2. Generating the spatiotemporal MST from the guidance images of the same foreground object in different frames. (a) shows the grayscale images of the same target in four different times; (b) is the guidance set $\mathcal{I}$ formed by stacking the images in (a); (c) is an enlarged visualization of the stacked 5∗5 pixel area near the center of $\mathcal{I}$. (d) shows the MST of the area of (c), in which the 4 horizontal layers are the 5∗5 pixel areas labeled in magenta from Fig. 2(a) to (c), respectively. In (d), the round nodes are painted with their real colors in the guidance images; and the green segments represent the edges in the generated MST of the whole graph $\mathcal{I}$. The edges that connect the nodes in (d) with the nodes outside of (d) are not shown here.

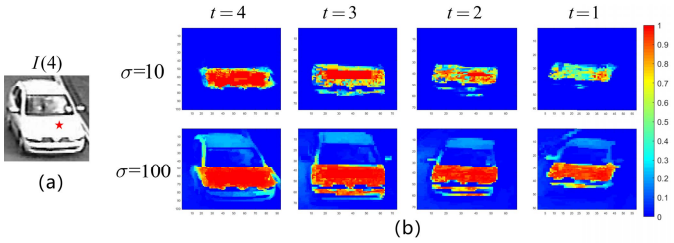


Fig. 3. Support weights computed by spatiotemporal tree filtering. (a) is the guidance image at $t = 4$ of Fig. 2(a) with a pixel marked in red star. (b) shows support weights received by the star pixel in heatmaps.

the same target at 4 different times, respectively. We stack the four guidance images together with their centers aligned in Fig. 2(b), and then compute the MST of this spatiotemporal volume. In order to exhibit details, we focus on a stacked 5∗5 pixel area labeled by magenta bounding boxes in Figs. 2(a), (b), and (c). We zoom in on the 5∗5 stacked area and show its part of MST in Fig. 2(d), in which we can see that pixels with similar gray values are connected to each other in spatiotemporal MST. Fig. 2(d) also reveals that similar and nearby pixels will deliver information in a fast and natural way. Therefore, the spatiotemporal structure of MST implies a "global" measure of pixel similarity. Fig. 3 uses heap maps to show the similarities of a pixel with all other pixels after information aggregated on the spatiotemporal MST, which also reflects the global supports of other pixels for the pixel. The first line of Fig. 3(b) shows the similarities between the pixel marked by the five-pointed star in Fig. 3(a) and all other pixels in $\mathcal{I}$ at $\sigma = 10$. The second line of Fig. 3(b) shows the similarities when $\sigma = 100$. The values of all similarities are normalized to the interval of [0, 1], and the similarities are illustrated by heatmaps with red color for high values and blue for the opposite. At all four guidance frames, those pixels that have similar grayscale intensity to the five-pointed star pixel are close to red, indicating a high support in filtering. The parameter σ controls the sensitivity of the similarity. When σ is small, only the pixels from the car hood can effectively support the star pixel; but when σ is large, the information becomes easier to transmit on the MST, causing supports from

other parts of the car (e.g., car roof, and body) that have similar color with the hood to become larger. As a result, the shapes of the high similarity areas in Fig. 3(b) at $\sigma = 100$ get closer to a real car.

E. Fast Spatiotemporal Tree Filtering (FSTF)

According to (7), when computing the filter output of pixel i we must calculate the similarity between it and all other pixels in $\mathcal{I}$. Furthermore, the computation complexity of $L_I(i, j)$ is related to the distance between j and i on the MST (the number of edges on the path connecting i and j). Therefore, in the light of (7), the floating-point addition need to be operated $\sum_{j=1}^{n} d(i, j)+2(n-1)$ times, and multiplication need to be operated $2n$ times, exp is calculated n times, and division once. $\sum_{j=1}^{n} d(i, j)$ is the sum of all distances from any node to the current node i; it cannot be easily computed. However, there is a clue that $\sum_{j=1}^{n} d(i, j)$ can be approximated by the average distance d_{ave} of the tree multiplied by n. The average distance of a graph is defined as the expected distance between a randomly chosen pair of distinct vertices. If the graph is a tree, then the average distance can be represented by [41]:

$$d_{ave} = \frac{1}{\binom{n}{2}} \sum_{u,v \subset \text{tree}} d\,(u, v). \tag{8}$$

When the graph is large, it becomes very costly to calculate the exact average distance via (8). The study of the average distance as a graph parameter began from 1940s [42]. Despite many theorems were proposed for the average distance of a connected graph, the majority of them focus on derivations of the upper or/and lower bounds. Mohar [43] shows an exact connection between eigenvalues and mean distance in graphs; given a tree of order n and let $\lambda_2, \lambda_3, \ldots, \lambda_n$ be the non-zero Laplacian eigenvalues of this tree, then

$$d_{ave} = \frac{2}{n-1} \sum_{i=2}^{n} \frac{1}{\lambda_i}. \tag{9}$$

This equation gives a closed-form solution for computing the average distance in a tree. According to the definition of the proposed spatiotemporal filter, the guidance set has a spatial structure that resembles to a thin plate. This is because the dimension of a guidance image is far larger than k, the thickness of the set. We have calculated the average distance using (9) on more than 200 different MSTs formed on real guidance sets with $k \leq 4$ from image patches such as Fig. 2., respectively; and discovered that d_{ave} always falls into an interval of $(n^{1/2}, 3n^{1/2})$. Therefore $\sum_{j=1}^{n} d(i, j)$ can be upper bounded by $3n^{3/2}$. Direct computation of (7) on all pixels in the guidance set $\mathcal{I}$ has a combined complexity of $O(n^{5/2})$ on floating point additions, $O(n^2)$ on both floating point multiplications and exponentials, and $O(n)$ for divisions. According to [44], the floating point addition takes 3-6 CPU clock cycles, the floating point multiplication takes 4-8 cycles, a division takes 20-45 cycles, and an exponential takes no less than 48 cycles. The floating-point additions account for the majority of the cost in direct computation of (7), and it is pretty slow especially when the spatiotemporal tree contains thousands of vertices.

The mechanism can be sped up by taking the advantage of both the tree structure and definition of the similarity by (5). Assume a simple case that the vertices u and v are connected by a vertex w on a path of the MST; then the similarity between u and v can be represented by a multiplication of similarity $S_I(u, w)$ and $S_I(w, v)$. Actually, the path that connects any two nodes on the MST is unique, otherwise two different paths connecting the same pair of nodes will form a circle, which contradicts with the fact that the tree is already an MST. If the path connecting the two nodes u and v to be represented by an edge set $E_{u,v}=\{e_1, e_2, \ldots, e_d\}$, then the length between u and v can be calculated by

$$L_{\mathcal{I}}(u, v) = \sum_{e_i \in E_{u,v}} e_i. \tag{10}$$

By using equation (5), the similarity of u and v becomes

$$S_{\mathcal{I}}(u, v) = \exp\left(-\sum_{e_i \in E_{u,v}} e_i/\sigma\right) = \prod_{e_i \in E_{u,v}} \exp\left(-\frac{e_i}{\sigma}\right). \tag{11}$$

If the guidance images are grayscale, the edge weight e_i is an integer ranging from 0 to 255, and the parameter σ is preset before computation. The exponentials in (11) can then be precomputed by a table of length 256, and the similarity between two arbitrary nodes now breaks into a multiplication of precomputed float numbers. Now the total computation complexity using the above speed-up for all pixels in $\mathcal{I}$ comprises $O(n^2)$ on floating-point additions, $O(n^{5/2})$ on floating point multiplications, and $O(n)$ for divisions. When the number of pixels in $\mathcal{I}$ is not large, the speedup by (11) is evident. However, applying (11) to a bigger guidance set costs even more time than the direct computation of (7) because a CPU normally needs slightly more cycles on doing multiplication than addition.

The key to this speed-up problem is to treat (7) as an information aggregation process on the MST beginning from leaf nodes to the root node, i.e., the current pixel i. This aggregation is therefore called the leaf-to-root process. The numerator and denominator of (7) are aggregated separately using the Claim 1 presented in [39]. The aggregated value for each node u is represented by $\mathcal{Q}_u^A = \mathcal{Q}_u^A(num)/\mathcal{Q}_u^A(den)$. The numerator can be computed by

$$\mathcal{Q}_u^A(num) = \mathcal{P}_u + \sum_{par(v)=u} S_{\mathcal{I}}\,(u, v)\,\mathcal{Q}_v^A(num), \tag{12}$$

in which $par(v)$ means the parent node of v. The denominator is then computed by

$$\mathcal{Q}_u^A(den) = 1 + \sum_{par(v)=u} S_{\mathcal{I}}\,(u, v)\,\mathcal{Q}_v^A(den). \tag{13}$$

For each leaf node, we have $\mathcal{Q}_{leaf}^A(num) = \mathcal{P}_{leaf}^A(num)$ and $\mathcal{Q}_{leaf}^A(den) = 1$. For the root node only, the equation $\mathcal{Q}_i = \mathcal{Q}_i^A$ holds. The computation complexity can be easily derived by

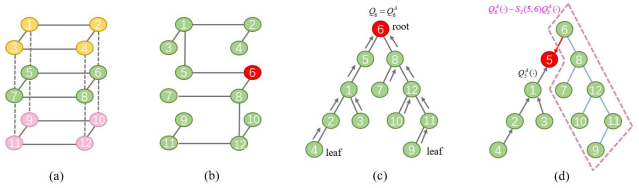


Fig. 4. A demonstration of fast spatiotemporal tree filtering. (a) shows a guidance graph $\mathcal{I}$ containing three layers of images from three different epochs, respectively. The spatial edges are solid lines, and the temporal edges are in dotted lines. (b) shows the MST computed from the graph (a). Fig. 4(c) shows the leaf-to-root information aggregation of the MST with node 6 as the root node; the arrows represent the direction of aggregation. (d) shows the computation of the node 5 using the root-to-leaf propagation. The filtering result of node 5 is a combination of two parts: (i) the aggregated information $\mathcal{Q}_5^A(\cdot)$ from the subtree that contains node 5 itself in the leaf-to-root process; and (ii) the part surrounded in dotted red area that propagated from the whole MST except the subtree containing node 5. In (d), the black arrows represent the aggregation process, and the red arrow represents the propagation.

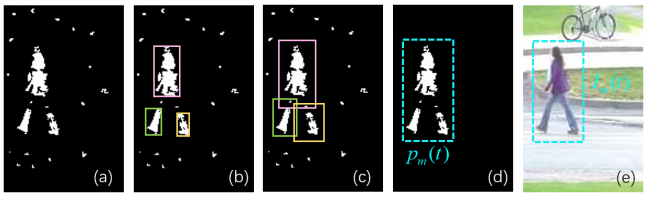


Fig. 5. Merging the fragmented foregrounds and removing false positives for generating intact foreground guidance images. (a) is the binary detection result of a foreground detection method; In (b), the foreground areas that contain enough white pixels are labeled by bounding boxes, respectively. Each bounding box in (b) is broadened in (c) by a few pixels to avoid missing false negatives. The overlapping foreground bounding boxes are bounded by a larger bounding box (dotted blue box) in (d), and the foreground area in this blue box is treated as the input of tree filtering. The corresponding bounding box area in the guidance image (e) is used to generate the guidance set for spatiotemporal tree filtering.

a simple example shown by Fig. 4. Fig. 4(a) is a spatiotemporal graph $\mathcal{I}$ with $k = 3$, and the corresponding MST is demonstrated by solid connections in Fig. 4(b). Assuming we are now going to compute the filtered value for node 6, the MST can immediately turn into an equivalent leaf-to-root structure shown in Fig. 4(c). The information aggregates along the arrows on the tree to the root node. The equations (12), (13) are computed sequentially along the arrows to the root, and it is therefore straightforward to find that either (12) or (13) takes one floating-point multiplication as well as one addition on each edge. So, each root node costs $2(n-1)$ multiplications, $2(n-1)$ additions, and 1 division to compute its filter output. After iterating the root on all nodes in the MST, the total complexity reduces to $O(n^2)$ on CPU clock cycles. Though faster than the speed-up using (11), the calculation is still formidable for large graphs.

The leaf-to-root aggregation spends a lot of repetitive calculations when the root node iterates from one node to another. In fact, it is possible to compute the aggregation values $\mathcal{Q}_u^A(num)$ and $\mathcal{Q}_u^A(den)$ only once for all nodes, and then compute all filter outputs $\mathcal{Q}_i$ without changing the root node on the graph. First, we randomly choose a root node in the graph, then do leaf-to-root aggregation, and the aggregated numerators and denominators for all nodes are stored. By applying the Claim 2 in [39], a root-to-leaf propagation process is used to compute the final filtering result for numerators of all other nodes in the set using (14):

$$\begin{aligned}\mathcal{Q}_u(num) = {} & \mathcal{Q}_u^A(num) \\ & + S_{\mathcal{I}}(par(u), u)\mathcal{Q}_{par(u)}(num) \\ & - S_{\mathcal{I}}(u, par(u)) \cdot \mathcal{Q}_u^A(num). \end{aligned} \tag{14}$$

If change all "num" in (14) to "den", we obtain the propagation for denominators. Equation (14) must be computed sequentially starting from the root node to its child nodes, and finally to all leaf nodes. In the example of Fig. 4(c), since the output $\mathcal{Q}_6$ of the root node 6 has been computed, we can propagate the support from node 6 to its child node 5. Then the final filtering result of node 5 is a combination of two parts: (i) $\mathcal{Q}_5^A(\cdot)$, the information aggregated from the subtree that contains node 5 itself in the leaf-to-root process; (ii) $\mathcal{Q}_6(\cdot) - S_{\mathcal{I}}(5,6)\mathcal{Q}_5^A(\cdot)$, the information that propagated from the whole MST except the subtree containing node 5. The first part can be represented by the black arrows along edges in Fig. 4(d), and the second part can be represented by the area surrounded by the dotted red box. When the second part propagates from the root node 6 to the node 5, the information has to go through the edge $e(6,5)$; therefore it must be multiplied by the similarity $S_{\mathcal{I}}(6,5)$ as the attenuation. After the numerator and denominator of node 5 are computed separately from (14), the final $\mathcal{Q}_5$ can be obtained via a division. Equation (14) can be further transformed into

$$\mathcal{Q}_u(\cdot) = S_{\mathcal{I}}\left(par(u), u\right)\mathcal{Q}_{par(u)}(\cdot) + \left[1 - S_{\mathcal{I}}^2\left(u, par(u)\right)\right]\mathcal{Q}_u^A(\cdot), \tag{15}$$

in which the notation $(\cdot)$ means either numerator or denominator. The two terms and $1 - S_{\mathcal{I}}^2$ in (15) can be precomputed by look-up tables, and $\mathcal{Q}_u^A(\cdot)$ is precomputed in the leaf-to-root process. Therefore, each non-root node needs 2 multiplications, 1 addition, and 1 division to compute the output. As a result, the total MST needs 1 leaf-to-root aggregation and $n-1$ root-to-leaf propagations. Comparing to the original complexity $O(n^{5/2})$ on computing (7) for all pixels, now the total complexity of the speed-up only takes $O(n)$ on CPU clock cycles, which is linear with the size of the MST. This speed-up makes real-time FSTF applications to be possible.

III. FSTF-Based Change Detection Enhancement

In this section, we propose a foreground detection enhancement method based on FSTF. The framework can be divided into three parts: (i) fragmented foreground parts merging and noise removal, (ii) enhancement of FSTF, and (iii) binarization.

(i) Fragmented foreground parts merging and noise removal. For the k-frame consecutive foreground masks obtained by change detection, we use 8-connected component labeling to count the number of pixels for all foreground regions in those foreground masks. If the number of pixels of a connected foreground region is larger than Th_{area}, we regard the region as a potential foreground area and then label it with a bounding box. For better handling of under-detection, we enlarge all object boxes with 10-20 pixels in height and

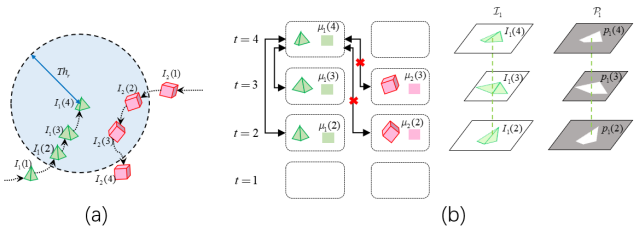


Fig. 6. The sketch map of forming the guidance image set and the coarse detection image set for a target in four consecutive frames for FSTF calculation. (a) shows the trajectories of the two targets in the video sequence and the searching area for target 1. (b) shows the process of forming sets $\mathcal{I}_1$ and $\mathcal{P}_1$ for target 1.

width. Then, the overlapping bounding boxes in the same image are bounded by a larger box that merges the fragments. Fig. 5(a) is the coarse binary result obtained by using a foreground detection algorithm on the original frame Fig. 5(e). The three small boxes in Fig. 5(b) are the bounding boxes of connected foregrounds that exceeds the size threshold Th_{area}. In Fig. 5(c), the three boxes are extended a little to prevent from missing details in case of under-detection. Because the three boxes have overlapping areas, in Fig.5(d) we use a larger dotted box to encompass all of them. The binary image area inside the box is denoted by $p_m(t)$, and it means the m-th foreground area at time t, in which $m \in \{1, \ldots, m_{\max}(t)\}$. The corresponding area of $p_m(t)$ in the original frame is defined by $I_m(t)$, which is shown in Fig. 5(e). $p_m(t)$ and $I_m(t)$ will directly participate in FSTF computation.

There is usually a large amount of white isolated noise in a coarse binary mask. If the number of pixels of a connected foreground area is less than Th_{area} and it is also not contained in any of bounding boxes, we consider it a false positive area. The false positive pixels are then directly changed to black pixels as a denoising step for removing dispersed detection noise. Following this denoising, the false positive areas in Fig. 5(d) are reduced comparing to Fig. 5(c), while the small white areas inside $p_m(t)$ remains unchanged (e.g., two small white points in the thighs).

(ii) Enhancement of FSTF. First calculate the mean intensity value $\mu_m(t)$ of the foreground pixels of $I_m(k)$, and compute the center coordinate of $I_m(k)$ as $c_m(t)$ for all foreground areas in all frames $t \in \{1, \ldots, k\}$. We establish a guidance graph $\mathcal{I}_m = I_m(k)$ for the mth foreground region at the latest time k, and establish the coarse input set $\mathcal{P}_m = p_m(k)$. At each $c_m(t)$ position with a radius of Th_r, we search whether it exists a center $c(t)$ of another foreground region falling into the circle region. If multiple foreground areas fall into the search region in a same frame, only the foreground that has the closest mean intensity to $\mu_m(k)$ is selected as the potential matching candidate. Then we stack the guidance images of the candidate foreground beneath the guidance graph $\mathcal{I}_m$, and then add the corresponding binary image area to the set $\mathcal{P}_m$. If no foreground area is found in a frame, two sets—$\mathcal{I}_m$ and $\mathcal{P}_m$ stay unchanged. After traversing all frames, the FSTF is applied on $\mathcal{I}_m$, $\mathcal{P}_m$ to obtain the output set $\mathcal{Q}_m = \{q_m(t) | t \in \{1, \ldots, k\}, m \in \{1, \ldots, m_{\max}(t)\}\}$. Fig. 6 shows how the guidance set and the coarse input set of a target are formed in four consecutive frames. Fig. 6(a) presents the trajectories of the two moving targets in the video sequence. Let $c_1(4)$ be the center of target 1 at frame 4, we search within a radius of Th_r for target regions in the spatiotemporal domain. In Fig. 6(b) we can see at $t = 3$, $c_1(3)$ and $c_2(3)$ both satisfy the distance condition. But $\mu_1(3)$ is much more close to $\mu_1(4)$ than $\mu_2(3)$, so $I_1(3)$ is the potential matching candidate of $I_1(4)$. Then we stack $I_1(3)$ beneath $I_1(4)$ to update $\mathcal{I}_1$, and meanwhile add $p_1(3)$ to $\mathcal{P}_1$. Similarly, at time $t = 2$, only $I_1(2)$ is regarded as the potential matching candidate of $I_1(4)$, and $\mathcal{I}_1$, $\mathcal{P}_1$ are also updated, respectively. At time $t = 1$, no foreground can be found in the search circle, therefore the two sets remain the same. After traversing all four frames, FSTF is applied to $\mathcal{I}_1$ and $\mathcal{P}_1$ to generate a resultant set $\mathcal{Q}_1 = \{q_1(4), q_1(3), q_1(2)\}$, which contains enhanced grayscale detection masks for target 1 at all three frames.

(iii) Binarization. Each pixel in the spatiotemporal output set $\mathcal{Q}_m$ is grayscale, so a threshold Th_b is needed to binarize $\mathcal{Q}_m$. If the filtered pixel value is greater than the threshold, it becomes a white foreground pixel; otherwise it becomes a black background pixel. After binarization, we obtain an enhanced binary mask $b_m(k)$ for $q_m(k)$. Finally, the region of $b_m(k)$ is used to cover the corresponding rectangular area $p_m(k)$ in the binary detection frame after noise removal to

TABLE I

THE PSEUDO-CODE OF THE CHANGE DETECTION ENHANCEMENT ALGORITHM BASED ON FSTF

Algorithm 1 Change detection enhancement based on FSTF

Input: The k consecutive frames and their corresponding binary detection images

Parameters: σ, Th_{area}, Th_t, and Th_b.

Output: The enhanced k consecutive binary images.

(i) Fragmented foreground parts merging. For all binary input images, all the foreground areas $p_m(t)$ and guidance areas $I_m(t)$ are obtained after merging. We have $t \in \{1,...,k\}$, $m \in \{1,...,m_{\max}(t)\}$.
Noise removal: For all input binary images, we turn all white connected areas that are outside of all $p_m(t)$ and meanwhile have less pixels than Th_{area} into black.

(ii) Enhancement of Fast Spatiotemporal Tree Filtering:

For each foreground area $p_m(k)$ **do**
  Set $\mathcal{I}_m = I_m(k)$, and $\mathcal{P}_m = p_m(k)$.
  For each frame time $t \in \{1,...,k-1\}$ **do**
    $temp\_\mu \leftarrow +\infty$, and $temp\_i \leftarrow 0$;
    For $i \in \{1,..,m_{\max}(t)\}$ foreground area $p_i(t)$ **do**
      If $\|c_m(k) - c_i(t)\| < Th_r$ && $|\mu_m(k) - \mu_i(t)| < temp\_\mu$
        $temp\_\mu \leftarrow \mu_i(t)$,
        $temp\_i \leftarrow i$;
      End if
    End for
    If $temp\_i > 0$
      Push $p_{temp\_i}(t)$ into $\mathcal{P}_m$;
      and stack $I_{temp\_i}(t)$ under $\mathcal{I}_m$.
    End if
  End for
  Use $\mathcal{I}_m$, and $\mathcal{P}_m$ to carry out FSTF with parameter σ, and output $\mathcal{Q}_m$
End for

(iii) Binarization:

For each filtered spatiotemporal set $\mathcal{Q}_m$ **do**
  Use threshold Th_b to binarize $q_m(k)$ to output $b_m(k)$, and then substitute foreground areas of $p_m(k)$ with their corresponding foreground areas of $b_m(k)$, respectively.
End for

form the final enhancement result. The pseudo-code for change detection enhancement based on FSTF is shown in Table I.

IV. Experiments

In this section, in order to demonstrate the superiority of the proposed FSTF, we first compare it with the other six algorithms as the first part of our experiments: morphological operators (MO), Foreground-adaptive MRF (FA MRF) [28], Li MRF [13], guided filtering (GF) [38], joint bilateral filtering (JBF) [34], and fully-connected Conditional Random Field (CRF) [31] on enhancing the coarse detection masks obtained from 4 different foreground detection algorithms: GMM [11], SpkmeansEM [13], ViBe [15], and FgSegNet [20], respectively. The comparison is performed on 5 video sequences from the CDnet2012 dataset [22]: "streetlight", "pedestrians", "office", "traffic", and "diningRoom". The "streetlight" is a 320*240 resolution sequence that captures an outdoor traffic scene using a fixed camera. The "traffic" sequence is also a 320*240 resolution sequence that records a near traffic scene with a fixed camera. The "office" is a 360*240 resolution sequence that records an office scene. The "pedestrians" is also a 360*240 resolution sequence that records a street scenario containing several walking pedestrians with subtle background motion. The "diningRoom" is a 320*240 resolution sequence recorded by a thermal camera. CDnet2012 offers ground truth for all sequences; but the region of interest (ROI) of "streetlight" only covers a very small region. Therefore, we manually label the ground truth images for this sequence. GMM and SpkmeansEM are statistical background-foreground subtraction based on mixture models that need unsupervised training. We use the first 600 frames of "streetlight", the first 400 frames of "pedestrians", the first 500 frames of "office", "traffic", and "diningRoom", respectively, to train background models for the two statistical algorithms. We adopted suggested learning rates in their original papers, and the maximum number of the Gaussian distributions is fixed to 5 at each pixel for both GMM and SpkmeansEM. The only parameter T in the foreground detection of GMM is fixed to 0.5 for all videos. There are four parameters in the foreground detection stage of SpkmeansEM, and we tune the parameters according to the suggestion in the original paper. ViBe is an unsupervised heuristic algorithm that uses a history pixel batch to represent background; four parameters are designed for updating the background batch as well as detecting moving foregrounds, and we use the default values recommended in its original paper. FgSegNet is a supervised method that uses convolutional neural networks for multiscale feature encoding. We choose 200 images from several different CDnet sequences containing human foregrounds as the FgSegNet training data to segment human targets, and choose 80 images from several sequences containing vehicles to train the FgSegNet for detecting vehicles. The enhancement methods are tested and compared on the coarse detection results on frames 704-713 of "streetlight", frames 1291-1303 of "traffic", frames 734-743 of "office", frames 576-585 of "pedestrians", and frames 833-842 of "diningRoom".

The seven detection enhancement methods including ours are tuned to produce their respective optimal results in experiments. The morphological operator comprises a fundamental "open" operation to reduce noise and a "close" operation to enhance foreground contours. The FA MRF and Li MRF methods are both Maximum *a posteriori* (MAP) MRF, which requires both the *a priori* background probability and the *a posteriori* foreground probability of every pixel; thus the two MRF methods are not suitable for non-statistical methods such as ViBe and FgSegNet. The Gibbs temperature γ and a scale parameter θ of the FA MRF are both fixed at 0.01, and the number of MRF iterations is no higher than 3 to avoid over-smoothing. The parameters for the Li MRF are the values recommended in the paper [13], and the number of iterations is also no higher than 3. The joint bilateral filter is controlled by four parameters, in which the spatial parameter σ_s, the range parameter σ_r, and the window radius r are fixed to 150, 0.2, and 5 for all video sequences, respectively. The binarization threshold Th_b for joint bilateral filtering is set to a range of 120-150 for FgSegNet results, and 50-80 for other results. The parameters r and ε for guided filtering are fixed to 4 and 0.16, respectively. The binarization threshold is set the same way as the joint bilateral filtering. For FSTF, the parameter σ is set to 25 for "streetlight" and "traffic" sequences, 10 for "office", 15 for both "pedestrians" and "diningRoom". The threshold Th_b is set to 100 for FgSegNet results and 20 for the other 3 methods. We fix parameters Th_{area} and Th_r to 50, and fix k to 4 for all experiments in this paper. The fully connected CRF contains 5 parameters and we use recommended values from [31] to realize the enhancement. CRF requires pixel-wise foreground probability map, thus it is not suitable for ViBe.

The second part of experiments focus on quantitative comparisons. In this part, we first compare the performances of the seven enhancement methods on consecutive frames from five CDnet2012 sequences. Then, we apply the proposed FSTF after five extra popular change detection methods on the complete CDnet2012 dataset to prove its wide applicability. At last, we compare FSTF with GF, CRF, and Li MRF under three metrics for improving coarse SpkmeansEM results.

Following the comparisons, we also provide with a number of parameter tuning tests. Suggestions are given on how to choose proper values for σ and Th_b based on different conditions. The stability test of the performance of FSTF under varying parameters is also performed. The influence of parameter k on FSTF is analyzed. At last, the average time costs for the seven enhancement algorithms on five CDnet sequences are compared. Our FSTF has satisfactory real-time speed.

All experiments are conducted on a desktop PC operated under Windows 10 with a 3.4 GHz Intel Core i7-3770 CPU and 16 GB memory. In this section, the quantitative evaluations are performed by using metrics such as F-measure, Mean Absolute Error (MAE), Mean Square Error (MSE), Peak Signal to Noise Ratio (PSNR), and the Receiver Operating Characteristics (ROC) curve. F-measure is defined by

$$F\text{-}measure = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{16}$$

where Precision = TPs/(TPs + FPs) and Recall = TPs/(TPs + FNs). True Positive (TP) is the foreground pixel in the ground truth correctly classified as the foreground by the algorithm. False Positive (FP) is the background pixel in

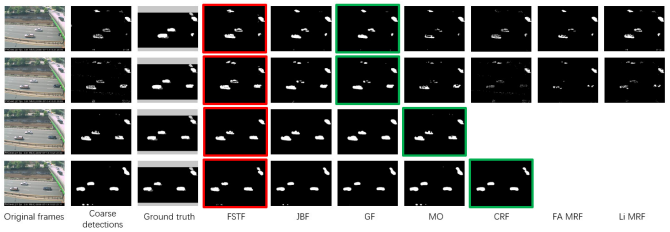


Fig. 7. The comparison of enhancements across seven methods on coarse detection results of the "streetlight" sequence. In order to avoid visual redundancy, we compare enhancements on GMM for frame 703 on row 1, enhancements on SpkmeansEM for frame 707 on row 2, enhancements on ViBe for frame 710 on row 3, and enhancements on FgSegNet for frame 713 on row 4. On each row, the enhancement result with the highest F-measure is highlighted by a red bounding box, the result with the second highest F-measure is labeled by a green bounding box.

the ground truth falsely classified as the foreground by the algorithm. False Negative (FN) is the foreground pixel in the ground truth incorrectly classified as the background. TN is the background pixel in the ground truth correctly classified as the background by the algorithm. The ROC curve reflects the relation between False Positive Rate (FPR) and True Positive Rate (TPR), in which FPR=FPs/(FPs+TNs) and TPR=Recall. MAE, MSE, and PSNR are defined by (17)-(19), in which GT_i means the i-th pixel of the binary ground truth image, and $MAX(b)$ means the maximum value of a pixel in the output binary image it can reach. In quantitative evaluations, we normalize all binary values into the range of [0, 1]. Then $b_i - GT_i$ has only three cases $\{-1, 0, 1\}$, and the MAE and MSE are the same all the time. Therefore, we integrate the two measures with a label "MAE/MSE".

$$MAE = \frac{1}{|b|}\sum_{i=1}^{|b|}|b_i - GT_i|, \tag{17}$$

$$MSE = \frac{1}{|b|}\sum_{i=1}^{|b|}(b_i - GT_i)^2, \tag{18}$$

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX^2(b)}{MSE}\right). \tag{19}$$

A. Qualitative and Quantitative Comparisons

The enhancement methods are first compared on the "streetlight" images in Fig. 7. In order to avoid visual redundancy, we show enhancements on different foreground detection algorithms on different frames, respectively. This is because we only focus on enhancing coarse detections from detection algorithms, rather than comparing different detection algorithms. Our FSTF generates the closest results to the ground truth across all seven enhancement methods in Fig. 7. In addition, FSTF is the only method that correctly separates two vehicles from a falsely connected foreground region on the upper-right corner of the FgSegNet mask. For the enhancement comparison on "traffic" sequence in Fig. 8, the proposed FSTF is the closest to the ground truth across all compared methods for GMM, ViBe, and FgSegNet. For SpkmeansEM, CRF narrowly wins over FSTF on the frame 1297 under SpkmeansEM. Particularly, the FSTF shows impressive improvement on FgSegNet, which evidently outperforms all other methods.

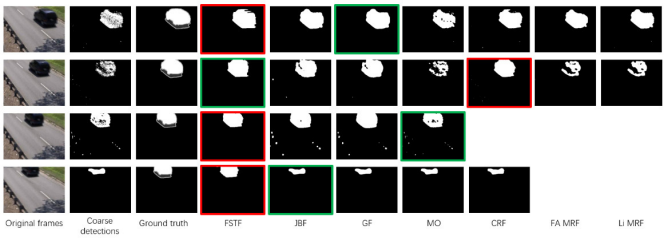


Fig. 8. The comparison of enhancements across seven methods on coarse detection results of the "traffic" sequence. We show enhancements on GMM coarse mask for frame 1294 on row 1, enhancements on SpkmeansEM for frame 1297 on row 2, enhancements on ViBe for frame 1300 on row 3, and enhancements on FgSegNet for frame 1303 on row 4. On each row, the result with the highest F-measure is labeled by a red bounding box, the result with the second highest F-measure is labeled by a green bounding box.

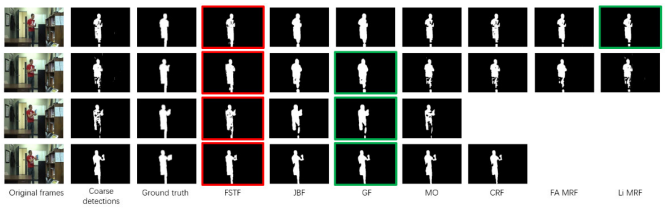


Fig. 9. The comparison of enhancements across seven methods on coarse detection results of the "office" sequence. We show enhancements on GMM coarse mask for frame 734 on row 1, enhancements on SpkmeansEM for frame 737 on row 2, enhancements on ViBe for frame 740 on row 3, and enhancements on FgSegNet for frame 743 on row 4.

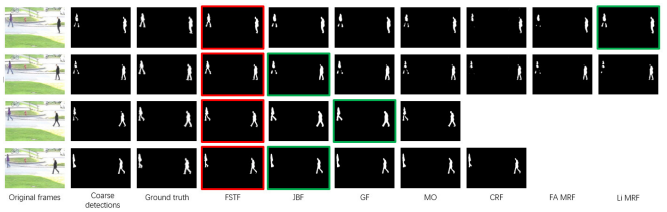


Fig. 10. The comparison of enhancements across seven methods on coarse detection results of the "pedestrians" sequence. We show enhancements on GMM coarse mask for frame 576 on row 1, enhancements on SpkmeansEM for frame 579 on row 2, enhancements on ViBe for frame 582 on row 3, and enhancements on FgSegNet for frame 585 on row 4.

Comparison on the "office" sequence is shown in Fig. 9. FSTF is still the method closest to the ground truth across all seven enhancement methods, and FSTF is the only method able to connect the leg area and the upper body in enhancements for GMM, SpkmeansEM, and ViBe. The JBF and GF methods show evident over-smoothing effects. For the enhancement comparison on "pedestrians" in Fig. 10, the proposed FSTF is still the closest to the ground truth across all seven enhancement methods, and FSTF is the only method capable of detecting the full contour of the left pedestrian on GMM and SpkmeansEM masks. FSTF is also the only method that can distinguish the two different feet of the left pedestrian from the over-smoothed FgSegNet result (row 4 of Fig. 10). For the enhancement comparison on "diningRoom" in Fig. 11, the proposed FSTF is still the closest to the ground truth across all enhancements, and it is the only method that completely fills the upper body area of the person with white pixels.

On each row of Figs. 7-11, the enhancement result with the highest F-measure is highlighted by a red bounding box, the

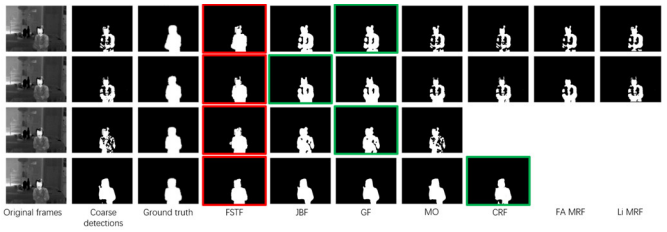


Fig. 11. The comparison of enhancements across seven methods on coarse detection results of the "diningRoom" sequence. We show enhancements on GMM coarse mask for frame 833 on row 1, enhancements on SpkmeansEM for frame 836 on row 2, enhancements on ViBe for frame 839 on row 3, and enhancements on FgSegNet for frame 842 on row 4.

TABLE II

QUANTITATIVE FSTF ENHANCEMENT RESULTS ON FIVE DIFFERENT CHANGE DETECTION ALGORITHMS FOR THE TOTAL CDNET2012 DATASET

Categories	Sequences	Detection methods									
		Histogram [45]		ED [46]		WeSamBE [47]		STBM [48]		SuBSENSE [49]	
		coarse	FSTF	coarse	FSTF	coarse	FSTF	coarse	FSTF	coarse	FSTF
baseline	highway	0.8809	**0.9002**	0.9053	**0.9636**	0.9723	**0.9762**	0.9792	**0.9804**	0.9750	**0.9813**
	office	0.9692	**0.9780**	0.8448	**0.8464**	0.9312	**0.9582**	0.9681	**0.9728**	0.9625	**0.9720**
	pedestrians	0.9562	**0.9657**	0.9623	**0.9687**	0.9646	0.9645	0.9682	0.9681	0.9625	**0.9630**
	PETS2006	0.8341	**0.8786**	0.8162	**0.8778**	0.9359	0.9352	0.8610	**0.8635**	0.9408	0.9399
cameraJitter	badminton	0.2907	**0.3910**	0.5999	**0.6473**	0.8305	**0.8478**	0.7739	**0.7967**	0.8814	**0.8946**
	boulevard	0.3907	**0.4364**	0.4552	**0.4869**	0.7157	**0.7888**	0.7425	**0.7636**	0.7528	**0.7899**
	sidewalk	0.1700	**0.1914**	0.3057	**0.3529**	0.8561	**0.8858**	0.8348	**0.9034**	0.8432	**0.8721**
	traffic	0.2739	**0.2657**	0.6445	**0.7180**	0.8833	**0.8970**	0.7346	**0.8159**	0.8807	**0.8975**
dynamicBackground	boats	0.2507	**0.7445**	0.7050	**0.8870**	0.6402	**0.6922**	0.8761	**0.9219**	0.6933	**0.7464**
	canoe	0.4741	**0.6513**	0.8239	**0.9060**	0.6131	**0.7070**	0.8349	**0.8886**	0.7923	**0.8692**
	fall	0.1478	**0.1701**	0.1800	**0.1975**	0.8193	**0.8482**	0.9152	**0.9255**	0.8716	**0.8923**
	fountain01	0.0248	**0.0325**	0.0394	**0.0460**	0.7329	**0.7495**	0.5061	**0.5850**	0.7531	**0.7617**
	fountain02	0.1598	**0.2260**	0.5089	**0.6020**	0.9434	**0.9434**	0.8809	**0.8909**	0.9441	**0.9448**
	overpass	0.4019	**0.5832**	0.7978	**0.8497**	0.7236	**0.7731**	0.9134	**0.9233**	0.8600	**0.8842**
intermittentObjectMotion	abandonedBox	0.5279	**0.6607**	0.6376	**0.7902**	0.9007	**0.9333**	0.3273	**0.3279**	0.8574	**0.8908**
	parking	0.4415	**0.5878**	0.5515	**0.6549**	0.8216	**0.8328**	0.9212	**0.9454**	0.4438	**0.5946**
	sofa	0.6638	**0.7036**	0.5927	**0.6041**	0.7591	**0.7715**	0.8417	**0.8469**	0.7424	**0.7515**
	streetLight	0.9293	**0.9711**	0.6833	**0.8572**	0.9911	**0.9912**	0.9610	**0.9714**	0.9883	**0.9884**
	tramstop	0.3725	0.3672	0.3155	**0.3488**	0.5652	**0.5934**	0.5411	0.5375	0.5559	**0.5904**
	winterDriveway	0.1374	0.1370	0.1616	0.1591	0.4161	**0.4554**	0.3073	**0.5214**	0.3698	**0.3991**
shadow	backdoor	0.4824	**0.4907**	0.5638	**0.5767**	0.9586	**0.9606**	0.8981	**0.9066**	0.9561	**0.9592**
	bungalows	0.8793	**0.9684**	0.8748	**0.9721**	0.9713	**0.9795**	0.9208	**0.9253**	0.9770	**0.9832**
	busStation	0.7990	**0.8698**	0.7646	**0.8099**	0.9067	**0.9076**	0.9094	**0.9115**	0.9104	**0.9116**
	copyMachine	0.8638	**0.8708**	0.7177	**0.7376**	0.9217	**0.9232**	0.9038	**0.9101**	0.9290	**0.9302**
	cubicle	0.2387	**0.2423**	0.4528	**0.4688**	0.9375	**0.9468**	0.8010	**0.8190**	0.9228	**0.9350**
	peopleInShade	0.9462	**0.9588**	0.9506	**0.9575**	0.9714	**0.9792**	0.9532	**0.9568**	0.9766	**0.9829**
thermal	corridor	0.8196	**0.8538**	0.8415	**0.9063**	0.8944	**0.9177**	0.8182	**0.8557**	0.8770	**0.9027**
	diningRoom	0.7675	**0.8217**	0.7125	**0.7942**	0.8818	**0.9117**	0.9066	**0.9220**	0.8808	**0.9211**
	lakeSide	0.3835	**0.4151**	0.3721	**0.4685**	0.6176	**0.7024**	0.7750	**0.8056**	0.6744	**0.7142**
	library	0.9534	**0.9535**	0.6550	**0.6769**	0.7901	**0.8160**	0.9536	**0.9548**	0.8571	**0.8655**
	park	0.5756	**0.8034**	0.5769	**0.8052**	0.7972	**0.8086**	0.8322	**0.8369**	0.7961	**0.8055**
Total average		0.5486	**0.6158**	0.6133	**0.6754**	0.8279	**0.8515**	0.8245	**0.8437**	0.8332	**0.8560**

result with the second highest F-measure is labeled by a green bounding box. It is clear that the proposed FSTF is especially good at finding the missing foreground parts in detection by exploiting the spatiotemporal guidance information. During enhancement, FSTF also exhibits satisfactory edge preserving ability in qualitative results.

Three quantitative experiments are carried out. In the first quantitative experiment, we compare the F-measures of the seven enhancement methods on consecutive frames from five CDnet2012 sequences. The seven enhancement methods are separately tested on the coarse results from GMM, SpkmeansEM, ViBe, and FgSegNet, respectively. The comparisons on F-measure are shown in Fig. 12, in which our FSTF obtains the highest F-measure value for most of the time. In the second quantitative experiment, we apply the proposed FSTF after five extra popular change detection methods including Histogram [45], ED [46], WeSamBE [47], STBM [48], and SuBSENSE [49], on the complete CDnet2012 dataset that contains 31 video sequences. For the five methods, FSTF raise around 2% to 7% in average F-measure. The increases of F-measures by FSTF are in boldface in Table II, and only a few cases are not enhanced. In the third quantitative evaluation, we compare FSTF with GF, CRF, and Li MRF under three metrics (F-measure, MAE/MSE, and PSNR) for improving coarse SpkmeansEM results from 10 representative sequences of CDnet2012 in Table III. For MAE/MSE, the lower the better; and the opposite for both PSNR and F-measure. Our FSTF shows evident superiority in comparisons.

TABLE III

THE QUANTITATIVE COMPARISON AMONG FSTF, GF, CRF, AND LI MRF UNDER THREE METRICS ON TEN SEQUENCES FROM CDNET2012. THE BEST RESULTS ARE IN BOLDFACE

	MAE/MSE					PSNR					F-measure				
	coarse	GF	CRF	Li MRF	FSTF	coarse	GF	CRF	Li MRF	FSTF	coarse	GF	CRF	Li MRF	FSTF
pedestrians	0.0011	0.0013	**0.0009**	0.0012	**0.0009**	29.9193	29.5487	30.7590	30.0164	**31.2361**	0.9543	0.9495	0.9610	0.9504	**0.9635**
office	0.0100	0.0065	0.0094	0.0103	**0.0042**	20.0880	22.2091	20.4434	19.9565	**24.0872**	0.9211	0.9500	0.9260	0.9185	**0.9692**
traffic	0.0445	0.0312	0.0472	0.0368	**0.0189**	16.2829	**27.1510**	14.7679	17.8478	27.1270	0.8004	0.8657	0.8183	0.8386	**0.9269**
boats	0.0174	0.0130	0.0115	0.0160	**0.0092**	17.6351	18.9303	19.4777	17.9878	**20.4986**	0.7923	0.8569	0.8690	0.8025	**0.8965**
overpass	0.0407	0.0296	0.0325	0.0386	**0.0182**	14.8247	16.5067	16.0378	15.1976	**18.0019**	0.7780	0.8419	0.8265	0.7848	**0.9131**
backdoor	0.0147	0.0095	0.0124	0.0146	**0.0090**	18.4758	20.6621	19.4552	18.4531	**20.8051**	0.8967	0.9356	0.9140	0.8961	**0.9403**
bungalows	0.0386	0.0226	0.0221	0.0430	**0.0105**	16.8707	20.2412	19.8561	16.4936	**21.4157**	0.8951	0.9423	0.9430	0.8817	**0.9729**
corridor	0.0419	0.0395	0.0429	0.0431	**0.0224**	14.5445	14.7208	14.3062	14.4487	**17.0810**	0.8071	0.8200	0.8018	0.8002	**0.9062**
dining room	0.0265	0.0209	0.0268	0.0292	**0.0121**	15.9047	17.1009	15.8344	15.4435	**19.5607**	0.8468	0.8831	0.8446	0.8284	**0.9375**
park	0.0215	0.0170	0.0280	0.0183	**0.0156**	16.7769	17.8621	15.6774	17.4902	**18.2594**	0.6500	0.7351	0.4295	0.7159	**0.8048**

B. Parameter Tuning

We take the 735th frame of the "office" sequence as an example for visually presenting the influence of σ on the result of FSTF. As shown in Fig. 13(c)-(g), the grayscale output of FSTF effectively removes isolated false positives and fills the black holes on the foreground. But when the parameter is too small (e.g., 5), the enhancement is not evident enough with black holes on the foreground remains largely unchanged. When the parameter is increased to 25, though the holes are filled, the upper body foreground leaks to the dark door frame, creating a white vertical line on the left side of the person. It can also be observed that the whiteboard background influences the book foreground whose color is also white, making the book to be incorrectly filtered as a background. A larger σ should be used for a complex scene for letting the foreground information easier to aggregate. However, a larger σ also makes it easier for background information to cross target boundary to affect the foreground area. Therefore, one should choose a proper σ to reach a good trade-off in a particular scene. For example, "streetlight" and "traffic" are both complex outdoor traffic sequences, the cameras suffer from jittering caused by wind. The targets in the two sequences are mostly vehicles that have non-uniform colors; e.g., the color of the windscreen and the roof are usually different. Due to specular reflection, the color of the same vehicle may also vary drastically in the video. Therefore, we set $\sigma = 25$ for "streetlight" and "traffic" sequences. The "pedestrians" and "office" sequences depict close-range scenes, and the targets are slowly moving pedestrians. Due to the relatively stable illumination conditions and background color distributions, we choose $\sigma = 15$ for both "pedestrians" and "diningRoom", and $\sigma = 10$ for "office".

For the threshold Th_b, we discover that its value should not be too large for enhancing background-foreground subtraction detection results. Take Fig. 13 for example, we first casually filter frame 735 of "office" with $\sigma = 10$, and then compare the qualitative and quantitative results across five different Th_b thresholds. It is obvious that $Th_b = 20$ can achieve the best visual effect and F-measure at the same time, which enjoys the best trade-off between the smallest area of black holes on the foreground and the lightest leak of the leg foreground to the surrounding background area. It is also interesting to

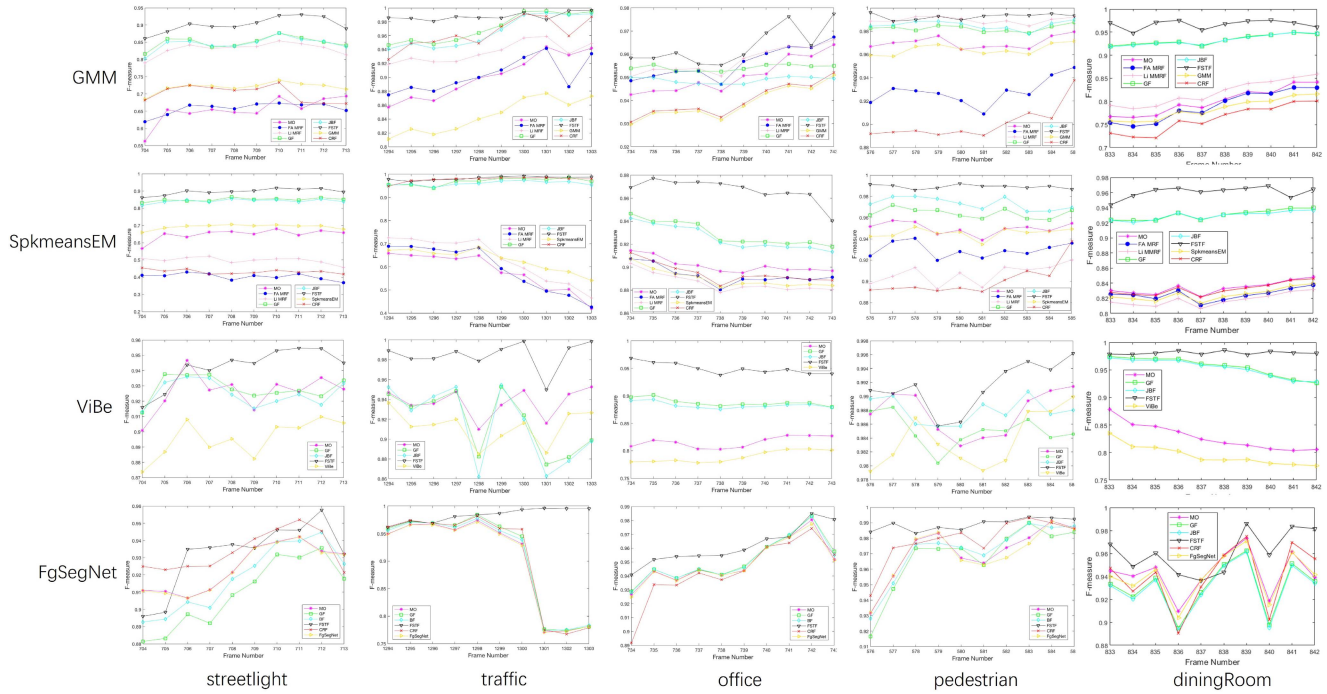


Fig. 12. The F-measure comparisons of different enhancement methods on consecutive detected frames of five CDnet2012 sequences. In each sub-graph, the vertical axis stands for F-measure (the higher the better), and the horizontal axis means frame indices. The 1st row of sub-graphs shows enhancements on coarse GMM detections, the 2nd row shows enhancements on SpkmeansEM results, the 3rd row shows enhancements on ViBe results, and the last row shows enhancements on FgSegNet. The 1st to 5th columns exhibit consecutive enhancement results for "streetlight", "traffic", "office", "pedestrian", and "diningRoom" sequences, respectively. Our FSTF obtains the highest F-measure values for most of the time.

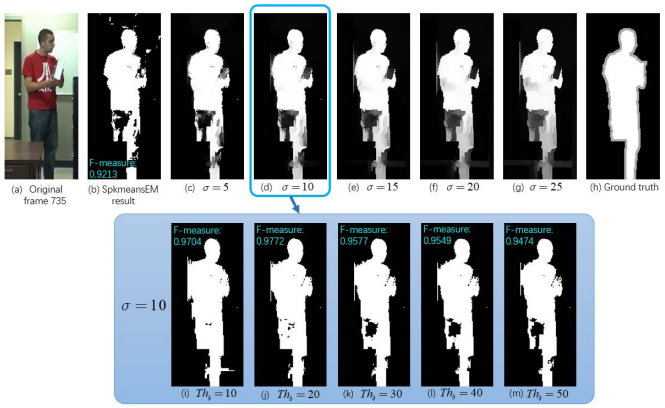


Fig. 13. The results of parameter tuning on frame 735 of the "office" sequence. The first row gives the filtering results for different σ before binarization, respectively. The second row shows the final FSTF results at $\sigma = 10.0$ with different Th_b settings, respectively.

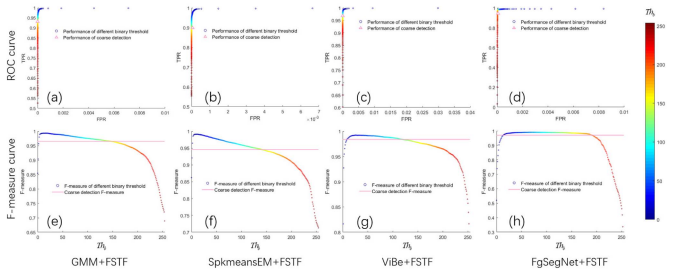


Fig. 14. The ROC curves (first row) and F-measure curves (second row) with varying Th_b for the "pedestrians" sequence under FSTF enhancements at a fixed $\sigma = 15$. For the ROC curves, the point that is closer to the upper-left corner means a better result. And for the F-measure curves, the higher the better.

find out that all the five cases on the second row of Fig. 13 have better F-measure than the original SpkmeansEM result shown by Fig. 13(b). This reveals a fact for our FSTF—the optimal configuration of σ and Th_b may be unique, but the sub-optimal solutions are widespread and easy to find.

In order to quantitatively analyze the influence of Th_b on the result of FSTF, we fix σ to 15.0 and do parameter tuning of Th_b on frames 576-585 of the "pedestrians" sequence after four different change detection methods. The ROC curves and F-measure curves with Th_b varying from 1 to 254 are presented in Fig. 14. The ROC curves of Fig. 14(a)-(c) show that a Th_b ranging from 10 to 50 show satisfactory performances, and the enhancement results generally locate near the upper-left corner of the ROC plot, quantitatively better than the original detection result (labeled by a pink triangle mark). The F-measure curves of Fig. 14(e)-(g) show that choosing Th_b from the interval of [10, 100] can generate far better results comparing to the original detection results, and the enhanced performance peaks with Th_b set around 20. Therefore, we suggest $Th_b = 20$ to be a good choice for FSTF to improve the results obtained by those pixel-wise change detection models such as GMM, SpkmeansEM, and ViBe. The foreground segmentation methods based on deep learning or RPCA usually output over-smoothed foreground regions; therefore Th_b should be adequately large for preventing from further expansion of foregrounds. Fig. 14(d) shows that FSTF obtains best performance when Th_b is set around 100 for FgSegNet. So, we suggest using a larger Th_b (around 100) for those change detection algorithms that tend to generate

TABLE IV

THE AVERAGED F-MEASURE VALUES UNDER DIFFERENT PARAMETER SETTINGS OF FSTF ON TWO CDNET SEQUENCES. THE HIGHTEST VALUE IS IN BOLDFACE, AND THE LOWEST IS UNDERLINED. THE VALUES HIGHER THAN THE COARSE CASE ARE WITH PINK BACKGROUND

"boulevard"

Coarse average F-measure of Subsense [49] = 0.7528					
FSTF	$Th_b = 20$	$Th_b = 40$	$Th_b = 60$	$Th_b = 80$	$Th_b = 100$
$s = 5$	**0.7899**	0.7843	0.7797	0.7737	0.7642
$s = 10$	0.7874	0.7844	0.7775	0.7687	0.7607
$s = 15$	0.7814	0.7813	0.7738	0.7637	0.7524
$s = 20$	0.7741	0.7789	0.7692	0.7566	0.7468
$s = 25$	0.7657	0.7755	0.7638	0.7512	0.7406
$s = 30$	0.7574	0.7713	0.7598	0.7457	<u>0.7347</u>

"canoe"

Coarse average F-measure of Subsense [49] = 0.7922					
FSTF	$Th_b = 20$	$Th_b = 40$	$Th_b = 60$	$Th_b = 80$	$Th_b = 100$
$s = 5$	0.8379	0.8288	0.8217	0.8152	<u>0.8092</u>
$s = 10$	0.8551	0.8440	0.8339	0.8248	0.8161
$s = 15$	0.8638	0.8521	0.8412	0.8305	0.8203
$s = 20$	0.8673	0.8568	0.8448	0.8339	0.8228
$s = 25$	0.8692	0.8597	0.8467	0.8353	0.8246
$s = 30$	**0.8704**	0.8608	0.8480	0.8365	0.8255

over-smoothed detections. In Fig.14(h), we can also see that the F-measure values stay at a high plateau when Th_b is within the range of [10, 150], which means our algorithm does not need very careful parameter tuning.

We also have looked into the stability of performance of FSTF under varying parameters to examine the fact that the sub-optimal parameters for FSTF are widespread. We varied the two parameters σ and Th_b at the same time to test FSTF results after SuBSENSE [49] on two sequences "boulevard" and "canoe" from CDnet2012. The averaged F-measure values under different parameter settings are given in Table IV. The numbers with pink background are higher than the average F-measure of coarse SuBSENSE detections. Table IV shows the parameter selection of FSTF is generally robust because most of the table area is sub-optimal, exhibiting pink background. This table also exhibits smooth landscapes of performance of FSTF under varying parameters. For the "boulevard" sequence, the worst parameter setting obtains an F-measure at 0.7347, which amounts to 93.0% performance of the highest F-measure at 0.7899. For the "canoe" sequence, the worst parameter setting has an F-measure at 0.8092, which also amounts to around 93.0% performance of the highest F-measure at 0.7899.

We have also investigated the influence of parameter k on FSTF. Fig. 15 shows outputs of FSTF with different k on change detection results of the AVSS2007 sequence [50] and the PETS2006 sequence [51]. According to Section II-E, the time of filtering is linear with the number of pixels of the MST. To balance the processing speed and the filtering effect, k should be set to a proper value. We do FSTF with k varying from 1 to 4 and meanwhile keep other parameters fixed in Fig. 15. With k increases, the integrity of the car window on the 1st row and the upper body of the person on the 2nd row improve gradually. It is not difficult to find that FSTF can fully utilize the spatiotemporal information of the same target with a large k. But to make the algorithm running in real time, we take $k = 4$ in all experiments throughout this paper. It should be noted that when $k = 1$, the FSTF degenerates to the original spatial tree filter [39] that searches the MST in the current image only.

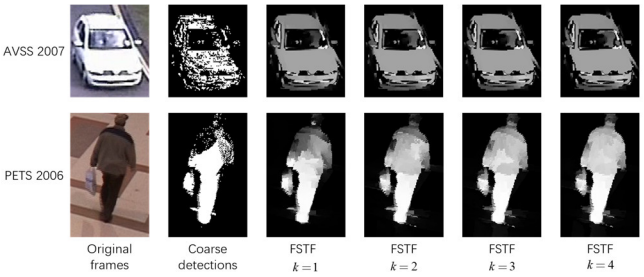


Fig. 15. Influence of k on FSTF results of a frame from AVSS2007 dataset and a frame from PETS2006. With an increasing k, the integrity of the foreground improves.

TABLE V

COMPARISON OF THE AVERAGE TIME COSTS (MILLISECONDS) FOR ENHANCING ONE FRAME ACROSS THE SEVEN ENHANCEMENT ALGORITHMS

Sequence	Resolution	Detection method	MO	FA MRF	Li MRF	JBF	GF	CRF	FSTF
streetlight	320×240	GMM	0.4	17.0	16.3	12.3	7.1	189.9	17.0
		SpkmeansEM	0.4	16.5	17.8	11.8	6.8	189.3	16.0
		ViBe	0.5	\	\	12.1	7.1	\	17.8
		FgSegNet	0.4	\	\	12.1	6.9	201.2	19.9
pedestrians	360×240	GMM	0.5	55.0	29.7	12.9	7.9	228.7	20.5
		SpkmeansEM	0.5	18.2	27.9	13.7	8.6	242.2	22.0
		ViBe	0.4	\	\	13.4	7.7	\	22.1
		FgSegNet	0.4	\	\	13.4	7.8	252.8	21.3
office	360×240	GMM	0.5	18.3	9.2	23.7	9.1	198.1	30.5
		SpkmeansEM	0.5	37.1	18.9	24.7	8.1	189.2	31.7
		ViBe	0.5	\	\	25.8	8.0	\	29.8
		FgSegNet	0.5	\	\	12.8	7.9	215.9	29.6
traffic	320×240	GMM	0.9	34.2	17.3	15.4	14.7	190.9	24.2
		SpkmeansEM	0.5	34.5	17.4	16.2	10.8	184.7	24.8
		ViBe	0.4	\	\	12.7	11.3	\	24.0
		FgSegNet	0.9	\	\	12.1	10.4	196.4	18.8
diningRoom	320×240	GMM	0.4	34.3	24.7	23.0	8.1	175.4	27.0
		SpkmeansEM	0.4	34.5	25.9	22.5	7.8	198.6	26.6
		ViBe	0.4	\	\	22.1	8.1	\	26.5
		FgSegNet	0.4	\	\	23.0	9.1	192.4	27.3

C. Speed

The average time costs for the seven enhancement algorithms on five CDnet sequences are listed in Table V. Each measured time is averaged on 10 frames with 3 repeats. The fastest enhancement algorithm is MO; however, its performance is far from satisfactory. The slowest one is the fully connected CRF. Although the proposed FSTF is slower than GF and JBF, it has the best enhancement performance and has real-time applicability. The processing speed of FSTF is closely related to three aspects: the number of objects in the newest frame, the number of guidance images in the guidance set, and the area of guidance images. Thus, the speeds are different depending on the type of the video stream.

V. DISCUSSION

A. Robustness Against Spatiotemporal Disturbances

Due to the limitation of the change detection algorithms, foreground targets may be over-detected or under-detected in the binary mask. Unstable detection performance may result in object bias, object mismatching in regions of the guidance set because each guidance region has the same position with the bounding box of its corresponding foreground area. In this subsection, we prepare inaccurate input and guidance sets by simulating four different kinds of challenges including over-detection, under-detection, object bias, and mismatching to evaluate the robustness of FSTF against spatiotemporal disturbances. We use PETS2003 and AVSS2007 as test datasets. Fig. 16 and Fig. 17 show the effects of FSTF for enhancing a soccer player's mask and a vehicle mask with disturbances,

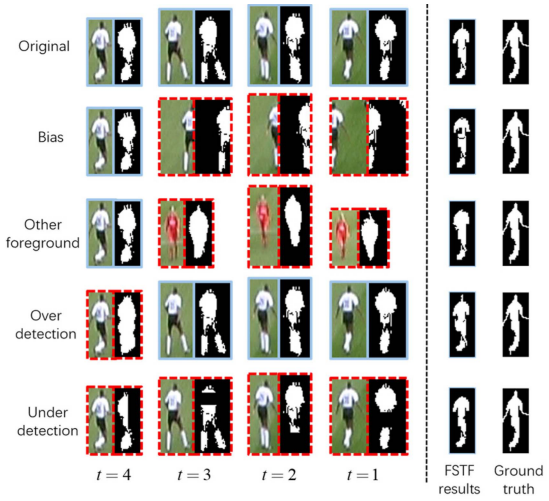


Fig. 16. Robustness performance of FSTF on PETS2003. From the 1st to the 4th columns are the guidance images and the SpkmeansEM coarse detection results for the current time $t = 4$ and three history frames, respectively. The 5th column shows the FSTF results with binarization for the target at $t = 4$ under different types of disturbances. The last column is the ground truth for $t = 4$. The 1st row is the baseline without disturbances. The 2nd row shows the enhancement given disturbance from foreground position bias in the guidance set. The 3rd row shows the case of disturbance from an incorrect foreground in the guidance set. The 4th row shows the case of disturbance from over-detection in the current frame. The last row simulates the disturbance from under-detection in all frames, in which, at time $t = 4$, the right half of the target has an occlusion; the upper body of the character has an occlusion at $t = 3$; occlusion at different positions in the lower part of the target at $t = 2$ and $t = 1$. The blue solid bounding box means no disturbance, and the red dashed bounding box means the opposite.

respectively. We use uniform parameters for all tests in this subsection.

For the PETS2003 example, the original four consecutive guidance images and their corresponding inputs of the same target are shown in the 1st row of Fig. 16 as the baseline. Our FSTF significantly improves the detection result at frame $t = 4$, and the enhanced binary region is closer to the ground truth comparing to the original detection. The 2nd row simulates the disturbance of object bias. At frames $t = 3, 2, 1$, the detection box deviates from the real position of the player, causing only half of the real foreground area to remain in the guidance region. Though the result of FSTF for this case is not as good as the first row, the overall shape of the foreground is still intact after enhancement. The 3rd row simulates the mismatch in searching for historical detections of the same object. At frames $t = 3, 2, 1$, we replace the correct target with another red soccer player; the FSTF result is still satisfactory. The 4th row simulates a sudden over-detection. Although the foreground area at time $t = 4$ increases suddenly to almost twice the area of the ground truth, FSTF successfully corrects its boundary by utilizing history information. The 5th row simulates the under-detection case that is usually caused by occlusion. In each frame, different parts of the foreground masks are intentionally covered with black background patches. However, our FSTF still exhibits strong enhancement and the final binary result is still close to the ground truth.

The original detection results of SpkmeansEM for the AVSS2007 sequence and the corresponding guidance images

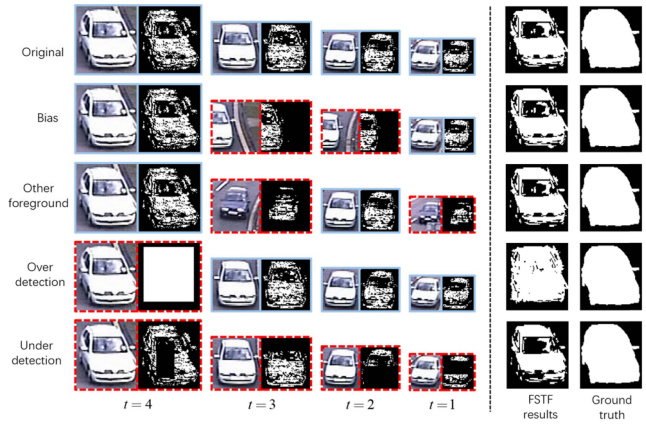


Fig. 17. Robustness performance on AVSS2007 PV sequence. From the 1st to the 4th columns are the guidance images and the SpkmeansEM coarse detection results for the current time $t = 4$ and three history frames, respectively. The 5th column shows the FSTF results after binarization for the current time at $t = 4$ under four types of disturbances. The last column is the ground truth for $t = 4$. The 1st row is the baseline case without disturbance. The 2nd row shows the enhancement given disturbance from foreground position bias in the guidance set. The 3rd row shows the case of disturbance from an incorrect foreground in the guidance set. The 4th row shows the case of disturbance from over-detection in the current frame. The last row simulates the disturbance from under-detection in all frames. The blue solid bounding box means no disturbance, and the red dashed bounding box means the opposite.

Fig. 18. Several FSTF enhancements on DSS results from the SOD dataset. The 1st column lists three original images: "bear", "church", and "sea anemone". The 2nd column shows the saliency detection results by DSS algorithm. The 3rd column shows the FSTF enhancement results with DSS inputs. The 4th column is the ground truth. We use $\sigma = 5$, $k = 4$ for filtering of the "bear" image;$\sigma = 25$, $k = 4$ for the "church" image; and $\sigma = 25$, $k = 4$ for the "sea anemone" image.

of four frames are shown by the 1st row of Fig. 17 as the baseline. We also have tested four kinds of disturbances for this example. On the 2nd row of Fig. 17, only half of the real vehicle area remains in the guidance region at frames $t = 3, 2, 1$, and the result of FSTF is only slightly inferior to the baseline situation in the region of the front windshield of the vehicle. The 3rd row simulates mismatch in searching historical detections, and the FSTF result is still satisfactory in this case. The 4th row simulates over-detection by enlarging the vehicle foreground mask to a big rectangle area at $t = 4$. Although the over-detection completely buries the real vehicle

boundary, the FSTF corrects its boundary to the ground truth by utilizing history information. The 5th row simulates the under-detection case by covering different parts of the foreground masks with black patches. Our FSTF still exhibits a satisfactory result. In summary, the four different disturbances only slightly affect the final result of FSTF comparing to the baseline, showing the strong robustness of FSTF.

B. Application to Saliency Detection

Saliency detection algorithms generate a grayscale saliency map from an image by extracting the information that is essentially important. The brighter a pixel of the saliency map is, the more salient it is. FSTF can also be applied to enhance the results of saliency detection. Although a saliency detection algorithm normally processes a single image rather than a video sequence, the case is akin to using a fast speed shutter camera to capture a series of images at almost the same time. By creating k duplicates of an image and then stacking them to form a guidance set $\mathcal{I}$, we can simulate a very narrow spatiotemporal volume for applying FSTF. We test our enhancement method on two saliency detection algorithms: Deeply Supervised Salient (DSS) [52] and saliency detection using Global Components (GC) representation [53]. For DSS, we first use the trained model by the authors to carry out saliency detection on 100 images from the SOD [54] dataset, and then apply FSTF to enhance the results. For the saliency detection of GC, we first use the original code of the paper to conduct detection on 1000 images from the MSRA_B [55] dataset, and then apply FSTF to enhance the saliency maps thereafter. Fig. 18 shows the qualitative results for enhancing DSS on several SOD images. We intentionally enlarge two areas on the "bear" image (1st row) to show the effectiveness of FSTF in recovering the actual boundary of the salient object; the FSTF result outlines the hairy silhouette of the bear better than the coarse DSS result. On the "church" image (2nd row of Fig. 18), FSTF successfully recovers the cross on the church roof (magenta box) and identifies the sky background area through the window (green box). On the "sea anemone" image (3rd row), our FSTF significantly improves the boundary of the sea anemone. Fig. 19 shows the qualitative results for enhancing GC on several MSRA_B images. The improved results by FSTF are closer to the ground truth images than the coarse GC detections.

We also have quantitatively evaluated the FSTF-enhanced saliency detections. As both of the original and the enhanced saliency map are grayscale, we hereby use MAE as the measure. In FSTF enhancements, all parameters are fixed. σ is set to 10.0, and k is set to 4. The coarse DSS results of the SOD dataset obtains a MAE of 0.1439, and the FSTF-enhanced DSS has a MAE of 0.1437. The coarse GC results of the MSRA_B dataset obtains a MAE of 0.1019, and the MAE of the FSTF-enhanced GC decreases to 0.1017. The improvements by FSTF on the two quantitative saliency tests are less significant than improvements on change detection cases in the former section. This is probably because the ground truths in SOD and MSRA_B are not perfectly labeled, and many labeled object masks lose boundary details.

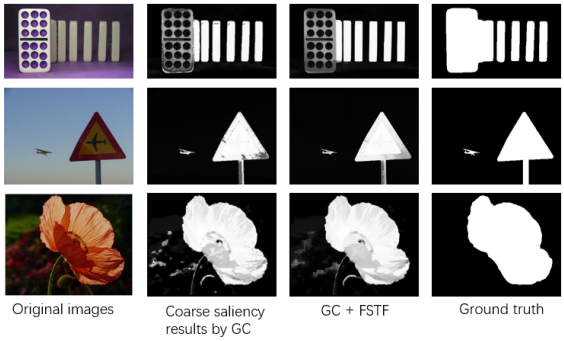


Fig. 19. Several FSTF enhancements on GC results from the MSRA_B dataset. The 1st column lists three original images: "dominoes", "signpost", and "flower". The 2nd column shows the saliency detection results generated by the GC algorithm. The 3rd column shows the FSTF enhancement results with GC inputs. The 4th column is the ground truth. We use $\sigma = 10$, $k = 4$ for filtering of the "dominoes" image; $\sigma = 25$, $k = 4$ for the "signpost" image; and $\sigma = 25$, $k = 4$ for the "flower" image.

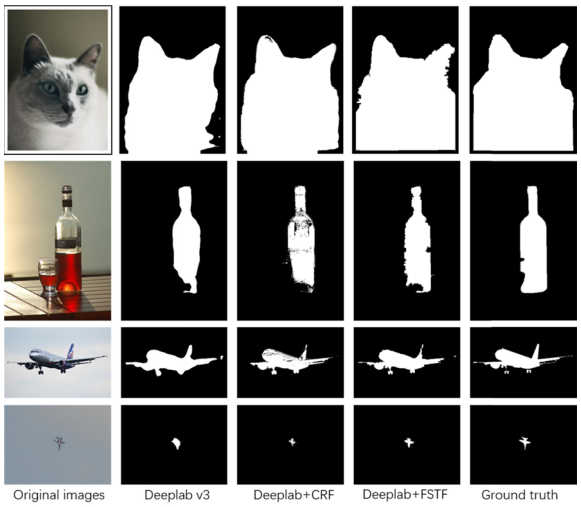


Fig. 20. The comparison of CRF enhancement and the proposed method (FSTF) on Deeplabv3 semantic image segmentation results. From top to down are the "cat", "wine", "plane", and "fighter" images. From left to right are the original images, Deeplabv3 coarse segmentation results, CRF enhancements on Deeplab results, FSTF enhancements on Deeplab results, and the ground truth.

C. Application to Semantic Image Segmentation

Our algorithm can also improve the results of semantic image segmentation; its improvement is superior than the most popular post-processing method—CRF in many tests we have performed. Similar to the way of improving a saliency map in the previous subsection, k copies of the same guidance image are stacked to form a spatiotemporal volume for applying FSTF to the semantic segmentation. We test the enhancement on the results of DeepLabv3 [32] on the PASCAL VOC 2012 [56] dataset. For the convenience of visualization, the foreground targets are uniformly labeled by white pixels in the masks, regardless of their semantic classes. The parameters of FSFT are fixed as $k = 4$, $\sigma = 20$, and $Th_b = 100$. We compare the FSTF results with the Fully-connected CRF results. The parameters of CRF are selected according to [32]. Several qualitative comparisons are shown in Fig. 20, in which we can see that both CRF and FSTF have boundary recovery

effect on the original segmentations. However, our method seems to be better at finding overlooked foreground patches. For the "cat" image on the 1st row of Fig. 20, FSTF finds the body of the cat more completely than CRF. As shown on the 2nd row of Fig. 20, FSTF outcompetes CRF by recovering the missing lower-left part of the bottle. In the 3rd example "plane", CRF and FSTF both improves the boundary of the coarsely segmented plane; but FSTF accurately recovers the thin boundary of the left wing and correct shapes of four wheels. On the 4th row of Fig. 20, the FSTF result for the "fighter" image is closer to the ground truth than the CRF counterpart.

D. Integrating FSTF With Other Local Filters

As we have summarized in the introduction, the local filters possess strong local edge-preserving ability. FSTF considers long-range correlation between any pair of pixels on the spatiotemporal domain. The combination of a local filter and a global filter can be a nice attempt because it may benefit from both sides. In [57], we present an integrated filter that comprises a weighted spatiotemporal tree filter and a weighted guided image filter. The integrated filter even outperforms the filter that focuses on either pure FSTF or pure local GF in some cases (See the ROC curves under different weight settings in Fig. 10 of [57]).

VI. Conclusion

In this paper, we propose a novel Fast Spatiotemporal Tree Filter (FSTF) that makes full use of foreground information for the current frame and history frames at the same time. Differently from local image filters such as bilateral filter [34] and guided image filter [38], FSTF extends the original tree filter [39] from the spatial domain to the spatiotemporal domain while keeping the global filtering ability. FSTF is a purely unsupervised method. The analysis based on graph theory shows it has linear time complexity. Experiments on some challenging video sequences demonstrate the superiority of the proposed FSTF over other state-of-the-art enhancement methods on both qualitative and quantitative aspects. FSTF also has broad applicability on enhancing change detections, saliency detections, and even semantic segmentations.

In the future, we plan to endow FSTF with the ability to simultaneously enhance multiple object masks with different class labels on the same image.

Acknowledgment

D. Li thanks Caigang Hu for his timely and substantial help during the revision of this paper.

References

[1] B.-H. Chen, L.-F. Shi, and X. Ke, "A robust moving object detection in multi-scenario big data for video surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 4, pp. 982–995, Apr. 2019.

[2] C. Lin, B. Yan, and W. Tan, "Foreground detection in surveillance video with fully convolutional semantic network," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 7–10.

[3] J. Liu, M. Gong, K. Qin, and P. Zhang, "A deep convolutional coupling network for change detection based on heterogeneous optical and radar images," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 3, pp. 545–559, Mar. 2018.

[4] W. Gu, Z. Lv, and M. Hao, "Change detection method for remote sensing images based on an improved Markov random field," *Multimedia Tools Appl.*, vol. 76, no. 17, pp. 17719–17734, Sep. 2017.

[5] Y. Zhong, A. Ma, Y. S. Ong, Z. Zhu, and L. Zhang, "Computational intelligence in optical remote sensing image processing," *Appl. Soft Comput.*, vol. 64, pp. 75–93, Mar. 2018.

[6] Z. Zhu and C. E. Woodcock, "Continuous change detection and classification of land cover using all available landsat data," *Remote Sens. Environ.*, vol. 144, pp. 152–171, Mar. 2014.

[7] B. DeVries, M. Decuyper, J. Verbesselt, A. Zeileis, M. Herold, and S. Joseph, "Tracking disturbance-regrowth dynamics in tropical forests using structural change detection and landsat time series," *Remote Sens. Environ.*, vol. 169, pp. 320–334, Nov. 2015.

[8] V. Nika, P. Babyn, and H. Zhu, "Change detection of medical images using dictionary learning techniques and principal component analysis," *J. Med. Imag.*, vol. 1, no. 2, Sep. 2014, Art. no. 024502.

[9] R. Jain and H.-H. Nagel, "On the analysis of accumulative difference pictures from image sequences of real world scenes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-1, no. 2, pp. 206–214, Apr. 1979.

[10] T. Aach and A. Kaup, "Bayesian algorithms for adaptive change detection in image sequences using Markov random fields," *Signal Process., Image Commun.*, vol. 7, no. 2, pp. 147–160, Aug. 1995.

[11] C. Stauffer and E. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 1999, pp. 2246–2252.

[12] Z. Zivkovic, "Improved adaptive Gaussian mixture model for background subtraction," in *Proc. Int. Conf. Pattern Recognit.*, 2004, pp. 28–31.

[13] D. Li, L. Xu, and E. D. Goodman, "Illumination-robust foreground detection in a video surveillance system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 10, pp. 1637–1650, Oct. 2013.

[14] A. M. Elgammal, D. Harwood, and L. S. Davis, "Non-parametric model for background subtraction," in *Proc. Eur. Conf. Comput. Vis.*, 2000, pp. 751–767.

[15] 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.

[16] E. J. Candes, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *J. ACM*, vol. 58, no. 3, pp. 1–37, 2011.

[17] X. Zhou, C. Yang, and W. Yu, "Moving object detection by detecting contiguous outliers in the low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 3, pp. 597–610, Mar. 2013.

[18] Z. Gao, L.-F. Cheong, and Y.-X. Wang, "Block-sparse RPCA for salient motion detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 1975–1987, Oct. 2014.

[19] X. Liu *et al.*, "Background subtraction using spatio-temporal group sparsity recovery," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 8, pp. 1737–1751, Aug. 2018.

[20] L. A. Lim and H. Y. Keles, "Foreground segmentation using convolutional neural networks for multiscale feature encoding," *Pattern Recognit. Lett.*, vol. 112, pp. 256–262, Sep. 2018.

[21] L. Yang, Y. Wang, X. Xiong, J. Yang, and A. K. Katsaggelos, "Efficient video object segmentation via network modulation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6499–6507.

[22] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar, "Changedetection.Net: A new change detection benchmark dataset," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1–8.

[23] T. Brox and J. Malik, "Object segmentation by long term analysis of point trajectories," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 282–295.

[24] F. Flohr and D. Gavrila, "PedCut: An iterative framework for pedestrian segmentation combining shape models and multiple data cues," in *Proc. Brit. Mach. Vis. Conf.*, 2013, pp. 66.1–66.11.

[25] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. London, U.K.: Pearson, 2010.

[26] P. Soille, "Opening and closing," in *Morphological Image Analysis: Principles and Applications*. Berlin, Germany: Springer, 1999, pp. 89–127.

[27] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-6, no. 6, pp. 721–741, Nov. 1984.

[28] J. M. McHugh, J. Konrad, V. Saligrama, and P.-M. Jodoin, "Foreground-adaptive background subtraction," *IEEE Signal Process. Lett.*, vol. 16, no. 5, pp. 390–393, May 2009.

[29] S. Gould, J. Rodgers, D. Cohen, G. Elidan, and D. Koller, "Multi-class segmentation with relative location prior," *Int. J. Comput. Vis.*, vol. 80, no. 3, pp. 300–316, Dec. 2008.

[30] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Proc. ICCV*, 2009, pp. 670–677.

[31] P. Krähenbuhl and V. Koltun, "Efficient inference in fully connected CRFs with Gaussian edge potentials," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 109–117.

[32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 834–848, Apr. 2018.

[33] S. Zheng *et al.*, "Conditional random fields as recurrent neural networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 1529–1537.

[34] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 1998, pp. 839–846.

[35] S. Perreault and P. Hebert, "Median filtering in constant time," *IEEE Trans. Image Process.*, vol. 16, no. 9, pp. 2389–2394, Sep. 2007.

[36] F. Porikli, "Constant time O(1) bilateral filtering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.

[37] Q. Yang, N. Ahuja, and K.-H. Tan, "Constant time median and bilateral filtering," *Int. J. Comput. Vis.*, vol. 112, no. 3, pp. 307–318, May 2015.

[38] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[39] Q. Yang, "Stereo matching using tree filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 834–846, Apr. 2015.

[40] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama, "Digital photography with flash and no-flash image pairs," in *Proc. ACM Siggraph*, 2004, pp. 664–672.

[41] M. Dehmer, *Structural Analysis of Complex Networks*. 2011, pp. 49–72.

[42] H. Wiener, "Structural determination of paraffin boiling points," *J. Amer. Chem. Soc.*, vol. 69, no. 1, pp. 17–20, 1947.

[43] B. Mohar, "Eigenvalues, diameter, and mean distance in graphs," *Graphs Combinatorics*, vol. 7, no. 1, pp. 53–64, Mar. 1991.

[44] A. Fog, "Optimizing software in C++: An optimization guide for windows, Linux and Mac platforms," Tech. Univ. Denmark, Lyngby, Denmark, Tech. Rep., Jan. 2020.

[45] J. Zheng, Y. Wang, N. L. Nihan, and M. E. Hallenbeck, "Extracting roadway background image:Mode-based approach," *Transp. Res. Rec. J. Transp. Res. Board.*, vol. 1944, no. 1, pp. 82–88, 2006.

[46] P.-M. Jodoin, "Comparative study of background subtraction algorithms," *J. Electron. Imag.*, vol. 19, no. 3, Jul. 2010, Art. no. 033003.

[47] S. Jiang and X. Lu, "WeSamBE: A weight-sample-based method for background subtraction," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 9, pp. 2105–2115, Sep. 2018.

[48] M. Chen, Q. Yang, Q. Li, G. Wang, and M. Yang, "Spatiotemporal background subtraction using minimum spanning tree and optical flow," in *Proc. ECCV*, 2014, pp. 521–534.

[49] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, "SuBSENSE: A universal change detection method with local adaptive sensitivity," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 359–373, Jan. 2015.

[50] (Sep. 2007). *IEEE Int. Conf. Advanced Video Signal Based Surveillance Dataset*. [Online]. Available: http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007.html

[51] (Jun. 2006). *9th IEEE Int. Workshop Performance Evaluation Tracking Surveillance Dataset*. [Online]. Available: http://pets2006.net/

[52] M. Cheng, X. Hu, A. Borji, Z. Tu, and P. Torr, "Deeply supervised salient object detection with short connections," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 2, pp. 353–367, Apr. 2011.

[53] M. Cheng, J. Warrell, W. Y. Lin, S. Zheng, V. Vineet, and N. Crook, "Efficient salient region detection with soft image abstraction," in *Proc. ICCV*, 2013, pp. 1529–1536.

[54] V. Movahedi and J. H. Elder, "Design and perceptual validation of performance measures for salient object segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop*, Jun. 2010, pp. 49–56.

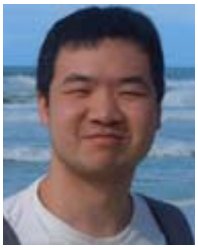
[55] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*. [Online]. Available: http://arxiv.org/abs/1706.05587

[56] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserma, "The PASCAL visual object classes challenge: A retrospective," *Int. J. Comput. Vis.*, vol. 111, pp. 98–136, Jun. 2014.

[57] D. Li, S. Yan, X. Cai, Y. Cao, and S. Wang, "An integrated image filter for enhancing change detection results," *IEEE Access*, vol. 7, pp. 91034–91051, 2019, doi: 10.1109/ACCESS.2019.2927255.

Dawei Li (Member, IEEE) received the B.Eng. degree in automation and the Ph.D. degree in control theory and control engineering from Tongji University, Shanghai, China, in 2006 and 2013, respectively. From 2009 to 2010, he was a Visiting Researcher with Genetic Algorithms Research and Applications Group (GARAGe), Michigan State University. From 2013 to 2015, he held a postdoctoral position at the Department of Computer Sciences and Technology, Tongji University. He is currently an Associate Professor with the College of Information Sciences and Technology, Donghua University, Shanghai. His current research interests include image processing, pattern recognition, plant phenotyping, and agricultural engineering. In 2010, he was bestowed the Finalist for the Best Paper Award at the 11th IEEE's International Conference on Control, Automation, Robotics and Vision.

Siyuan Yan received the B.S. degree in electrical engineering and automation from Donghua University, Shanghai, China, in 2017, where she is currently pursuing the M.S. degree in control theory and control engineering. Her research interests include image processing and machine learning.



Mingbo Zhao (Senior Member, IEEE) received the Ph.D. degree in computer engineering from the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in January 2013. He was with the City University of Hong Kong as a Postdoctoral Researcher. He is currently a Full Professor at Donghua University, Shanghai, China. He has authored or coauthored over 50 technical articles published at prestigious international journals and conferences, including the IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the *ACM Transactions on Intelligent Systems and Technology*, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, and *Pattern Recognition*. His current research interests include pattern recognition and machine learning.

Tommy W. S. Chow (Fellow, IEEE) received the B.Sc. (Hons.) and Ph.D. degrees in electrical and electronic engineering from the University of Sunderland, Sunderland, U.K., in 1984 and 1988, respectively.

He is currently a Professor with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong. His current research interests include neural networks, machine learning, fault diagnosis, and documents analysis. He has authored or coauthored over 190 international technical journal articles related to his research, five book chapters, and one book.

Dr. Chow was a recipient of the Best Paper Award at the 2002 IEEE Industrial Electronics Society Annual Meeting in Seville, Spain. He was the Chairman of the Control Instrumentation and Automation Division, Hong Kong Institution of Engineers, from 1997 to 1998. He was the Guest Editor of *Neural Computing and Applications* on the 2010 Special Issue on The Emerging Applications of Neural Networks. He is currently an Associate Editor of the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS and *Neural Processing Letters*. Under the Google Scholar, his works have received a citation of over 6000.